CA IDMS - 19.0
Administrating Security for IDMS

Date: 10-Jan-2018
This Documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the "Documentation") is for your informational purposes only and is subject to change or withdrawal by CA at any time. This Documentation is proprietary information of CA and may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA.

If you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION “AS IS” WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with “Restricted Rights.” Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2017 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.
# Table of Contents

**CA IDMS Centralized Security Overview** ............................................. 20
  - Architecture ....................................................................................... 21
  - The SRTT ......................................................................................... 22
  - External Security Enforcement .......................................................... 23
  - Internal Security Enforcement ........................................................... 24
  - Security Definitions ........................................................................... 24
  - Runtime Security Processing .............................................................. 26

**Security Application Programming Interface** .................................. 27

**Activating CA IDMS Security** .......................................................... 27
  - Installation Defaults ........................................................................... 28
  - Creating Security Definitions .............................................................. 29
  - Activating Security ............................................................................ 30
  - How to Generate the SRTT ................................................................. 32
  - Dynamic Security Refresh .................................................................. 34

**Using External Security** .................................................................. 35
  - SRTT Requirements ........................................................................... 35
  - Constructing an External Resource Name ........................................... 36
  - Defining External Signon Security ...................................................... 40
    - Defining SRTT Entries ...................................................................... 40
      - Example for CA Top Secret (TSS) ............................................... 40
      - Example for CA ACF2 .................................................................. 40
    - Optionally Defining PassTickets ..................................................... 41
      - Example for CA Top Secret (CA TSS) ......................................... 41
      - Example for CA ACF2 .................................................................. 42
      - Example for IBM RACF ............................................................... 42
    - External Signon Security Checking ................................................ 43
  - External Database Security Considerations ..................................... 43
  - Identifying Authorities to the External System ................................. 43
Securing Tables ......................................................................................................................... 114
Securing Access Modules ........................................................................................................ 116
Securing the Dictionaries and the User catalog ..................................................................... 118

Securing Application Dictionary Resources ............................................................................ 121
Application Dictionary ............................................................................................................. 121
Securing the Dictionary As a Database .................................................................................. 121
Signon To the Dictionary ......................................................................................................... 122
Compiler Security Within the Dictionary .............................................................................. 122
  Checking Compiler Security ................................................................................................. 124
  Checking Registration Override Security ............................................................................. 124
  Checking Verb Security ...................................................................................................... 125
  Checking Component Security ............................................................................................ 126

Syntax for Assembler Macros ................................................................................................. 127
CTABGEN ................................................................................................................................. 128
  Syntax ................................................................................................................................. 128
  Parameters .......................................................................................................................... 128
  Usage ................................................................................................................................. 128
    Coding Considerations ...................................................................................................... 129
    Examples .......................................................................................................................... 130
  DCMT Command Codes ...................................................................................................... 131
UTABGEN ................................................................................................................................. 147
  UTABGEN Syntax ................................................................................................................ 147
  UTABGEN Parameters ........................................................................................................ 147
  UTABGEN Usage ................................................................................................................ 148
  UTABGEN Utility Command Codes .................................................................................. 149
GTABGEN ................................................................................................................................. 151
  GTABGEN Syntax ................................................................................................................ 151
  GTABGEN Parameters ........................................................................................................ 151
  GTABGEN Usage ................................................................................................................ 152
SECHECK ................................................................................................................................. 153
  SECHECK Authorization .................................................................................................... 153
  SECHECK Syntax ................................................................................................................ 153
  SECHECK Parameters ....................................................................................................... 154
  SECHECK Usage ................................................................................................................ 159
SECRRTT ................................................................................................................................. 160
  SECRRTT Authorization ..................................................................................................... 161
  SECRRTT Syntax ................................................................................................................ 161

Administrating Security for IDMS 7
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER USER PROFILE Authorization</td>
<td>187</td>
</tr>
<tr>
<td>ALTER USER PROFILE Syntax</td>
<td>188</td>
</tr>
<tr>
<td>ALTER USER PROFILE Parameters</td>
<td>188</td>
</tr>
<tr>
<td>ALTER USER PROFILE Usage</td>
<td>189</td>
</tr>
<tr>
<td>ALTER USER PROFILE More Information</td>
<td>190</td>
</tr>
<tr>
<td>CREATE GROUP</td>
<td>190</td>
</tr>
<tr>
<td>CREATE GROUP Authorization</td>
<td>190</td>
</tr>
<tr>
<td>CREATE GROUP Syntax</td>
<td>190</td>
</tr>
<tr>
<td>CREATE GROUP Parameters</td>
<td>190</td>
</tr>
<tr>
<td>CREATE GROUP Usage</td>
<td>191</td>
</tr>
<tr>
<td>CREATE GROUP More Information</td>
<td>191</td>
</tr>
<tr>
<td>CREATE USER</td>
<td>191</td>
</tr>
<tr>
<td>CREATE USER Authorization</td>
<td>192</td>
</tr>
<tr>
<td>CREATE USER Syntax</td>
<td>192</td>
</tr>
<tr>
<td>CREATE USER Parameters</td>
<td>192</td>
</tr>
<tr>
<td>CREATE USER Usage</td>
<td>193</td>
</tr>
<tr>
<td>CREATE USER More Information</td>
<td>194</td>
</tr>
<tr>
<td>CREATE USER PROFILE</td>
<td>194</td>
</tr>
<tr>
<td>CREATE USER PROFILE Authorization</td>
<td>194</td>
</tr>
<tr>
<td>CREATE USER PROFILE Syntax</td>
<td>194</td>
</tr>
<tr>
<td>CREATE USER PROFILE Parameters</td>
<td>195</td>
</tr>
<tr>
<td>CREATE USER PROFILE Usage</td>
<td>196</td>
</tr>
<tr>
<td>CREATE USER PROFILE More Information</td>
<td>196</td>
</tr>
<tr>
<td>DROP GROUP</td>
<td>196</td>
</tr>
<tr>
<td>DROP GROUP Authorization</td>
<td>197</td>
</tr>
<tr>
<td>DROP GROUP Syntax</td>
<td>197</td>
</tr>
<tr>
<td>DROP GROUP Parameters</td>
<td>197</td>
</tr>
<tr>
<td>DROP GROUP Usage</td>
<td>197</td>
</tr>
<tr>
<td>DROP GROUP More Information</td>
<td>198</td>
</tr>
<tr>
<td>DROP USER</td>
<td>198</td>
</tr>
<tr>
<td>DROP USER Authorization</td>
<td>199</td>
</tr>
<tr>
<td>DROP USER Syntax</td>
<td>199</td>
</tr>
<tr>
<td>DROP USER Parameters</td>
<td>199</td>
</tr>
<tr>
<td>DROP USER Usage</td>
<td>199</td>
</tr>
<tr>
<td>DROP USER More Information</td>
<td>200</td>
</tr>
<tr>
<td>DROP USER PROFILE</td>
<td>200</td>
</tr>
<tr>
<td>DROP USER PROFILE Authorization</td>
<td>201</td>
</tr>
<tr>
<td>DROP USER PROFILE Syntax</td>
<td>201</td>
</tr>
<tr>
<td>DROP USER PROFILE Parameters</td>
<td>201</td>
</tr>
<tr>
<td>DROP USER PROFILE Usage</td>
<td>201</td>
</tr>
<tr>
<td>DROP USER PROFILE More Information</td>
<td>201</td>
</tr>
<tr>
<td>GRANT Administration Privilege</td>
<td>201</td>
</tr>
<tr>
<td>Syntax for Securing System Resources</td>
<td>211</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>ALTER RESOURCE</td>
<td>213</td>
</tr>
<tr>
<td>Authorization</td>
<td>213</td>
</tr>
<tr>
<td>Syntax</td>
<td>213</td>
</tr>
<tr>
<td>Parameters</td>
<td>213</td>
</tr>
<tr>
<td>Usage</td>
<td>215</td>
</tr>
<tr>
<td>Examples</td>
<td>216</td>
</tr>
<tr>
<td>More information</td>
<td>216</td>
</tr>
<tr>
<td>CREATE RESOURCE</td>
<td>217</td>
</tr>
<tr>
<td>Authorization</td>
<td>217</td>
</tr>
<tr>
<td>Syntax</td>
<td>217</td>
</tr>
<tr>
<td>Parameters</td>
<td>217</td>
</tr>
<tr>
<td>Usage</td>
<td>220</td>
</tr>
<tr>
<td>Examples</td>
<td>221</td>
</tr>
<tr>
<td>More information</td>
<td>221</td>
</tr>
<tr>
<td>DROP RESOURCE</td>
<td>221</td>
</tr>
<tr>
<td>Authorization</td>
<td>221</td>
</tr>
</tbody>
</table>
Syntax for Securing Database Resources ........................................................................ 238

GRANT Access Module Execution Privilege ................................................................. 238
  GRANT Access Module Execution Privilege Authorization ........................................ 239
  GRANT Access Module Execution Privilege Syntax ................................................. 239
  GRANT Access Module Execution Privilege Parameters ........................................ 239
  GRANT Access Module Execution Privilege Usage .................................................. 240
  GRANT Access Module Execution Privilege More Information ............................. 240

GRANT Administration Privilege 1 .............................................................................. 240
  GRANT Administration Privilege Authorization ....................................................... 241
  GRANT Administration Privilege Syntax ............................................................... 241
  GRANT Administration Privilege Parameters ....................................................... 241
  GRANT Administration Privilege More Information .............................................. 242

GRANT All Table Privileges ....................................................................................... 242
  GRANT All Table Privileges Authorization ............................................................. 242
  GRANT All Table Privileges Syntax ....................................................................... 242
  GRANT All Table Privileges Parameters ............................................................... 243
  GRANT All Table Privileges Usage ....................................................................... 243
  GRANT All Table Privileges More Information ..................................................... 244

GRANT Area Access Privileges .................................................................................. 245
  GRANT Area Access Privileges Authorization ....................................................... 245
  GRANT Area Access Privileges Syntax .................................................................. 245
  GRANT Area Access Privileges Parameters .......................................................... 245
  GRANT Area Access Privileges More Information ................................................. 247

GRANT Non-SQL Definition Privilege ...................................................................... 247
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax for Security Display Statements</td>
<td>279</td>
</tr>
<tr>
<td>Notes on DISPLAY/PUNCH Statement Syntax</td>
<td>279</td>
</tr>
<tr>
<td>DISPLAY SYSADMIN PRIVILEGES</td>
<td>281</td>
</tr>
<tr>
<td>DISPLAY PRIVILEGES on a Global Resource</td>
<td>283</td>
</tr>
<tr>
<td>REVOKE Physical Database Definition Privileges</td>
<td>268</td>
</tr>
<tr>
<td>Authorization</td>
<td>269</td>
</tr>
<tr>
<td>Syntax</td>
<td>269</td>
</tr>
<tr>
<td>Parameters</td>
<td>269</td>
</tr>
<tr>
<td>Usage</td>
<td>271</td>
</tr>
<tr>
<td>More Information</td>
<td>271</td>
</tr>
<tr>
<td>REVOKE SQL Definition Privileges</td>
<td>271</td>
</tr>
<tr>
<td>Authorization</td>
<td>272</td>
</tr>
<tr>
<td>Syntax</td>
<td>272</td>
</tr>
<tr>
<td>Parameters</td>
<td>272</td>
</tr>
<tr>
<td>Usage</td>
<td>274</td>
</tr>
<tr>
<td>More Information</td>
<td>274</td>
</tr>
<tr>
<td>REVOKE Table Access Privileges</td>
<td>274</td>
</tr>
<tr>
<td>Authorization</td>
<td>275</td>
</tr>
<tr>
<td>Syntax</td>
<td>275</td>
</tr>
<tr>
<td>Parameters</td>
<td>275</td>
</tr>
<tr>
<td>Usage</td>
<td>276</td>
</tr>
<tr>
<td>More Information</td>
<td>277</td>
</tr>
<tr>
<td>TRANSFER OWNERSHIP</td>
<td>277</td>
</tr>
<tr>
<td>Authorization</td>
<td>277</td>
</tr>
<tr>
<td>Syntax</td>
<td>277</td>
</tr>
<tr>
<td>Parameters</td>
<td>278</td>
</tr>
<tr>
<td>Usage</td>
<td>278</td>
</tr>
<tr>
<td>Syntax</td>
<td>267</td>
</tr>
<tr>
<td>Parameters</td>
<td>267</td>
</tr>
<tr>
<td>More Information</td>
<td>268</td>
</tr>
<tr>
<td>REVOKE Physical Database Definition Privileges</td>
<td>268</td>
</tr>
<tr>
<td>Authorization</td>
<td>269</td>
</tr>
<tr>
<td>Syntax</td>
<td>269</td>
</tr>
<tr>
<td>Parameters</td>
<td>269</td>
</tr>
<tr>
<td>Usage</td>
<td>271</td>
</tr>
<tr>
<td>More Information</td>
<td>271</td>
</tr>
<tr>
<td>REVOKE SQL Definition Privileges</td>
<td>271</td>
</tr>
<tr>
<td>Authorization</td>
<td>272</td>
</tr>
<tr>
<td>Syntax</td>
<td>272</td>
</tr>
<tr>
<td>Parameters</td>
<td>272</td>
</tr>
<tr>
<td>Usage</td>
<td>274</td>
</tr>
<tr>
<td>More Information</td>
<td>274</td>
</tr>
<tr>
<td>REVOKE Table Access Privileges</td>
<td>274</td>
</tr>
<tr>
<td>Authorization</td>
<td>275</td>
</tr>
<tr>
<td>Syntax</td>
<td>275</td>
</tr>
<tr>
<td>Parameters</td>
<td>275</td>
</tr>
<tr>
<td>Usage</td>
<td>276</td>
</tr>
<tr>
<td>More Information</td>
<td>277</td>
</tr>
<tr>
<td>TRANSFER OWNERSHIP</td>
<td>277</td>
</tr>
<tr>
<td>Authorization</td>
<td>277</td>
</tr>
<tr>
<td>Syntax</td>
<td>277</td>
</tr>
<tr>
<td>Parameters</td>
<td>278</td>
</tr>
<tr>
<td>Usage</td>
<td>278</td>
</tr>
<tr>
<td>SYSTEM.RESOURCE</td>
<td>336</td>
</tr>
<tr>
<td>SYSTEM.RESOURCEAUTH</td>
<td>337</td>
</tr>
<tr>
<td>SYSTEM.RESOURCEGROUP</td>
<td>339</td>
</tr>
<tr>
<td>SATTDS</td>
<td>340</td>
</tr>
<tr>
<td>SDUSDS</td>
<td>341</td>
</tr>
<tr>
<td>SECACAB</td>
<td>341</td>
</tr>
<tr>
<td>SECEQU</td>
<td>342</td>
</tr>
<tr>
<td>SECRLST</td>
<td>348</td>
</tr>
<tr>
<td>SECRRTTD</td>
<td>348</td>
</tr>
<tr>
<td>SPRFDS</td>
<td>351</td>
</tr>
<tr>
<td>SRESDS</td>
<td>351</td>
</tr>
<tr>
<td>SRGADS</td>
<td>352</td>
</tr>
<tr>
<td>SRGPDS</td>
<td>353</td>
</tr>
<tr>
<td>SROPDS</td>
<td>353</td>
</tr>
<tr>
<td>SRSADS</td>
<td>354</td>
</tr>
<tr>
<td>SUGPDS</td>
<td>355</td>
</tr>
<tr>
<td>SUSRDDS</td>
<td>355</td>
</tr>
<tr>
<td>SUSRDS</td>
<td>356</td>
</tr>
</tbody>
</table>

**Privileges Required for Statements** .................................................... 356

- **DDL Statements** ............................................................................ 357
- **SQL Statements** ............................................................................ 358
  - **SQL DDL statements** ................................................................. 359
  - **SQL DML Statements** ............................................................... 360
  - **Access Module Management Statements** ................................. 361

**User-Defined System Security Rules** .................................................. 361

- **Installation Codes** ....................................................................... 362
- **Using Terminal Autotasks** ............................................................. 362
- **Associating an Autotask with Selected Logical Terminals** .......... 363
- **Signon and Signoff Functions for an Autotask** ............................ 364
- **Associating Terminals with Devices** ............................................ 365
- **Checking Authority to Access a Particular Terminal** .................. 366
- **Design Considerations** ................................................................. 366
Administrating Security for IDMS

In the data processing environment, security administration, whether performed by a full-time security administrator in a large shop or by a DBA in a small shop, is a vital component of corporate success.

Why Secure Your System?
You secure your system to do the following:

- Protect confidential information from deliberate or accidental exposure
- Maintain the integrity of your corporate databases and dictionaries
- Prohibit or deter unauthorized access
- Meet corporate and departmental security standards
- Fulfill government and Department of Defense (DOD) requirements
- Adhere to privacy laws

Security Strategy
A comprehensive corporate security strategy must account for all types of physical and electronic access to systems, including the following:

- Physical access to the computer room
- Electronic access to the computer room (such as by dial-in lines)
- Access to hardware
- Access to software
- Access to corporate databases
- Access to applications
- Access to production, test, and quality assurance (QA) systems
- Access to data sets

A strategy that has not considered all types of authorized and unauthorized access is open to intentional and accidental corruption.

Installing and Implementing Security
A security system needs to be installed and then implemented. A common approach to organizing this process is to designate the following:
A security administrator who manages and coordinates overall information security for the site

A team that helps the security administrator plan for, install, and implement the security system

CA IDMS Centralized Security Administration

The security approach described in the previous section is well suited for CA IDMS centralized security administration. Three important reasons are as follows:

- CA IDMS centralized security can interface with an external security software system to protect CA IDMS resources.

- All CA IDMS authorities derive from the absolute authority of the security administrator.

- You can easily delegate to DCAs and DBAs the set of authorities they need to administer security on resources that are specific to systems and databases.

For more information, see the following topics:
CA IDMS Centralized Security Overview

This section describes the purposes of centralized security and defines the CA IDMS security domain. The purposes of CA IDMS centralized security are to provide the following:

- A system for protecting CA IDMS resources when an external security system is not available or not used to protect CA IDMS resources.

- A system for protecting CA IDMS resources that is not administered or enforced with user exits.

- A system that can interface with an external security system, such as CA Top Secret for z/OS or RACF®, for the protection of some CA IDMS resources.

Protection of resources external to CA IDMS is not a purpose of CA IDMS centralized security.

Contents

- Architecture (see page 21)
- The SRTT (see page 22)
- External Security Enforcement (see page 23)
- Internal Security Enforcement (see page 24)
- Security Definitions (see page 24)
- Runtime Security Processing (see page 26)
Security Domain

The CA IDMS security domain is the set of CA IDMS DC and UCF systems and local mode jobs that share a set of user definitions.

If you specify that CA IDMS user validation is to be performed by an external security system, the CA IDMS security is the corporate security domain. If user validation is performed by CA IDMS internal security, the CA IDMS security domain is the set of DC systems that share a user catalog, the repository of CA IDMS user definitions.

Note: For more information on CA IDMS user validation, see Signon Processing (https://docops.ca.com/display/IDMS19/Signon+Processing).

Architecture

This section describes the features and architecture of CA IDMS centralized security.

Architectural Features

CA IDMS centralized security architecture includes the following features:

- Full integration with CA IDMS software
- Availability to both system-supplied and user-written applications executing under the CA IDMS central version, in local mode, and through supported front-end software such as CICS and TSO in the CA IDMS environment
- Support for a distributed, client/server environment
- Compatibility with the Command Facility tool used for CA IDMS data definition
- ANSI-compliant security syntax where possible

Multi-layered Security Scheme

In combination with application security for dictionary resources, this architecture offers multi-layered security for CA IDMS systems, as shown in the following illustration:

Host access security
(CA ACF2 for z/OS, CA Top Secret for z/OS, ...)

CA IDMS centralized security

Application security
The SRTT

This section describes the Security Resource Type Table (SRTT).

Purpose of the SRTT

The Security Resource Type Table (SRTT) is a load module in which you store this information that CA IDMS centralized security needs at runtime:

- Each resource type to be secured.
- The system that will enforce security on the resource (internal or external).
- For resources to be secured externally, information that the external security system needs to service a security check request on the resource.

Using the SRTT

At installation, CA IDMS provides the default RHDCSRTT module. The security option for each resource defined by CA IDMS is set to 'OFF' (no security), and SVCNUM is set to the SVC number specified in the installation parameters.

You modify the RHDCSRTT module with an assembly of the #SECRTT macro. The SRTT is loaded at system start-up, and can be reloaded dynamically using the DCMT VARY NUCLEUS command for the RHDCSRTT module.

The scope of the SRTT is one or more CA IDMS systems, depending on your security scheme.

Security Options by Resource Type

For each resource type you can specify one of these security options in the SRTT:

- EXTERNAL -- Security enforcement for the resource follows rules defined in the external security system.
- INTERNAL -- Security enforcement for the resource follows rules defined to the CA IDMS internal security system.
- OFF -- No security enforcement for the resource (the default at installation).

Generating the SRTT

The SRTT is created by issuing a sequence of #SECRTT macros.

The #SECRTT macro can specify one of four types of action:
• **Initial** -- Denotes the beginning of the SRTT assembly.

• **Entry** -- Specifies a security option for *all* occurrences of a given resource type.

• **Occurrence override** -- Specifies a security option for one or more occurrences of a given resource type that overrides the entry specification for the resource type (not applicable to all resource types).

• **Final** -- Denotes the end of the SRTT assembly.

**Occurrence Overrides**

For an individual occurrence of a database, task, or program, you can override in the SRTT the security option you specify for the corresponding resource type.

For example, you can omit an SRTT entry for the database resource type and create an occurrence override for database PROD specifying internal security. The result will be that any checks on resources associated with database PROD will be routed to internal security, while checks on resources associated with all other databases will not be checked.

⚠️ **Note:** The resource name you give on an occurrence override is treated as a wildcard. In the previous example, all databases in the domain of the SRTT with names beginning 'PROD' are secured internally.

**Performance Consideration**

You may gain a performance advantage by using an override to turn off security for occurrences of a secured resource type. Runtime security processing checks for an occurrence override in the SRTT before checking resource authorizations in the security database.

**External Security Enforcement**

**External Security Specifications**

In each SRTT entry that specifies external security, you define the format of the resource name that will be routed to the external security system in a security check. At runtime, the central security interface uses this information to map the IDMS internal resource name to the external resource name before routing the request for a security check to the external system.

**Standard Security Interface**

CA IDMS centralized security uses IBM’s System Authorization Facility (SAF) as the interface to external security systems. On a security check, IDMS centralized security issues RACROUTE calls providing the names of resource type and resource occurrence being checked and the keyword that equates to the authority needed.

**Security Definitions**
You do not need a user catalog if you plan to protect all CA IDMS resources with an external system. All required definitions would reside in the external security system. The one requirement within CA IDMS would be to build the SRTT with external security specifications for all secured resources.

However, user definitions and user profile definitions are accessed during signon processing if they exist, regardless of how signon is secured. Therefore, you may wish to use the user catalog even if all security checks are routed to the external system. For more information on profiles in signon processing, see Securing User Profiles (see page 67).

Internal Security Enforcement

CA IDMS Internal Security

When an entry for a resource type in the SRTT specifies internal security, only users and groups who have been defined in the user catalog can be granted the privilege to access the resource. You grant privileges with the appropriate GRANT statements.

CA IDMS Internal Security Administration

You can delegate internal security administration by granting to selected users:

- Administration privileges
- The privilege to grant their privileges to other users

You perform CA IDMS internal security administration functions by issuing security authorization statements through the CA IDMS Command Facility. When you do this, CA IDMS does the following:

1. Accepts syntax that specifies the security administration request (for example, a CREATE USER statement).
2. Verifies that you, the issuer of the request, have the authority to issue the request.
3. Updates the data in the appropriate internal security repository.

Security Definitions

Required for Security Checks

When CA IDMS centralized security receives a request for a security check, it first determines from the SRTT whether the resource is secured. If it is, centralized security routes the request to CA IDMS internal security or the external security system, depending on the security option specified for the resource in the SRTT.

The resulting security check accesses security definitions of resources and resource authorizations to determine whether the executing user has the authority to access the resource in the way that is indicated on the security request.

Resource Definitions
Securable resources are the entities in the CA IDMS environment defined by CA IDMS or the user to which you control access.

Securable resources defined by CA IDMS are as follows:

- User
- Group
- User profile
- System
- Signon
- System profile
- Application activity
- Queue (1)
- Access module (runtime) (1)
- Load module (loadable entity) (1)
- Program (load module) (1)
- Task (1)
- Database
- Area (2)
- Run unit (1) (2)
- Access module (definition) (2)
- Non-SQL schema (2)
- SQL schema (2)
- Table (2)
- Database name table
- DMCL

(1) Occurrences of this resource can be grouped in a category using the CREATE RESOURCE statement.

(2) This resource type is secured automatically when the database resource type is secured.

Authorization IDs
An authorization ID identifies a user or group whom you can authorize to access resources.

If the security option for the resource is external, the authorization ID (and the authorities given to it) are defined in the external security system.

If the security option for the resource is internal, the authorization ID (and the privileges granted to it) are defined in the internal security system. You define authorization IDs to CA IDMS with the CREATE USER and CREATE GROUP statements.

Resource Authorization

An authority is the ability to access a resource in a particular way. A resource authorization is an authority that is associated with a resource definition and an authorization ID.

If the security option for a resource is external, resource authorizations are specified in the external security system.

If the security option for a resource is internal, resource authorizations are specified in the internal security system by granting privileges. You give users the privilege to access a resource with a GRANT statement, and you take away the privilege with a REVOKE statement.

Runtime Security Processing

Security Checking

The CA IDMS centralized security facility handles all security checks issued during CA IDMS processing. Security requests are routed to the central security interface to provide uniform validation of requests.

Security Enforcement

The security option that is specified in the SRTT for the resource type or resource type occurrence determines how security is enforced.

Example: You can control the execution of tasks with the external security system, but control access to a particular database with CA IDMS internal security.

If the security option for the resource being checked is external, the request is routed to the external security system. The external security system returns a value to the centralized security interface representing the result of the check.

If the security option for the resource being checked is internal, CA IDMS centralized security verifies that the user holds the required permission.

Centralized Security Diagram

This illustration shows the flow of processing in the CA IDMS centralized security system:
Security Application Programming Interface

**What You can Do**

A user-written application in Assembler language can issue security requests to CA IDMS centralized security. All requests pass a Security Request Block (SRB). A request can be issued from a user mode or a system mode application.

**Security Macros**

You can issue a security request using the following:

- **#SECHECK** -- To check authorization for accessing a resource.
- **#SECSGON** -- To validate the user and sign the user on.
- **#SECSGOF** -- To sign the user off.

**Note:** For more information on how to use security macros, see the following sections:

- #SECHECK (see page 153)
- #SECSGON (see page 173)
- #SECSGOF (see page 170)

**Activating CA IDMS Security**

**Getting Started**
The basic questions in planning your CA IDMS security scheme are:

- What resources will be secured?
- For secured resources, will enforcement be external or internal?
- Will the same security option apply to a given resource across the domain?

To help you answer these questions, you should become familiar with the information in the following sections:

- Using CA IDMS Internal Security (see page 47)
- Securing Database Resources (see page 88)

For more information, see the following topics:

- Installation Defaults (see page 28)
- Creating Security Definitions (see page 29)
- Activating Security (see page 30)
- How to Generate the SRTT (see page 32)
- Dynamic Security Refresh (see page 34)

## Installation Defaults

### Security at Installation

The RHDCSRTT module provided with installation of CA IDMS contains an initialized entry for each resource defined by CA IDMS.

The initial security option for each resource is 'OFF'. This means that at installation, there is no security enforcement through CA IDMS centralized security. Any user can create security definitions and grant privileges. Security information is stored in the security databases but has no effect until security is activated. Furthermore, the SVCNUM parameter is set to the SVC number that has been specified in the installation parameters.

### Note:

The identifier of the user who performed the installation is recorded as the owner of the demonstration database schemas in the schema definitions. Schema ownership affects security processing if security for the DB resource is subsequently activated.

### Preserving the Initialized RHDCSRTT

You should preserve a secure copy of the initialized RHDCSRTT module provided at installation. This ensures that after you activate security, you can turn it off again simply by using the initialized RHDCSRTT module.

To preserve the initialized RHDCSRTT, you can take one of these steps:
Create a backup copy of the initialized RHDCSRTT module that is provided in the installed CA IDMS load library. Store the backup copy in a secure load library and restore it to the CA IDMS installed load library if needed.

Link your modified SRTT into a different load library during initial testing. To activate security, concatenate the load library containing the new SRTT with the CA IDMS load library. To deactivate security, remove the load library containing the new SRTT from the STEPLIB/CDMSLIB concatenation, or rename the RHDCSRTT load module.

Creating Security Definitions

External Security Definitions

If security enforcement for a CA IDMS resource is external, you do not need to store information on the resource in the CA IDMS security database.

Before you create the SRTT entries that contain external security information, be sure to complete the steps necessary to define resources and security rules for resources in the external security system. When you create the SRTT with entries that include SECBY=EXTERNAL, you activate external security enforcement for the resource types specified in those entries.

For external security checking, you must specify, in the #SECRTT entry, the format of the resource name that centralized security should forward to the external security system. You construct the external resource name format in the SRTT to match the format you specified for the resource in the external security system.

Internal Security Definitions

To secure one or more resources internally, you must define users to the user catalog and grant privileges on the resources to users. You can create any or all of these definitions before activating security in the SRTT. Remember, however, that until you activate security for CA IDMS administration privileges, any users signed on to the system can also manipulate internal security definitions (and any user can sign on until signon is secured).

As you plan the sequence of creating the security definitions needed for your security scheme, be aware of these considerations:

- You can add a user to a group in one of these ways:
  - Using the GROUP parameter of the CREATE/ALTER USER statement. This specifies the user’s default group. Only one group can be specified.
  - Using the ADD USER clause of the CREATE/ALTER GROUP statement. You can add the user to any number of groups in this way. You must first create the user.

- You associate a user profile with a user in the CREATE/ALTER USER statement. You can do this before you define the profile; the information is stored, and a warning is issued.

- Profiles may be associated with user identifiers but not group identifiers.
- The scope of a user profile is the domain, but a system profile, like the grant of signon privilege which associates a system profile with a user, is system-specific.

- You can grant privileges on resources defined by CA IDMS to authorization IDs before the authorization IDs are defined; the information is stored, and a warning is issued.

- You can grant privileges on resources defined by CA IDMS to authorization IDs before the resources are defined except for resources specified in the CREATE RESOURCE statement (systems, activities, and categories).

- If you create groups corresponding to sets of privileges that you will grant, the number of CREATE GROUP statements required is likely to be much less than the number of GRANT statements to users that you would otherwise issue.

- In general, it is easier to restrict grants of privilege only to groups because it is easier to grant a set of privileges implicitly to a user by adding the user to the group than it is to grant each privilege individually to the user.

- A group may consist of only one user.

- The group PUBLIC is an authorization ID (defined by the system the first time a privilege is granted to it) to which all users belong by default. An appropriate step to consider is transferring ownership of the demonstration database schemas to PUBLIC.

- Categories allow you to group system resources so that you can grant execution privilege on the category (multiple resources) to a user or a group (multiple users) with one statement.

- A resource occurrence may participate in only one category.

- In general, you ease the administration of internal security if you establish a 1:1:1 correspondence among groups, categories, and execution privileges.

- When you create application activities, name the activities in a way that will allow you to use a wildcard in grants of execution privilege on activities.

- If you plan to activate security for the system dictionary and the user catalog, consider doing it with DB occurrence overrides.

- **Note:** For guidance in planning security for the system dictionary and the user catalog, see Securing the Dictionaries and the User catalog (see page 118).

---

**Activating Security**

**Planning to Activate Security**

Before you modify the initialized SRTT, do the following:

- Designate one system for testing security processing.
• Plan the sequence of activating security options so that you can activate and test one at a time.

• Consider activating the SGON (signon) resource first.

• If you plan to secure signon externally and one or more other resources internally, activate security for administration privileges before you grant signon to users who will not hold administration privileges.

External Security for Signon Processing

To activate external security for signon processing, the #SECRTT assembly must include an entry for resource type SGON.

Note: For more information, see External Signon Security (see page 40).

Planning to Activate Internal Security

Before you activate internal security, you should:

1. Verify that the user catalog (SYSUSER.DDLSEC) area is specified in the startup JCL for the test system.
   If the default access mode to the area, as specified in the DMCL, is not UPDATE, then you must issue a DCMT VARY AREA statement before you update the user catalog.

   Note: For more information on DCMT commands, see the CA IDMS System Tasks and Operator Reference section.

2. At the least, create the following definitions:
   • Users -- defining the users who will hold SYSADMIN privilege is sufficient to begin. Creating a SYSADMIN group is recommended.
   • Signon and SYSADMIN privileges granted to the designated users.
     If you now activate internal security, the designated users will be able to sign on to the CA IDMS system and to administer the security system.

Activating Internal Security

If you have granted signon and SYSADMIN privileges, the logical first step to activate internal security is to secure the signon and SYSADMIN resources. You do this by including these entries in the #SECRTT assembly:

#SECRTT TYPE=ENTRY, RESTYPE=SYSA, SECBY=INTERNAL X
#SECRTT TYPE=ENTRY, X
Securing Security Definitions

A knowledgeable user can access security definitions with local mode access to the user catalog or system dictionary. You can prevent this access by securing these entities as databases and granting privileges on categories of run units.

Note: For more information, see Securing the Dictionaries and the User Catalog (see page 118).

How to Generate the SRTT

The #SECRTT Macro

You generate a new SRTT by using the #SECRTT macro:

1. The first #SECRTT macro initializes SRTT values for all CA IDMS resources. The value of the TYPE parameter on the first macro must be INITIAL.

2. One or more additional #SECRTT macros to override initial values. The value of the TYPE parameter on these macros must be ENTRY if the security option is for a resource type (or OCCURRENCE if the security option is for an individual occurrence of a database, task, or program). If you specify SECBY=EXTERNAL, you must also specify the following:
   - The external resource class that you have defined in the external security system as the equivalent of the CA IDMS resource type.
   - The external resource name format that you have defined in the external security system for identifying the CA IDMS resource.

3. The final #SECRTT indicates that the table is to be generated with values as specified by the preceding macros in the series. The value of the TYPE parameter on the last macro must be FINAL.

#SECRTT Assembly

The table is generated and linked only if each #SECRTT statement in the series assembles without error. If one or more statements receives an error, only a listing results.

Note: For complete documentation of the #SECRTT, see #SECRTT (see page 160).
The RHDCSRRT module should be linked into a secure dataset to prevent unwarranted access to or manipulation of the security system.

#SECRRT Macro Example

```
#SECRRT TYPE=INITIAL,                     X
   ENVNAME=TEST,                     X
   SVCNUM=235

#SECRRT TYPE=OCCURRENCE,                     X
   RESTYPE=DB,                     X
   RESNAME='CUSTDB',                      X
   SECBY=INTERNAL

#SECRRT TYPE=ENTRY,                      X
   RESTYPE=SYST,                      X
   SECBY=INTERNAL

#SECRRT TYPE=ENTRY,                      X
   RESTYPE=SGON,                      X
   SECBY=EXTERNAL,                  X
   EXTCLS='SYSTEM',                   X
   EXTNAME=(RESNAME)

#SECRRT TYPE=ENTRY,                      X
   RESTYPE=SPGM,                      X
   SECBY=EXTERNAL,                  X
   EXTCLS='PROGRAM',                   X
   EXTNAME=(RESNAME)

#SECRRT TYPE=OCCURRENCE,                     X
   RESTYPE=SPGM,                     X
   RESNAME='RHDCBYE',                      X
   SECBY=OFF

#SECRRT TYPE=FINAL
END
```

Notes on the Example

- TYPE=INITIAL begins the SRTT definition.
- ENVNAME=TEST provides a default name qualifier that you can use in constructing an external resource name.
- SVCNUM=235 specifies the SVC number defined at install time.
- TYPE=OCCURRENCE signifies an occurrence override. In this case:
  - Resource checks for database CUSTDB and associated schemas, tables, and access modules will be processed by internal security; all other databases and their resources are unsecured.
  - Resource checks for program RHDCBYE will not be routed to the external system, but resource checks for all other programs will be routed to the external security system.
Resource checks on system resources (RESTYPE=SYST) will be processed by internal security, but signon processing (including password validation) will be processed by the external security system.

TYPE=FINAL ends the SRTT.

Note: For more information on #SECRRT usage and #SECRRT syntax, see Syntax for Assembler Macros (see page 127).

Dynamic Security Refresh

You can make changes to your security scheme and then activate those changes without cycling a CV. After changing security definitions using the #SECRRT macro and reassembling the RHDCSRTT module, you issue existing DCMT commands to vary the RHDCSRTT nucleus module to new copy and reload it.

Benefit

You can respond to changes in your security environment without bringing down a system and cycling a CV. For example, you can change the security mapping for a resource type or you can make changes to category and activity definitions.

What Gets Refreshed

When you reload the RHDCSRTT module, the following security definitions are refreshed and any changes you made to them are immediately implemented:

- Access module table
- Category tables
- Activity and category bit map tables

Signon Security Changes Not Immediately Implemented

Signon and system group security definitions are not refreshed when RHDCSRTT is reloaded; users signed on to the system remain signed on even after the reload. Any changes made to signon and system group security for users signed on to a system when a reload is done, do not take place until those users sign off of the system and then sign on again.

Example

After you change a security scheme and modify the RHDCSRTT module, perform the following to activate the changes:

- Issue the DCMT VARY NUCLEUS syntax to vary module RHDCSRTT to new copy.
Issue the DCMT VARY NUCLEUS RELOAD command to reload the changed (new) RHDCSRTT nucleus module.

The following example shows these commands.

dcmt vary nucleus module rhdcsrcrtt n c

VARY NUCLEUS MODULE RHDCSRTT NEW COPY
IDMS DC283001 V104 USER:ABBTH01 NUCLEUS MODULE RHDCSRTT MARKED TO NEW COPY

VARY NUCLEUS RELOAD
IDMS DC283003 V104 USER:ABBTH01 NUCLEUS MODULE RHDCSRTT RELOADED
IDMS DC283004 V104 USER:ABBTH01 CSA/NUCLEUS VECTOR TABLE UPDATED
FOR NUCLEUS MODULE RHDCSRTT
IDMS DC283007 V104 USER:ABBTH01 SECURITY TABLES REFRESHED SUCCESSFULLY

Note: For more information on the DCMT VARY NUCLEUS command, see the CA IDMS System Tasks and Operator Reference section.

Using External Security

SRTT Requirements

Essential to the Security System

The SRTT is the essential foundation of the CA IDMS security system because a resource is unsecured unless security for it is specified in the SRTT.

To secure a resource externally, you must include information in the SRTT that identifies the resource to the external system. This information must include an external resource class and an external resource name.

For external security, you do not need to create any resource definitions within CA IDMS itself.

SRTT Entries for External Enforcement

You maintain the following information in the SRTT about resources that are secured externally:

- **Resource type** -- A keyword representing a type of resource, such as program, table, or database. Certain keywords are reserved for resource types defined by CA IDMS. You can specify any one-to four-character keyword to define your own resource type as long as the meaning of and rules for the resource type are defined in your external system.

  Note: For keywords reserved by CA IDMS, see #SECRTT (see page 160).

- **Security option** -- Always EXTERNAL, specified in the SECBY= parameter.
- **External resource class** -- The name of the resource type as defined in the external security software.

- **Resource name** (optional) -- A specific occurrence of a resource type (resource types database, task, and program only).

- **External resource name format** -- The format of the resource name as defined in the external security software.

- **Environment name** (optional, specified on the initial #SECRRT macro) -- The name of a CA IDMS processing environment to be associated with the resource.

### Specifying External Resource Class and Name

An external security check on a resource occurrence depends upon an external resource class and external resource name supplied on the entry for the resource type in the SRTT. External resource classes and names specified on occurrence overrides are ignored by the runtime system.

Therefore, you must create an SRTT entry with the external resource class and name for a resource type whether you are securing all occurrences of the resource type externally or only some occurrences.

In the following example, an SRTT entry for tasks is created even though the specified security option is 'OFF'. The purpose of the entry is to provide information needed to perform an external security check on the OPER task, for which external security is specified in the occurrence override that follows.

```
#SECRRT TYPE=ENTRY,
  RESTYPE=TASK,
  SECBY=OFF,
  EXTNAME=(RESTYPE,RESNAME)
  EXTCLS='IDMSTASK'
```

```
#SECRRT TYPE=OCCUR,
  RESTYPE=TASK,
  RESNAME='OPER',
  SECBY=EXT
```

### Constructing an External Resource Name

**How You Do It**

As in the previous example, the external resource name format is specified in the EXTNAME parameter of the #SECRRT macro. In this parameter, you list keywords to represent the fields that comprise the external resource name format.

**Note:** For complete documentation of the macro, see #SECRRT (see page 160).

If you do not specify the EXTNAME parameter, the external resource name by default consists of only the name of the base resource on the security request.
Runtime Usage of EXTNAME Values

At runtime, the external resource name is constructed using the values in the current security request that correspond to the keywords you specified in the EXTNAME parameter.

For example, if the SRTT entry for resource type TABL (table) includes an EXTNAME parameter that specifies (ENVIR,RESTYPE,SCHEMA,RESNAME), the external resource name format for a table is:

\[ \text{environment-name.TABL.schema-name.table-name} \]

The following example represents the actual resource name sent to the external security system using values from the current security request:

\[ \text{PROD.TABL.U.SA.EMPLOYEE} \]

Order of Name Fields

The order of the fields in the external resource name passed to the external security system is determined by the order of the keywords that you list on the EXTNAME parameter of the #SECRRTT entry. For a given set of fields, you can specify any possible order to format the external resource name.

The format of the external resource name defined in the EXTNAME parameter of the SRTT must match the format used to identify the resource in the external security system.

Environment Name Qualifier

The environment name qualifier is significant only when security is external. You specify environment name on the initial #SECRRTT macro.

The environment name distinguishes resources in the domain of the current SRTT from like-named resources in the domain of another SRTT. You specify an environment name if such distinctions are necessary to your security scheme.

For example, you can specify PROD as the environment name in the SRTT that governs production systems. This means that you can qualify the external resource names of resources in production systems with PROD and specify rules for them in the external security system that are different from like-named resources in test systems, which may have different environment names or no environment name.

Thus, if you have a database named EMPDB in both the test and production environments, you can write a security rule in the external system that is applied only when the security check is for EMPDB qualified by 'PROD' (that is, PROD.EMPDB if the external resource name is \[ \text{environment-name. database-name} \], or EMPDB.PROD if the external resource name is \[ \text{database-name.environment-name} \]).

External Resource Name Keywords

This table presents the keywords that you can specify in the EXTNAME parameter of the #SECRRTT macro (the required characters appear in upper case) and the value from the current security request that corresponds to each keyword:
CA IDMS - 19.0

**EXTNAME keyword**  
Value from current security request

**ACTivity**  
Concatenation of application name and application function number. For more information, see #SECRRT (see page 160).

**APPLname**  
*application-name*

**DBName**  
*database-name*

**DDName**  
*dd-name*

**ENVr**  
*environment-name*

**RESName**  
The name of the resource occurrence(1)

**RESType**  
The resource type keyword, from the SRTT (for example, SLOD)

**SCHEma**  
*schema-name* (SQL)

**SSName**  
*subschema-name*

**SYSTem**  
*system-identifier*

**VERSion**  
*version-number*

---

⚠️ **Note:** (1) For RESTYPE ACTI, the RESNAME value is *application-name*.

You can always specify the RESNAME, RESTYPE, and ENVIR keywords in formatting the external resource name. The tables that follow indicate the values of RESNAME and RESTYPE for each resource type and the other keywords available in constructing an external resource name for the resource type.

**Naming Global Resources**

This table presents the keywords that you can use to construct external resource names for global resources:

<table>
<thead>
<tr>
<th>Resource</th>
<th>RESNAME</th>
<th>RESTYPE</th>
<th>Other available keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSADMIN</td>
<td>@RESERVED@</td>
<td>SYSA</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>user-identifier</td>
<td>USER</td>
<td></td>
</tr>
<tr>
<td>Group</td>
<td>group-identifier</td>
<td>GROU</td>
<td></td>
</tr>
<tr>
<td>User profile</td>
<td>profile-name</td>
<td>UPRF</td>
<td></td>
</tr>
</tbody>
</table>

**Naming System Resources**

This table presents the keywords that you can use to construct external resource names for system resources:

<table>
<thead>
<tr>
<th>Resource</th>
<th>RESNAME</th>
<th>RESTYPE</th>
<th>Other available keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCADMIN</td>
<td>@RESERVED@</td>
<td>DCA</td>
<td></td>
</tr>
</tbody>
</table>
### Naming Database Resources

This table presents the keywords that you can use to construct external resource names for database resources:

<table>
<thead>
<tr>
<th>Resource</th>
<th>RESNAME</th>
<th>RESTYPE</th>
<th>Other available keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>database-name</td>
<td>DB</td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>area-name</td>
<td>AREA</td>
<td>DBNAme</td>
</tr>
<tr>
<td>Rununit</td>
<td>program-name</td>
<td>NRU</td>
<td>DBNAme,SSNAme</td>
</tr>
<tr>
<td>SQL schema</td>
<td>schema-name</td>
<td>QSCH</td>
<td>DBNAme</td>
</tr>
<tr>
<td>Non-SQL defined schema</td>
<td>nonsql-schema-name</td>
<td>NSCH</td>
<td>DBNAme,VERSion</td>
</tr>
<tr>
<td>Access module</td>
<td>access-module-name</td>
<td>DACC</td>
<td>DBNAme,SCHEma</td>
</tr>
<tr>
<td>Table</td>
<td>table-name</td>
<td>TABL</td>
<td>DBNAme,SCHEma</td>
</tr>
<tr>
<td>DMCL</td>
<td>dmcl-name</td>
<td>DMCL</td>
<td></td>
</tr>
<tr>
<td>Database name table</td>
<td>database-table-name</td>
<td>DBTB</td>
<td></td>
</tr>
</tbody>
</table>

⚠️ **Note:** There is no resource type keyword for DBADMIN privilege.

### Naming Examples

This example presents the possible combinations of external resource name fields for a DC task. The actual number of fields that you specify depends on how the resource name is defined in the external security system:

```
environment-name.TASK.system-identifier.task-codeenvironment-name.TASK.task-codeenvironment-name.system-identifier.task-codeenvironment-name.task-codeTASK.environment-name.system-identifier.task-codeTASK.task-codeTASK.system-identifier.task-codeTASK.task-codeSYSTEM.task-code
```
Defining External Signon Security

Contents
- Defining SRTT Entries (see page 40)
  - Example for CA Top Secret (TSS) (see page 40)
  - Example for CA ACF2 (see page 40)
- Optionally Defining PassTickets (see page 41)
  - Example for CA Top Secret (CA TSS) (see page 41)
  - Example for CA ACF2 (see page 42)
  - Example for IBM RACF (see page 42)
- External Signon Security Checking (see page 43)

Defining SRTT Entries

To secure system signon externally, add an entry to the SRTT for the signon (SGON) resource type.

The applicable resource name for the signon resource type is system-identifier; it matches the value in the SYSTEM ID parameter of the system generation SYSTEM statement. Thus, the name of the resource defined in the external system must match the system identifier.

The following examples for CA TSS and CA ACF2 show the relationships that must exist between the system identifier in system generation and the resource identifier in the external security definition; and between the resource class in the external security definition and the external class in the SRTT entry.

Example for CA Top Secret (TSS)

SYSGEN syntax

MOD SYSTEM 120 SYSTEM ID IS IDMSD

CA TSS for z/OS syntax

TSS PERMIT(user-identifier) SGO(IDMSD)

#SECRTT syntax

#SECRTT TYPE=ENTRY, X
   RESTYPE=SGON, X
   EXTCLS='SGO', X
   EXTNAME=(RESNAME)

Example for CA ACF2

SYSGEN syntax

MOD SYSTEM 120 SYSTEM ID IS IDMSD

CA ACF2 for z/OS syntax
Optionally Defining PassTickets

PassTickets can be used as an alternative to a password.

⚠️ **Note:** For more information on PassTickets, see [Signon Processing](https://docops.ca.com/display/IDMS19/Signon+Processing).

To use PassTickets for externally secured signon, add PassTicket definitions to the particular external security system being used. Depending on the external security system in use, these definitions can include:

- Defining a resource class for PassTickets.
- Granting ownership of the resource used for PassTickets.
- Defining a session key for each application for which PassTickets are used.
- Granting permission to a user to the resource used for PassTicket validation.

**Determining Applid**

The applid specified in the definition of the PassTicket to the external security system is a unique identifier for the IDMS CV system. It is composed of the first VTAM line defined to the system. If no VTAM lines exist, it is composed of the system nodename.

See the following sections for examples of the external security definitions needed to allow PassTicket use.

**Example for CA Top Secret (CA TSS)**

This CA TSS example shows the external security definitions needed to allow PassTicket use:

1. Define the resource class PTKTDATA:
   ```
   TSS ADDTO(RDT) RESCLASS(PTKTDATA) ACLIST(ALL,READ,UPDATE) MAXLEN(37)
   ```
2. Add IDMSDEPT department ownership for resources of class PTKTDATA:
   ```
   TSS ADDTO(IDMSDEPT) PTKTDATA(IRRPTAUTH)
   ```
3. Add a session key for each applid (PSTKAPPL):
   ```
   TSS ADDTO(PSTKAPPL) PTKTDATA(IRRPTAUTH)
   ```
4. Add permission for JOHN_SMITH to generate and use a PassTicket for SYSTEM 73:

TSS PERMIT (JOHN_SMITH) PTKTDATA (IRRPTAUTH.IDMSSY73.JOHN_SMITH) ACCESS (READ, UPDATE)

Example for CA ACF2

This CA ACF2 example shows the external security definitions needed to allow PassTicket use:

1. Define the CA IDMS PassTicket session key(s) and assign them to IDMS application IDs (or CV nodenames):

   SET PROFILE (PTKTDATA) DIVISION (SSIGNON)
   INSERT IDMSSY73 SSKEY (0123456789ABCDEF)
   INSERT IDMSSY74 SSKEY (ABCDEF0123456789)
   F ACF2,REBUILD (PTK),CLASS (P)
   END

2. Issue the following commands to assign PassTicket session key(s) for specific user(s) (here: JOHN_SMITH):

   ACFNRULE KEY (IRRPTAUTH) TYPE (PTK) ADD (IDMSSY73.JOHN_SMITH) UID JOHN_SMITH
   (JOHN_SMITH) SERVICE (READ, UPDATE) ALLOW
   F ACF2,REBUILD (PTK)

Example for IBM RACF

This IBM RACF example shows the external security definitions needed to allow PassTicket use:

1. Issue the following commands to activate the PassTicket class:

   SETROPTS CLASSACT (PTKTDATA)
   SETROPTS RACLIST (PTKTDATA)
   SETROPTS GENERIC (PTKTDATA)

2. Issue the following commands to define profile(s) for the IDMS application IDs (or CV nodenames) and specify the session key(s):

   RDEFINE PTKTDATA IDMSSY73 SSIGNON (KEYMASKED (0123456789ABCDEF)) UACC (NONE)
   RDEFINE PTKTDATA IDMSSY74 SSIGNON (KEYMASKED (ABCDEF0123456789)) UACC (NONE)

3. Issue the following commands to define profile(s) and enable UPDATE access to the IDMS PassTicket resource for specific user(s) (here: JOHN_SMITH):

   RDEFINE PTKTDATA IRRPTAUTH.IDMSSY73.JOHN_SMITH UACC (NONE)
   PERMIT IRRPTAUTH.IDMSSY73.JOHN_SMITH CLASS (PTKTDATA) ID (JOHN_SMITH) ACCESS (READ, UPDATE)
   RDEFINE PTKTDATA IRRPTAUTH.IDMSSY74.JOHN_SMITH UACC (NONE)
   PERMIT IRRPTAUTH.IDMSSY74.JOHN_SMITH CLASS (PTKTDATA) ID (JOHN_SMITH) ACCESS (READ, UPDATE)

4. Issue the following command to refresh the PTKTDATA class:

   SETROPTS RACLIST (PTKTDATA) REFRESH
External Signon Security Checking

External signon security checking consists of two phases:

1. Validation of the executing user in the external security system.
2. If the first phase is successful, a check on the user’s authority to access the system identified in the current request.

**Note:** For more information, see Signon Processing (https://docops.ca.com/display/IDMS19/Signon+Processing).

External Database Security Considerations

Before you secure a database externally, weigh the following considerations:

- If you add an SRTT entry that secures the DB resource type externally, you automatically secure a group of database resource types externally.

  **Note:** For more information, see Securing Database Resources (see page 88).

- If the SRTT contains one or more occurrence overrides that specify external security for resource type DB, you must also add an SRTT entry specifying external resource class and external resource name for each of the database resource types that are automatically secured externally.

- External security checks for database resources initiated by SQL bulk processing may have a discernible effect on processing time.

  **Note:** For more information, see SQL Security Enforcement (see page 102).

Identifying Authorities to the External System

**How It is Done**
When a security check is issued for a resource that is secured externally, CA IDMS central security converts the authority that is being checked to a RACROUTE keyword. This table shows the RACROUTE keywords that may be used if security is external and the comparable privileges in CA IDMS internal security.

<table>
<thead>
<tr>
<th>Type of authorities</th>
<th>RACROUTE keyword</th>
<th>CA IDMS privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime authorities</td>
<td>CONTROL</td>
<td>DELETE</td>
</tr>
<tr>
<td></td>
<td>UPDATE</td>
<td>INSERT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UPDATE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBAWRITE</td>
</tr>
<tr>
<td></td>
<td>READ</td>
<td>SELECT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXECUTE(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBAREAD</td>
</tr>
<tr>
<td>Definition authorities</td>
<td>ALTER</td>
<td>REFERENCES</td>
</tr>
<tr>
<td></td>
<td>READ</td>
<td>DISPLAY</td>
</tr>
<tr>
<td></td>
<td>USE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CONTROL</td>
<td>DROP</td>
</tr>
<tr>
<td></td>
<td>UPDATE</td>
<td>ALTER</td>
</tr>
<tr>
<td></td>
<td>CONTROL</td>
<td>CREATE</td>
</tr>
<tr>
<td>Administration authorities</td>
<td>ALTER</td>
<td>SYSADMIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DCADMIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBADMIN</td>
</tr>
</tbody>
</table>

(1) No keyword is passed if EXECUTE is for a resource that can be categorized.

Defining External Performance Monitor Security

The CA IDMS Performance Monitor Real-time Monitor provides the optional functionality to do the following:

- Cancel a Remote Task (see page 45)
- Monitor a Secured External IDMS CV (see page 46)

These functions are performed by authorized users on a local CV, and the functions require the definition of the resource class CA@IDMSI within an external security database—CA Top Secret, CA ACF2, or IBM RACF. After the CA@IDMSI resource class is defined, and before access permissions can be granted to the resource, resource names are defined and assigned an owner.

The following table shows the required format for the resource class, and the access permissions that can be granted to the resource:

<table>
<thead>
<tr>
<th>Task</th>
<th>Resource Class Name</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canceling a remote task</td>
<td>TASK.targetCVname.taskcode</td>
<td>CONTROL access</td>
</tr>
</tbody>
</table>
### Task

<table>
<thead>
<tr>
<th>Task</th>
<th>Resource Class Name</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Securing an external IDMS CV</td>
<td>SYST.targetCVname</td>
<td>READ access</td>
</tr>
</tbody>
</table>

The `targetCVname` is the name of the CV startup job of the remote CV, and `taskcode` is the name of the task code to be canceled on the remote CV.

**Note:** Be careful when assigning resource ownership. Resource ownership typically means the user ID that is assigned ownership has an access level of ALL.

Once the resource name is defined and assigned an owner, permission can be granted to authorize access to the resource by user IDs or groups. Unauthorized user IDs or groups are not allowed to cancel remote tasks or monitor external IDMS CVs. An error message PMRM panel is displayed within the PMRM function panel displaying the “Access Denied” message and a non-zero return code.

### Considerations

When an external IDMS CV is secured by an external security manager, the secured External IDMS CV is included in the list of Active Systems that are displayed in the local PMRM session.

Setting currency to a secured target, IDMS CV, is allowed only in the local PMRM session by an authorized user. Undefined or unauthorized users attempting to set currency to a secured target CV from a local PMRM session receive a “Request denied” error condition.

Message output that is generated for PMRM by the external security product (CA Top Secret, CA ACF2, or IBM RACF) is managed by the SAFSUPP=YES/NO parameter of the #SRTT macro and Optional bits 51 and 226.

### Cancel a Remote Task

Within the CA IDMS Performance Monitor Real-time Monitor, you can cancel an active task executing on a remote Central Version directly from the Active User Task Detail (PF4) screen.

**To enable canceling tasks on remote systems you need to:**

1. Use an external security manager—CA Top Secret (TSS), CA ACF2, or IBM RACF.
2. Grant access to TASK resources for specific users in the external security manager.
3. Ensure that the IDMSINFO address space is active.
4. Set the monitored Central Version to run non-swappable.
Monitor a Secured External IDMS CV

Monitoring an external IDMS CV, by the CA IDMS Performance Monitor Real-time Monitor, is secured using an external security manager—CA Top Secret, CA ACF2, or IBM RACF. The authority to monitor an external IDMS CV can be assigned to user IDs or groups that are defined in an external security database.

To secure monitoring of an external system you need to:

1. Recompile the RHDSRTT module and include the new SECXMON=YES parameter in the RESTYPE=SYST entry of the #SECRTT macro, as shown in the following example:

```
#SECRTT TYPE=ENTRY, X
   RESTYPE=SYST, X
   SECBY=<See Note below>, X
   SECXMON=YES
```

2. The recompiled RHDCSRTT load module must be included in a CDMSLIB loadlib in the startup JCL of the secured target CV.

3. Use an external security manager—CA Top Secret (TSS), CA ACF2, or IBM RACF.

4. Add external security product definitions to authorize monitoring of an external IDMS CV to secured user IDs.
   For example, create a resource class (CDT profile) of CA@IDMSI. Follow the pattern "SYST. targetCVname" for resource names to authorize the monitoring of an external system by a secured user ID.

Secured external monitoring requires permissions to be granted for a resource (the target CV name) by a pre-defined user ID within an IDMS pre-defined resource class (CA@IDMSI). The target CV name can be the JOB name or the started task name (STC) of the target CV startup job.

**Note:** SECBY= Should match your current SECBY= setting for RESTYPE=SYST. If no current setting exists, specify SECBY=OFF.

Examples

The following examples define the CA@IDMSI resource class and assign access privileges to USER1 for job name SYSTEM74, for the specified external security database:

**Example for CA Top Secret**

1. Define the CA@IDMSI resource class.

   ```
   TSS ADDTO(RDT) RESCLASS(CA@IDMSI) RESCODE(120) - 
       ACLST(ALL,READ) DEFACC(READ)
   ```

2. Assign ownership to the resource name.
CA IDMS resource types are grouped as follows:

- Global Resources (see page 47)
- System Resources (see page 49)
- Database Resources (see page 51)

Global Resources

What is a Global Resource?
A global resource is an entity which is shared by all CA IDMS processing in the security domain.

The following table shows the global resource types and the corresponding resource type keywords used in the SRTT and security information databases:

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>USER</td>
</tr>
<tr>
<td>Group</td>
<td>GROU</td>
</tr>
<tr>
<td>User profile</td>
<td>UPRF</td>
</tr>
</tbody>
</table>

**User Catalog**

The definition of a global resource is stored in the user catalog. The user catalog is an area (SYSUSER.DDLSEC) that is shared for retrieval by all CA IDMS processing in the security domain.

**Users**

The user resource type represents the end users, programmers, and administrators who will be accessing systems and databases in the CA IDMS security domain. Users are identified by a user ID that must be unique across the domain.

You maintain the definitions of users in CA IDMS with the CREATE/ALTER/DROP USER statements.

**Groups**

The group resource type represents a collection of users. The following are some important concepts related to groups:

- All users in a group implicitly hold all privileges granted to the group.
- You can assign a user to any number of groups.
- Every user belongs to the group PUBLIC.
- A group cannot be assigned to another group.

You maintain groups with the CREATE/ALTER/DROP GROUP statements.

**User Profiles**

A user profile is a set of attributes that apply to a given user for both online and batch execution in any system in the domain. You create a user profile with the CREATE USER PROFILE statement.

An attribute specifies an environmental default for a user session. An attribute is expressed as a keyword and an associated value for the keyword. For example, SCHEMA=MISTEST is an attribute, of which the keyword is SCHEMA.

Even though the user profile is defined in the CA IDMS user catalog, it is possible for user profile attributes to be invoked whether signon is secured internally or externally.
System Profiles

You can also create a system profile to associate with one or more users in granting signon privilege to a given system. The attributes in a system profile apply to a user session on a specified system. If both a user profile and a system profile are found when the user signs on, the attributes in the two profiles are merged into a user session profile. System profile attributes take precedence over user profile attributes for those attributes defined with an OVERRIDE parameter equal to YES.

System Resources

What is a System Resource

A system resource is an entity shared by all CA IDMS processing under the central version.

The following table shows the system resource types and the corresponding resource type keywords used in the SRTT and security information databases:

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity</td>
<td>ACTI</td>
</tr>
<tr>
<td>Application</td>
<td>SAPP</td>
</tr>
<tr>
<td>Category</td>
<td>CATE</td>
</tr>
<tr>
<td>Signon</td>
<td>SGON</td>
</tr>
<tr>
<td>System</td>
<td>SYST</td>
</tr>
<tr>
<td>System profile</td>
<td>SPRF</td>
</tr>
</tbody>
</table>

System Dictionary

The system dictionary includes all information required to establish, maintain, and control the processing environment. System resources are defined in the DDLDML area of the system dictionary. A system resource is available to all systems generated from the system dictionary.

Purpose of Categories

The category is a mechanism that allows you to group occurrences of several resource types that you have secured internally so that you can grant privilege on the group of resources.
When you create a category, you assign it a name, allowing you to associate a meaningful identifier with the resources. For example, if you secure tasks internally, you might create a category 'SYS_TASKS' and add the DCMT and DCUF tasks to it. If you secure both tasks and programs, one category could contain both task and program resources.

You can define as many as 32,768 categories for your security scheme.

The following table shows resource types that can be categorized and the corresponding resource type keywords used in the SRTT and security information databases:

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>TASK</td>
</tr>
<tr>
<td>Program</td>
<td>SPGM</td>
</tr>
<tr>
<td>Load module</td>
<td>SLOD</td>
</tr>
<tr>
<td>Access module (loadable entity)</td>
<td>SACC</td>
</tr>
<tr>
<td>Run unit</td>
<td>NRU</td>
</tr>
<tr>
<td>Queue</td>
<td>QUEU</td>
</tr>
</tbody>
</table>

Important! If you secure the DB resource, you secure run units and access modules system-wide. You must then categorize load modules in order to grant users execution privilege on them, and you must do the same with access modules unless you choose to grant execution privilege on individual access modules rather than grouping them first.

For more information, see Securing Database Resources (see page 88).

Defining a Category

You add resources to a category with a CREATE or ALTER CATEGORY statement, as in this example:

```
create category dcmt
    add program cdmslib.rhdcmt*;
```

Granting Privilege on the Category

After you define the category, the only means of access to a resource in the category is execution privilege on the category. You give this privilege to a user with a GRANT statement, as illustrated in this example:

```
grant execute
    on category dcmt
    to sam;
```

Runtime Category Selection

At runtime a given resource name may appear to qualify for assignment in more than one category. Consider these two categories:

```
create category dcmt
    add program cdmslib.rhdcmt*;
```
create category dcmtab
  add program cdmslib.rhdcmtab;

When the security system processes as security check, it determines the category of the resource being checked by selecting the mask that is closest to the fully qualified name of the resource. For example, given the preceding two categories, the security system will determine that:

- Use of resource CDMSLIB.RHDCMTXY requires execution privilege on category DCMT.
- Use of resource CDMSLIB.RHDCMTAB requires execution privilege on category DCMTAB.

Database Resources

**What is a Database Resource**

A database resource is an entity associated with the definition of or access to a database.

**Database Resource Types**

The following table shows the database resources type and the corresponding resource type keywords used in the SRTT and security information databases:

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>DB</td>
</tr>
<tr>
<td>Area</td>
<td>AREA</td>
</tr>
<tr>
<td>Run unit</td>
<td>NRU</td>
</tr>
<tr>
<td>SQL schema</td>
<td>QSCH</td>
</tr>
<tr>
<td>Non-SQL defined schema</td>
<td>NSCH</td>
</tr>
<tr>
<td>Table</td>
<td>TABLE</td>
</tr>
<tr>
<td>Access module</td>
<td>DACC</td>
</tr>
<tr>
<td>DBTABLE</td>
<td>DBTB</td>
</tr>
<tr>
<td>DMCL</td>
<td>DMCL</td>
</tr>
</tbody>
</table>

**Securing Database Resources**

If you specify internal security for the database (DB) resource type, you automatically secure the other resource types listed with DB in the preceding table.

You can grant privileges on the individual resource types, but you cannot turn security off in the SRTT for the resource types that are grouped with DB when DB is secured.

**Database Occurrence Overrides**

Using an occurrence override in the SRTT, you can specify a security option for an individual database associated with the system dictionary. For example, in one SRTT entry you can specify no security (the default) for resource type DB and in another entry specify internal security for the production database.

**Ownership**
Ownership is an attribute of an SQL schema. A user who issues a CREATE SCHEMA statement owns the schema that is created.

A schema owner implicitly holds all access and definition privileges on the tables, functions, procedures, table procedures, views, and access modules associated with the schema. The owner also has the authority to grant those privileges to others.

An owner cannot *grant* ownership to another user but can *transfer* ownership. In this way, ownership and its privileges are relinquished to the other user.

The DBMS does not check for ownership. It requests a check for a specific privilege such as SELECT privilege on a table, and the security system returns a positive response if the user in question is the owner of the object.

For more information, see the following topics:
- [CA IDMS Privileges](#) (see page 52)
- [Efficiency Considerations](#) (see page 59)

---

**CA IDMS Privileges**

**Contents**
- Administration Privileges (see page 53)
- Definition Privileges (see page 53)
- Access Privileges (see page 53)
- Granting and Revoking Privileges (see page 54)
- Granting WITH GRANT OPTION (see page 56)
- Specifying Groups (see page 57)
- Using a Wildcard (see page 57)

**How Privileges Work**

If you specify the internal security option for a resource, the resource is secured against access by users who have not been granted privilege on the resource. This security applies to the resource in all systems within the CA IDMS security domain that share the SRTT.

Whoever has authority to grant a privilege has the authority to revoke it. All authority to grant and revoke CA IDMS privileges derives from users who hold SYSADMIN privilege.

**Types of Privilege**

There are three types of CA IDMS privileges:

<table>
<thead>
<tr>
<th>Administration privileges</th>
<th>Definition privileges</th>
<th>Access privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSADMIN</td>
<td></td>
<td>SIGNON</td>
</tr>
<tr>
<td>DCADMIN</td>
<td></td>
<td>EXECUTE</td>
</tr>
<tr>
<td>DBADMIN</td>
<td></td>
<td>SELECT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSERT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UPDATE</td>
</tr>
</tbody>
</table>
Administration Privileges

Administration privileges allow a user to grant and revoke security privileges within a particular scope:

- **SYSADMIN** in effect allows a user to grant all privileges and should be restricted to the security administrator.
- **DCADMIN** allows a user to grant privileges for one or more CA IDMS systems.
- **DBADMIN** allows a user to grant privileges for a database.

Definition Privileges

Definition privileges allow a user to manipulate the definition of certain resources. You can grant definition privileges singly or as a group (the DEFINE privilege).

The following are the individual definition privileges:

- Create
- Alter
- Drop
- Display
- Use
- References (tables)

Access Privileges

Access privileges give users the authority to access specified resources at runtime.

There are three categories of access privileges:

- **Execution** privilege allows a user to execute an access module, activity, or category.
CA IDMS - 19.0

- **Table access** privileges allow a user to perform these operations on data contained in a table:
  - Select
  - Insert
  - Update
  - Delete

- **Special access** privileges refer to the authority to signon to a system or to execute utility functions against an area of the database:
  - SIGNON allows a user to sign on to a specified CA IDMS system.
  - DBAREAD allows a user to run read-only utilities against an area.
  - DBAWRITE allows a user to run utilities that perform read-write functions against an area.

**Granting and Revoking Privileges**

**Absolute Authority of SYSADMIN**

Once you have defined the authorization ID of the security administrator with SYSADMIN privilege and you have secured administration privileges, the privilege to create additional users in the CA IDMS security domain, and the privilege to grant those users privileges, must derive from the security administrator.

**Applicability of Privileges**

CA IDMS privileges are applicable to resources for which the security option is 'INTERNAL'. If the security option for a resource is 'OFF', a user can access the resource without holding a privilege. If the security option for a resource is 'EXTERNAL', the user’s authority to access the resource is determined by the external security system.

Therefore, to use the system of CA IDMS privileges, you must ensure that the runtime security option for the resources to which privileges apply is 'INTERNAL'.

**Granting privilege**

You grant privileges with a GRANT statement. Implicit in each administration privilege is the authority to grant certain privileges:

- SYSADMIN can grant privileges on SYSADMIN, DCADMIN, DBADMIN, and on global resources.
- DCADMIN can grant privileges on DCADMIN and on system resources.
- DBADMIN can grant privileges on DBADMIN and on database resources.

A GRANT statement includes an ON parameter which specifies the resource to which the privileges apply and a TO parameter which specifies the users or groups to whom you are giving the privileges.
Note: For more information on GRANT statement syntax, see the following sections:

- Syntax for Securing Global Resources (see page 183)
- Syntax for Securing System Resources (see page 211)
- Syntax for Securing Database Resources (see page 238)

Duration of Privileges

A user holds privileges explicitly granted to the user until one of these actions occurs:

- The privileges are explicitly taken away by means of the REVOKE statement.
- The user is physically deleted from the user catalog.

A user implicitly holds privileges granted to a group to which the user belongs until one of these actions occurs:

- The user is dropped from the group.
- The privileges are revoked from the group.
- The group is dropped.

Revoking Privileges

Privileges are taken away with the REVOKE statement. A user who has the authority to grant a privilege also has the authority to revoke it.

A REVOKE statement includes an ON parameter which specifies the resource to which the privileges apply and a FROM parameter which specifies the users or groups from whom you are revoking the privileges.

Note: For more information on REVOKE statement syntax, see the following sections:

- Syntax for Securing Global Resources (see page 183)
- Syntax for Securing System Resources (see page 211)
- Syntax for Securing Database Resources (see page 238)

GRANT and REVOKE Example

The first statement gives a table access privilege to user PSD, and the second statement revokes the privilege:
Granting WITH GRANT OPTION

**Grantee Privilege**

When you grant a definition or access privilege to a user, you can also give the user the authority to grant the same privilege to another user; in effect, to pass on the privilege. This authority is called the grantable privilege.

To give a grantable privilege to a user, you specify WITH GRANT OPTION at the end of the GRANT statement.

Giving grantable privileges is an essential technique in decentralizing security administration.

**Grantee Privilege Example**

In this example, the GRANT statement gives user PSD SELECT privilege on the demoempl.employee table, as well as the authority to assign that privilege to other users:

```sql
grant select
  on table demoempl.employee
  to psd
  with grant option;
```

User PSD can now use the GRANT statement to issue the SELECT privilege on the demoempl.employee table to other users.

**Restrictions on Grantable Privilege**

Not all privileges can be grantable privileges. These privileges *cannot* be grantable:

- All administration privileges
- Signon privilege
- Execution privileges on activities and categories

A user holding a grantable privilege does not necessarily have the authority to grant the privilege WITH GRANT OPTION.

⚠️ **Note:** For more information on restrictions on passing grantable privilege, see the discussion of the WITH GRANT OPTION parameter under the applicable GRANT statements in the following sections:
Omitting WITH GRANT OPTION

If you omit WITH GRANT OPTION when you grant a definition or access privilege, the named users receive the definition or access privilege, but it is not grantable. Therefore, the users cannot give the privilege to other users.

Grantable Privilege with REVOKE Statements

Unless you hold an administration privilege, you can revoke a privilege only if you hold the same grantable privilege. For example, a user cannot revoke CREATE privilege on SYSTEM88 unless the user holds grantable CREATE privilege on SYSTEM88.

Specifying Groups

Granting Privileges to a Group

You can grant privileges to a group as well as to individual users. All users in the group hold privileges that you give to the group. Users you add to a group hold all privileges assigned to the group; users you remove from the group lose all group privileges.

Revoking Group Privileges

You cannot revoke a privilege from an individual user if the user belongs to a group that holds the privilege. Rather, you must take one of these steps:

- Drop the user from the group using the ALTER GROUP statement. This action removes all privileges the user held as a result of being in the group.

- Remove the privilege from the group. This action removes the privilege from all members of the group who hold the privilege as a result of being in the group.

Using a Wildcard

What Wildcarding Is

Wildcarding is the use of a single character to represent one or more characters omitted from a string. An entity name with a wildcard character identifies all the entities whose names match the pattern established by the wildcarded name.

Why You Use Wildcards

In most cases, you can use a wildcard when naming the resources to which the privileges in a GRANT statement apply. This allows you to do the following:
- Enforce high-level naming conventions
- Grant privileges on groups of resources

**Document convention:** If a parameter value in a security statement can include a wildcard, the parameter description that follows the statement syntax diagram explicitly notes your ability to use a wildcard.

**How to Wildcard:** The wildcard character is the asterisk (*). You can use the wildcard only as the last character in a resource name. For example, * and A* and ABCD* are valid, but *A and A*BC are not.

**Wildcarding Qualified Resource Names**

- APPLDICT.HR*
- APPLDICT.*
- APPLDICT*
- APPL*

Specific restrictions on wildcarding are described appropriately in the syntax parameter descriptions found in the statement syntax sections later in this section.

⚠️ **Note:** Special considerations apply to the effect of using a wildcard in reference to categories generally and in CREATE RESOURCE statement particularly. For more information, see the [Usage (see page 215)](#) section of CREATE RESOURCE in the section Syntax for Securing System Resources.

**Granting and Revoking with a Wildcard**

When you grant a privilege on resources using a wildcard, you must use the same wildcard to revoke the privileges.

For example, if you grant CREATE privilege on category HR* to user ABC, you must issue a REVOKE CREATE or REVOKE DEFINE statement on category HR* to revoke the privilege from user ABC. Revoking privilege on category *, category H*, category HR, or category HRA has no effect on privileges granted on category HR*.

**Considerations in Revoking Privileges**

Through the use of groups and wildcards in a GRANT statement, a user can be given the same privilege on a resource more than once. A REVOKE statement revokes the privileges specified in the statement only on the specified resource name and only from the specified user or group. Thus, it is possible for a user to retain a privilege even after it has been revoked.

For example, suppose:

- User PKB is in the group SALES_ADMIN.
PKB has been granted the CREATE privilege on the access module name SALES_SCH. SALES_FORECAST.

SALES_ADMIN has been granted the CREATE privilege on all access modules named SALES_SCH. SALES* where * is a wildcard character.

You can revoke the CREATE privilege on SALES_FORECAST from the user identifier PKB. However, PKB can still create an access module by that name in the SALES_SCH schema because PKB is a member of SALES_ADMIN.

Efficiency Considerations

In security administration, you can perform your task more efficiently by making consistent use of groups, wildcards, and categories. This strategy will also produce runtime efficiency.

Using Groups

Your security strategy should isolate user roles that require similar types of privileges. You can then establish groups for each user role. This allows you to grant and revoke privileges at the group level, thus reducing the number of statements needed to administer the security scheme.

![Note: Groups also enhance efficiency by improving runtime performance.](image)

Comparison of Groups and No Groups

- **Without groups**
  Without groups, you must list each user ID for each GRANT statement:

  ```
  grant access on table qa.employee to psd, rkn, jfd, wxe, lsb;
  grant access on table qa.job to psd, rkn, jfd, wxe, lsb;
  grant access on table qa.benefits to psd, rkn, jfd, wxe, lsb;
  grant access on table qa.department to psd, rkn, jfd, wxe, lsb;
  grant execute on access module qa.empdbmod to psd, rkn, jfd, wxe, lsb;
  grant execute on access module qa.empdbret to psd, rkn, jfd, wxe, lsb;
  ```

  Now to revoke privileges for one of the users, you must code a REVOKE statement for each GRANT statement:

  ```
  revoke access on table qa.employee from rkn;
  revoke access on table qa.job from rkn;
  revoke access on table qa.benefits from rkn;
  revoke access on table qa.department from rkn;
  ```

- **With groups**
  After you create the group, you list only the group name on each GRANT statement:

  ```
  create group qagroup
  add user psd, rkn, jfd, wxe, lsb;
  ```
grant access on table qa.employee to qagroup;
grant access on table qa.job to qagroup;
grant access on table qa.benefits to qagroup;
grant access on table qa.department to qagroup;
grant execute on access module qa.empdbmod to qagroup;
grant execute on access module qa.empdbret to qagroup;

To revoke privileges from one of the users, you simply drop the user from the group:

alter group qagroup
  drop user rkn;

Using Wildcards

Your strategy should isolate resources that require similar types of security. You can then grant privileges on them using a wildcard. This allows you to implement your strategy at a higher level, thus eliminating the need to issue a GRANT statement for individual resources.

Wildcard Examples

Without wildcards, you must issue a separate statement for each table when you grant table access privileges:

- Without wildcards
  grant access on table qa.employee to qagroup;
grant access on table qa.job to qagroup;
grant access on table qa.benefits to qagroup;
grant access on table qa.department to qagroup;
grant execute on access module qa.empdbmod to qagroup;
grant execute on access module qa.empdbret to qagroup;

- With wildcards
  You can use a wildcard to grant table access privileges on all tables in the qa schema:
  grant access on table qa.* to qagroup;
grant execute on access module qa.* to qagroup;

Using categories

CA IDMS provides the category mechanism to help you to manage privileges on runtime resources efficiently.

Summary Example

This series of statements uses groups, wildcards, and categories to secure the resources available for two levels of use, as described in the definition of groups hrdisp and hrupd:

create group hrdisp
description 'HR users who can display Employee'
  add user lsd, lhn, pxw, gsr, hxm, fbs;
create group hrupd
description 'HR users who can update Employee'
  add user gsr, hxm, fbs;
create resource category benefits display
  add access module appldict.prod.bendis
  add load module appldict.v0001.benefits
  add program cdmslib.bendisp
add task bendisp;

create resource category benefits update
   add access module appldict.prod.benupd
   add program cdmsslib.benupd
add task benupd;

grant execute on category benefits_display to hrdisp;
grant execute on category benefits_* to hrupd;

Securing Global Resources

CA IDMS Security Domain

The CA IDMS security domain is the set of DC systems and local mode applications sharing a single user catalog and SRTT.

Global Resources

The scope of global resources is domain-wide. These resources are:

- Users
- Groups
- User profiles
- SYSADMIN privilege

User catalog

The user catalog is the CA IDMS repository that contains:

- The definition of all authorization IDs (users and groups) within the domain
- The specification of authorization IDs holding SYSADMIN privilege for the domain
- The definition of user profiles
- The privileges held by users and groups on global resources

For more information, see the following topics:

- The User catalog (see page 62)
- Securing SYSADMIN Privilege (see page 63)
- Securing Users (see page 64)
- Securing Groups (see page 66)
- Securing User Profiles (see page 67)
The User catalog

Defining a CA IDMS Security Domain

You include multiple CA IDMS systems in a security domain by specifying an identical set of physical characteristics for the SYSUSER.DDLSEC segment in each system in the domain, and specifying the same physical data set in the startup JCL or the global DMCL.

Note: For more information on defining physical database characteristics, see the CA IDMS Database Administering section.

Use of the User Catalog

The user catalog is accessed by all DC/UCF systems and local mode batch applications executing in the security domain. It is a central location used for validating passwords and retrieving user information.

Only a DC/UCF system that has the user catalog in update mode can be used to define and administer global resources.

Securing the User Catalog

After you have specified a resource option (other than 'OFF') for the DB resource type, the user catalog is secured.

You can grant access to the user catalog in one or more of these ways:

- Granting DBAREAD/DBAWRITE privileges on area SYSUSER.DDLSEC
- Granting DBADMIN privilege on DB SYSUSER
- Granting USE privilege on non-SQL defined schema IDMSSECU and granting definition privilege on an SQL schema for IDMSSECU

Note: For more information, see Securing the Dictionaries and the User catalog (see page 118).

Ensuring Use of the Correct User Catalog

You can ensure that only the correct user catalog is accessed at runtime.

If the operating system or spooler supports installation-written exits for scanning and validating JCL, a system programmer can write an exit to verify that the correct system dictionary and user catalog are used by each central version and local mode job.
Alternatively, in an operating system that supports dynamic file allocation, you can specify the data set name of the user catalog in the DSNAME parameter of the CREATE FILE statement and NULL for the external file name in the ASSIGN TO parameter. At runtime the data set name is obtained from the DMCL, which contains the segment associated with the file.

Securing SYSADMIN Privilege

About SYSADMIN Privilege

SYSADMIN privilege authorizes the holder to grant and revoke privileges on any resource within the domain. It also enables the holder to define resources and to delegate administration privileges.

In sum, the holder of SYSADMIN privilege can administer the security system.

Until you secure the SYSADMIN resource, any user can administer SYSADMIN privilege.

How to Secure SYSADMIN

To secure SYSADMIN internally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
  RESTYPE=SYSA, X
  SECBY=INTERNAL
```

To secure SYSADMIN externally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
  RESTYPE=SYSA, X
  SECBY=EXTERNAL, X
  Additional parameters required
```

Note: For more information, see #SECRTT (see page 160).

Restricting SYSADMIN

Since SYSADMIN is the master security definition privilege, it is very important to restrict the granting of SYSADMIN authority.

Consider assigning SYSADMIN to a group rather than an individual user so that security can be administered in a timely fashion should the primary administrator be unavailable.

Decentralizing Administration

The holder of SYSADMIN can decentralize security administration by granting to appropriate users:

- DCADMIN privilege, which allows a user to define system resources and grant access to those resources
- DBADMIN privilege on a database, which allows a user to define database resources and grant access to those resources
Definition privilege on global resources

You should carefully restrict grants of administration privileges. A user with administrative privilege can grant and revoke privileges on all resources within the scope of the administration privilege.

**Granting Administration Privileges**

You can give SYSADMIN, DCADMIN, and DBADMIN privileges to one or more users with a grant statement, as in this example of a statement that grants DBADMIN privilege on a specified database:

```sql
grant dbadmin
on db testdb
to devdba;
```

**More Information**

For more information on granting administration privileges, see the following sections:

- **GRANT Administration Privilege (see page 201)** in the section "Syntax for Securing Global Resources"
- **GRANT Administration Privilege (see page 201)** in the section "Syntax for Securing System Resources"
- **GRANT Administration Privilege (see page 201)** in the section "Syntax for Securing Database Resources"

### Securing Users

**About Users**

Defining users in the CA IDMS user catalog is essential if the security option for one or more resources is internal, even if signon processing is controlled externally. A security check on an internally secured resource fails if the executing user is not defined in the user catalog.

Until you secure the user resource, any user can define users in the user catalog.

**How to Secure Users**

To secure users internally, include an entry in the SRTT:

```text
#SECRTT    TYPE=ENTRY,
           RESTYPE=USER,
           SECBY=INTERNAL
```

To secure users externally, include an entry in the SRTT:

```text
#SECRTT    TYPE=ENTRY,
           RESTYPE=USER,
           SECBY=EXTERNAL,
           Additional parameters required
```
How to Define Users

You define a user with a CREATE USER statement. For example, this statement creates user RKN:

```
create user rkn
group mis
name 'Randall K. Nelken'
password ranken
profile misprof;
```

Note: For more information, see CREATE USER (see page 191).

Maintaining User Definitions

You can alter the definition of a user with an ALTER USER statement. You can drop the definition of a user with a DROP USER statement.

Note: For more information, see the following sections:

- ALTER USER (see page 185)
- DROP USER (see page 198)

Granting Definition Privileges on Users

You can delegate the authority to define and maintain users by granting definition privileges on users. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can specify all definition privileges (DEFINE). You can specify WITH GRANT OPTION when you grant these privileges to allow the user to grant the same privileges to another user.

In this example, user mis1 is given the privilege to create or alter the definition of users whose user IDs begin with 'mis':

```
grant alter, create
on user mis*
to mis1;
```

Note: For more information, see GRANT Definition Privileges (see page 203).
Securing Groups

About Groups

You define groups for administrative efficiency. You group users according to the privileges that they require. Then you grant the privileges to the group rather than to individual users.

If you create a group of 10 users, you can grant each user the same five privileges by issuing five GRANT statements to the group. You would issue 50 statements to accomplish the same task if you did not first create the group.

Until you secure the group resource, any user can maintain definitions of groups in the user catalog.

How to Secure Groups

To secure the groups internally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,
RESTYPE=GROU,               X
SECBY=INTERNAL             X
```

To secure the groups externally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,
RESTYPE=GROU,               X
SECBY=EXTERNAL,             X
```

Additional parameters required

Note: For more information, see #SECRTT (see page 160).

How to Define Groups

You define a group by specifying the users that comprise the group with the CREATE GROUP statement.

```
create group mis_group
  description 'Management Information Services'
  add user mis1, mis2, mis3, mis4, mis5;
```

Note: A group cannot be a member of another group.

For example, this statement creates mis_group:

```
create group mis_group
  description 'Management Information Services'
  add user mis1, mis2, mis3, mis4, mis5;
```

Note: For more information on defining and maintaining group definitions, see the following sections:
Granting Privileges to a Group

When you grant privileges to a group, each member of the group is implicitly granted the specified
privileges.

For example, this statement grants mis_group the privilege of retrieving data from SYSTEM tables:

```
grant select
    on table system.*
    to mis_group;
```

A member of a group can hold additional privileges as an individual user or as a member of a
different group. For example, mis1 might hold an administrative privilege that other members of
mis_group do not hold.

Dropping Users and Groups

When a user is dropped from a group, all privileges inherited from the group are implicitly revoked
from the user.

When a group is dropped, all privileges granted to that group are automatically revoked.

Granting Definition Privileges on Groups

You can delegate the authority to define and maintain groups by granting definition privileges on
groups. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can
specify all definition privileges (DEFINE). You can specify WITH GRANT OPTION when you grant these
privileges to allow the user to grant the same privileges to another user.

⚠️ Note: For more information, see GRANT Definition Privileges (see page 203).

Securing User Profiles

About User Profiles

You define a user profile to specify attributes for a user session in the domain (that is, irrespective of
the system to which the user is signed on) whether the execution mode is online or batch.

Until you secure the user profile resource, any user can maintain user profile definitions in the user
catalog.
How to Secure User Profiles

To secure user profiles internally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,
       RESTYPE=UPRF,
          SECBY=INTERNAL
```

To secure the user profiles externally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,
       RESTYPE=UPRF,
          SECBY=EXTERNAL,
```

Additional parameters required

Note: For more information, see #SECRTT (see page 160).

Attributes

An attribute is the combination of a keyword and a value associated with the keyword. A user profile can contain multiple attributes.

Attributes are used by CA IDMS software to control the user session. Attributes can also be retrieved by application programs for additional application security and other purposes.

When you specify an attribute in a user profile definition, you have the option of marking it to indicate that the user is not permitted to override the attribute value at runtime with a DCUF SET PROFILE statement or, for attribute keywords with meaning to CA IDMS, with a SYSIDMS parameter.

Note: For more information on DCUF SET PROFILE, see the CA IDMS System Tasks and Operator Reference section.

For more information on SYSIDMS parameters, see the Using section.

How to Define User Profiles

You define a user profile with the CREATE USER PROFILE statement.

In this example, the first statement creates a user profile called MISPROF and the second statement associates the profile with user RKN:

```
create user profile misprof
attributes
       dept='0056' override no,
       jobcode='42' override no,
       schema=&user'.,
       prtdest='gdnc005';

alter user rkn
    profile misprof;
```
Note: For more information on creating and maintaining user profile definitions, see the following sections:

- CREATE USER PROFILE (see page 194)
- ALTER USER PROFILE (see page 187)
- DROP USER PROFILE (see page 200)

Granting Definition Privileges on User Profiles

You can delegate the authority to define and maintain user profiles by granting definition privileges on user profiles. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can specify all definition privileges (DEFINE). You can specify WITH GRANT OPTION when you grant these privileges to allow the user to grant the same privileges to another user.

Note: For more information, see GRANT Definition Privileges (see page 203).

Associating User Profiles with Users

You can associate a user profile with a user in one of the following ways:

- Explicitly in the PROFILE parameter of a CREATE USER or ALTER USER statement.

Note: For more information, see the following sections:

- Implicitly by assigning the user profile a name matching the user ID. This user profile is located at signon if no user profile has been specified in the user definition.

User Attributes in a System Profile

A system profile allows you to set the attributes of a user’s session for a specific system. The system profile associated with the user is determined in one of the following ways:

- By the specification, if any, made in the GRANT SIGNON statement for the user.
- By the specification of default system profile made on the initial #SECRTT macro.
- If not specified on the #SECRTT macro, the system profile DEFAULT, if it exists.

Even if there is no system profile specification in GRANT SIGNON or on the #SECRTT, you can tailor a system profile to a user or the user’s default group by specifying INCLUDE='&USER'. or INCLUDE='&GROUP'. in a system profile named 'DEFAULT.' because the system will search for the system profile DEFAULT at signon time.
If you have created a system profile with a name that matches the ID of the signed-on user and system profile DEFAULT contains INCLUDE='&USER'., the attributes of the nested system profile with a name matching &USER. are set for the session profile.

If you have created a system profile with a name that matches the name of the signed-on user's default group and system profile DEFAULT contains INCLUDE='&GROUP'., the attributes of the nested system profile with a name matching &GROUP. are set for session profile.

**Scope of Profiles**

The scope of system profile DEFAULT is the set of systems that share the SYSTEM.DDLDML area in which DEFAULT is defined. The scope of a user profile is the CA IDMS domain, which is the set of systems that share the SYSUSER.DDLSEC area. System profile attributes take precedence over matching user profile attributes unless the user profile attribute is defined with the OVERRIDE NO parameter.

**Note:** For more information on system profiles, see the *CA IDMS System Tasks and Operator Reference* section.

---

**Securing System Resources**

**Securing Definitions In The System Dictionary**

Security information on system resources (and some database resources) is stored in the system dictionary. To secure the security information itself, secure the system dictionary.

**How You Do It**

Since the system dictionary is a CA IDMS database, you secure the system dictionary by activating security for resource type DB. You can specify an SRTT entry to secure all databases in the system, or you can specify an individual entry for each database you choose to secure.

**Important!** Before securing any database in the system, become familiar with all considerations related to database security, as described in [Securing Database Resources](http://wiki-dev.ca.com/display/IDMS/Securing+Database+Resources).

For more information, see the following topics:

- Securing DCADMIN (see page 71)
- Securing Systems (see page 71)
- Securing Signon (see page 73)
- Securing System Profiles (see page 74)
Securing DCADMIN

About the DCADMIN Privilege

A holder of DCADMIN privilege can perform system administration functions. The DCADMIN user holds definition and access privileges on /DC system resources. CREATE, ALTER, DROP, and DISPLAY privileges on a system allow the user to maintain the system configuration using CA IDMS system generation.

The holder of DCADMIN privilege can grant all system privileges to one or more users.

Until you secure the DCADMIN resource, any user can administer security on system resources.

How to Secure DCADMIN

To secure DCADMIN internally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,                    X
           RESTYPE=DCA,                        X
           SECBY=INTERNAL
```

To secure DCADMIN externally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,                    X
           RESTYPE=DCA,                        X
           SECBY=EXTERNAL,                     X

Additional parameters required
```

Note: For more information, see #SECRTT (see page 160).

How to Grant the DCADMIN Privilege

You can grant DCADMIN privilege to one or more users with a GRANT DCADMIN statement. To issue this statement, you must hold either SYSADMIN privilege or DCADMIN privilege.

Note: For more information, see GRANT Administration Privilege (see page 223) in the section Syntax for Securing System Resources.

Securing Systems

About Systems
The system resource represents DC systems in the domain.

Until you secure systems, any user can create and maintain a system using CA IDMS system generation.

**How to Secure Systems**

To secure systems internally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
RESTYPE=SYST, X
SECBY=INTERNAL
```

To secure systems externally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
RESTYPE=SYST, X
SECBY=EXTERNAL, X
Additional parameters required
```

⚠️ **Note:** For more information, see #SECRTT (see page 160).

**How to Define a System**

You define a system as a secured resource with a CREATE RESOURCE SYSTEM statement, specifying the identifier of the system to match the ID parameter of the SYSTEM statement in system generation.

⚠️ **Note:** For more information on defining and maintaining system resources, see the following sections in the section Syntax for Securing System Resources:

- CREATE RESOURCE (see page 217)
- DROP RESOURCE (see page 221)

**How to Grant Definition Privileges on Systems**

You can delegate the authority to define and maintain systems by granting definition privileges on systems. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can specify all definition privileges (DEFINE). As a holder of DCADMIN privilege, you can specify WITH GRANT OPTION when you grant definition privileges to allow the recipient to grant the same privileges to another user.

⚠️ **Note:** For more information on administering privileges on systems, see the following sections in the section Syntax for Securing System Resources:
Securing Signon

About Signon

The signon resource controls access to DC systems, whether the signon is explicit or implicit. The security option you specify for signon determines whether password validation is enforced by the external system, internally, or not at all.

⚠️ **Note:** For more information on signon processing, see [Signon Processing](https://docops.ca.com/display/IDMS19/Signon+Processing).

⚠️ Until you secure the signon resource, any user can sign on to any system in the domain.

How to Secure Signon

To secure signon internally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,            X
           RESTYPE=SGON,            X
           SECBY=INTERNAL
```

To secure signon externally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,            X
           RESTYPE=SGON,            X
           SECBY=EXTERNAL,          X
     Additional parameters required
```

⚠️ **Note:** For more information, see [#SECRTT](#) (see page 160).

Granting System Signon Privilege

To control access to a system, you grant signon privilege to a user. You cannot grant signon to a group.

⚠️ **Note:** For more information, see the following sections in the section Syntax for Securing System Resources:
CREATE RESOURCE (see page 217)

DROP RESOURCE (see page 221)

When you grant signon, you can associate a system profile with the user.

Note: For more information on system profiles, see the CA IDMS System Tasks and Operator Reference section.

Securing System Profiles

About System Profiles

The system profile resource represents a profile that is defined for a given system and can be associated with one or more users.

Although system profile is a resource that you can protect with CA IDMS centralized security, system profiles are not considered part of the security architecture because the scope of their influence on user session attributes is system-wide, not domain-wide.

Until you secure system profiles, any user can create and maintain a system profile.

How to Secure System Profiles

To secure system profiles internally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
  RESTYPE=SPRF, X
  SECBY=INTERNAL
```

To secure system profiles externally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
  RESTYPE=SPRF, X
  SECBY=EXTERNAL, X
  Additional parameters required
```

Note: For more information, see #SECRTT (see page 160).

Defining a System Profile

You define a system profile with a CREATE SYSTEM PROFILE statement, specifying the profile name and profile attributes.
Granting Definition Privileges on System Profiles

You grant definition privileges on systems profiles with a GRANT statement. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can specify all definition privileges (DEFINE). As a holder of DCADMIN privilege, you can specify WITH GRANT OPTION when you grant definition privileges to allow the recipient to grant the same privileges to another user.

Note: For more information on administering privileges on system profiles, see the following sections in the section Syntax for Securing System Resources:

- GRANT System Definition Privileges (see page 229)
- REVOKE System Definition Privileges (see page 236)

Securing Resources That Can Be Categorized

If you secure certain system resources internally, you must group occurrences of these resources in categories and grant execution privilege on the categories to allow access.

You create categories using CREATE RESOURCE CATEGORY statements. You authorize access with GRANT EXECUTE ON CATEGORY statements.

Categories are not meaningful to external security enforcement. However, you can choose to specify external security for any resource type that can be categorized.

If you specify external security for a resource type that can be categorized, you must also specify in the external security system rules for all occurrences of the resource type.

- Resource Types That Can Be Categorized (see page 75)
- Securing Programs (see page 76)
- Securing Load Modules (see page 78)
- Securing Queues (see page 79)
- Securing Tasks (see page 80)
- Category Bit Assignments (see page 81)

Resource Types That Can Be Categorized

The following table shows resource types that can be categorized and the keywords that you specify in the SRTT to secure them.
Note: Run units and access modules are secured internally by specifying resource type 'DB'.

<table>
<thead>
<tr>
<th>Resource</th>
<th>SRTT</th>
<th>keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal security</td>
<td>TASK</td>
<td>TASK</td>
</tr>
<tr>
<td>External security</td>
<td>SLOD</td>
<td>SLOD</td>
</tr>
<tr>
<td>Task</td>
<td>SPGM</td>
<td>SPGM</td>
</tr>
<tr>
<td>Load module</td>
<td>SACC</td>
<td>SACC</td>
</tr>
<tr>
<td>Access module(1)</td>
<td>NRU</td>
<td>NRU</td>
</tr>
<tr>
<td>Program</td>
<td>QUEU</td>
<td>QUEU</td>
</tr>
</tbody>
</table>

(1) For more information on securing the DB resource type, see Securing Database Resources (see page 88).

Wildcards

To simplify the process of category management, you can use wildcards when you specify the resource occurrences to add to a category.

In the following example, load modules, tasks, and queues associated with an account receivable application are added to a category:

```
create resource category ar
    add load module appldict.v0001.car*
    add task car
    add queue car*, ap* ;
```

You can also wildcard the category name when you grant privilege. For example, if you create several categories for the accounts receivable application and assign names that begin "AR," you can grant privilege on all of the accounts receivable categories in this way, as shown in the following example:

```
grant execute on category ar*
    to ar_sys_admin;
```

To implement security using categories, follow these steps:

1. Create categories for groups of resource occurrences.
2. Grant users execution privilege on categories.
3. Activate internal security for each categorized resource type.

Securing Programs

When you secure programs, you can control who can execute programs maintained in an operating system load library. Until you secure programs, any user can execute a program in the operating system load library.
To secure programs internally, include an entry in the SRTT, as shown in the following example:

```
#SECRTT    TYPE=ENTRY,           X
    RESTYPE=SPGM,           X
    SECBY=INTERNAL
```

To secure programs externally, include an entry in the SRTT, as shown in the following example:

```
#SECRTT    TYPE=ENTRY,           X
    RESTYPE=SPGM,           X
    SECBY=EXTERNAL,         X
```

Additional parameters required

If you secure programs, only an authorized user can execute a user-mode program, including any CA IDMS user-mode program. You should carefully weigh the requirements for administering program security. For example, if you secure programs externally, you must identify, to the external system, all user-mode programs supplied by CA IDMS, and site-specific application programs that users need to execute and specify the rules for securing these programs.

If you secure programs internally, you can take advantage of categories, wildcards, and groups to simplify this process.

To identify CA IDMS user-mode programs, view the following members in your installation source library:

- SECASF
- SECCULP
- SEICCMS
- SECIDD
- SECITOOL
- SECOCF
- SECOLQ
- SECTOOLS

**Alternative to Program Security**

An approach to securing programs, which may be easier to administer, is to secure databases or database occurrences, as follows:

- For program access to non-network defined databases, you categorize run units and grant execution privilege on the categories. Categorizing run units is simplified by including database, subschema, and program names in the run unit identifier.

- For run unit information on CA IDMS user-mode programs, view the following members in your installation source library:
  - SECASF
For program access to SQL-defined databases, you grant applicable privileges on access modules.

**Program Occurrence Overrides**

You can specify occurrence overrides in the SRTT for the SPGM resource type. If you secure programs externally, you *must* add to the SRTT an occurrence override to unsecure the signon program (RHDCSNON). Without this override, any attempt to signon fails.

In the following example, security for programs is external, but occurrence overrides makes RHDCSNON and RHDCBYE unsecured:

```
#SECRTT TYPE=ENTRY, X
  RESTYPE=SPGM, X
  SECBY=EXTERNAL, X
  Additional parameters required
#SECRTT TYPE=OCCURRENCE, X
  RESTYPE=SPGM, X
  RESNAME='RHDCSNON', X
  SECBY=OFF
#SECRTT TYPE=OCCURRENCE, X
  RESTYPE=SPGM, X
  RESNAME='RHDCBYE', X
  SECBY=OFF
```

More

- #SECRTT (see page 160)
- CREATE RESOURCE (see page 217)
- Category Security Processing (see page )

**Securing Load Modules**

When you secure load modules, you control who can execute load modules that reside in the DDLDCLOD area of the dictionary. Until you secure load modules, any user can execute a load module.
To secure load modules internally, include an entry in the SRTT, as follows:

```
#SECRTT    TYPE=ENTRY,    
RESTYPE=SLOD,    
SECBY=INTERNAL  
```

To secure load modules externally, include an entry in the SRTT, as follows:

```
#SECRTT    TYPE=ENTRY,    
RESTYPE=SLOD,    
SECBY=EXTERNAL,  
```

Additional parameters required

**More**

- #SECRTT (see page 160), in the section "Syntax for Assembler Macros"
- CREATE RESOURCE (see page 217), in the section "Syntax for Securing System Resources"
- Category Security Processing (see page , in the section "Securing System Resources"

**Securing Queues**

When you secure queues, you control who can access a queue. Until you secure queues, any user can access queues.

To secure queues internally, include an entry in the SRTT, as follows:

```
#SECRTT    TYPE=ENTRY,    
RESTYPE=QUEU,    
SECBY=INTERNAL  
```

To secure queues externally, include an entry in the SRTT, as follows:

```
#SECRTT    TYPE=ENTRY,    
RESTYPE=QUEU,    
SECBY=EXTERNAL,  
```

Additional parameters required

**Queue Ownership**

For runtime efficiency, queues are protected by ownership and categories. The user who creates the queue owns the queue.

A queue can be shared if it is assigned to a category. Users with execution privilege on the category can share the queue. An unshared queue should not be assigned to a category. Ownership is used to protect unshared queues.

**How Queue Security Works**
When a user attempts to create a queue at runtime, the queue manager calls the security system to determine if the queue is assigned to a category.

If the queue is in a category:

- The queue is created if the user has execution privilege on the category, and the user ID is recorded as the owner of the queue.

- For subsequent access to the queue, the system compares the requester’s user ID to the queue owner’s user ID and does the following:
  - Grants access if the IDs match.
  - Calls the security system, which attempts to verify that the requester user ID has execution privilege on the queue.

If the queue is not in a category:

- The queue is created and the user ID is recorded as the owner of the queue.

- For subsequent access to the queue, the system compares the requester’s user ID to the queue owner’s user ID and allows access only if they match.

The ownership mechanism allows security for unshared queues to be managed efficiently regardless of whether security for categorized queues is handled by CA IDMS or an external security facility.

Securing Tasks

When you secure tasks, you control who can invoke a task. Until you secure tasks, any user can invoke any task.

To secure tasks internally, include an entry in the SRTT, as follows:

```
#SECRRT TYPE=ENTRY, X
  RESTYPE=_TASK, X
  SECBY=INTERNAL
```

To secure tasks externally, include an entry in the SRTT, as follows:

```
#SECRRT TYPE=ENTRY, X
  RESTYPE=TASK, X
  SECBY=EXTERNAL, X
```
Additional parameters required

Task Occurrence Overrides

In the SRTT you can specify occurrence overrides for the task resource. If you secure tasks externally, you must specify an occurrence override to unsecure the SIGNON task. Without this override, any attempt to sign on fails.

Unsecured Tasks
After task security is activated, you can allow a user to execute certain tasks, whether or not the user is signed on, by creating a category of those tasks and granting execution privilege on the category to group PUBLIC.

If you create a category of tasks that all users can execute if they are signed on, grant the privilege to a group of all users that you explicitly create, not to group PUBLIC.

For external run units such as local utility jobs and access through client/server technology, you must grant execute authority on tasks RHDCNP3S and RHDCNP3J to group PUBLIC or all groups. Alternatively, you can turn off security for tasks RHDCNP3S and RHDCNP3J by including an entry in the SRTT.

**Tasks Started by the System**

If signon and tasks are secured, the identifier of the user who submitted the job to start the system must be authorized to execute tasks that are created directly or indirectly by startup and shutdown autotasks. An autotask is defined in the AUTOTASK statement of system generation.

No security checking is performed for an autotask. However, any task invoked as a result of autotask execution causes a security check for the task.

If signon and tasks are secured internally, perform the following steps to ensure that the tasks can be executed:

1. If not yet done, define the user who submits the job to start the system.
2. If not yet done, grant signon privilege to the user.
3. Define a category and add the tasks invoked by autotasks.
4. Grant execution privilege on the category to the user.

**Category Bit Assignments**

When you create a category, the system represents the category internally by a half-word number. The system maintains category bit assignments to ensure that category numbers are assigned in ascending sequence. The number corresponds to a bit position in a Category bit map.

The runtime system maintains a category bit map for each user. The category bit map indicates the categories the user may access. When a user is granted access to a category, the corresponding bit in the category bit map is turned on.

The bit map is loaded on the first security request for a categorized resource which is internally secured. If none of the categorized resource types is internally secured, no category bit map will exist.

When the user attempts to access a resource that is protected by category, the resource category is checked against the category bit map of the user. Access is denied if the corresponding bit in the user’s category bit map is off.
Implementing Application Security

Contents
- Securing Activities (see page 82)
- Activity Security Processing (see page 83)
- CA ADS Security (see page 84)
- DCMT Security (see page 85)
- OCF/BCF Security (see page 86)
- Online Debugger Security (see page 86)
- Implementing Multi-level Application Security (see page 87)

About Activities

An activity is an application function defined as a resource to CA IDMS security. Activity security is an enhancement of the security class mechanism of Release 10.2.

You assign activity names and activity numbers to application functions with the CREATE RESOURCE STATEMENT. If you secure the ACTI resource internally, you grant users execution privilege on activities.

You can define up to 256 discrete activities for an application. You can choose to associate more than one function of the application with a given activity name and number.

⚠️ Note: If an application needs more than 256 activities, you can use multiple application names. The limit of 256 per application name is designed to provide upward compatibility with Release 10.2 systems and to keep the scheme simple and efficient.

Securing Activities

Why You Secure Activities

When you secure activities, you control who can execute a given application function.

Until you secure activities, any user can execute any application function.
How to Secure Activities

To secure activities internally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
RESTYPE=ACTI, X
SECBY=INTERNAL
```

To secure activities externally, include an entry in the SRTT:

```
#SECRTT TYPE=ENTRY, X
RESTYPE=ACTI, X
SECBY=EXTERNAL, X
```

Additional parameters required

Note: For more information, see #SECRTT (see page 160) and CREATE RESOURCES.

About External Activity Security

The EXTNAME values that you can specify for RESTYPE ACTI are RESNAME, APPLNAME, and ACTIVITY. For the purposes of security checking, the value of both RESNAME and APPLNAME is application-name. Only the ACTIVITY value contains the application function number. Thus, to secure individual activities externally, you must include ACTIVITY in the SRTT entry for RESTYPE ACTI.

Defining an Activity Resource

If you assign activity numbers to functions within the application, you can define each activity as a system resource with a CREATE RESOURCE statement. In a CREATE RESOURCE statement, you associate the application activity number with an external activity name. The activity name can be up to 18 characters and must be qualified with the application name, as in this example:

```
create resource
  activity dcmt.vary_terminals
  number 14;
```

Granting Execution Privilege on the Activity

After you have defined an activity, you can give users the privilege of executing the application functions represented by the activity, using a GRANT statement as in this example:

```
grant execute
  on activity dcmt.vary_terminals
  to support;
```

Activity Security Processing

Activity Bit Map

For each application that the user can access, the system maintains a bit map that indicates the activities the user is allowed to execute. The activity number corresponds to a bit position; for example, bit 43 corresponds to activity 43.
If the ACTI resource is secured internally, the application activity access bit map is brought into memory when the application issues its first security check request. The application bit map is chained to the signon block, and user and group activity access authorities are merged into the bit map.

When a user attempts to execute an activity, the application issues a call for a security check to the central security interface, specifying the application name and activity number. The application is expected to enforce the activity security based on the return code from the central security interface.

**Internal Security Check on an Activity**

If the ACTI resource is secured internally, the system searches for the activity bit map for the application and the user when the first security check for an application is processed. If the bit map is found, the system checks the activity number against the corresponding bit position in the activity access bit map, and returns a YES or NO answer to the application.

If the bit map is not found, the system then looks for an activity bit map for the user and the application named 'DEFAULT.' If found, the bit map for the DEFAULT activity is used in the security check.

**Application DEFAULT**

The DEFAULT application is a mechanism that allows execution of existing applications that use Release 10.2 security classes without having to explicitly define activities for each application. The security system will allow execution of an application function if:

- The security class assigned to that function matches the activity number of an activity in the DEFAULT application.
- The user has been granted execution privilege on that activity.

You can use the RHDCSMIG program to generate statement syntax to create 255 DEFAULT application activities (activity numbers 1 through 255) and to grant execution privilege on the activities to users who have the matching security classes in the Release 10.2 dictionary.

⚠️ **Note:** For more information on RHDCSMIG, see the *CA IDMS Conversion Guide*.

**External Security Check on Activity**

If the ACTI resource is secured externally, no activity bit map is involved in security checking. The central security interface creates the identifier of the activity that it passes to the external system by concatenating the shorter of the application name or the first five characters of the application name with the three-digit function number supplied by the application.

---

**CA ADS Security**

**Application Name**
The *application-name* you specify when defining a CA ADS activity must be the ADB name; that is, the name specified when the application was defined with the CA ADS application compiler (ADSA).

**Note:** The ADB name may differ from the task code or codes of the application. If necessary, you can determine the ADB name by viewing the application name by displaying the TAT table with DCMT DISPLAY MEMORY PROGRAM $ACF@TAT.

### CA ADS Security Classes

CA ADS allows you to associate a security class with the application and with any one or more application responses.

If the application has been assigned a security class, a security check is requested when a user attempts to execute the application. The user must be authorized to execute the activity whose activity number matches the security class of the application.

Similarly, if a response has been secured with a security class, a security check is requested when a user attempts to execute the response. The user must be authorized to execute the activity whose activity number matches the security class of the response.

**Note:** For more information on assigning security classes in CA ADS, see the CA ADS Reference section.

### DCMT Security

#### Application Name

When you create an activity name for the DCMT application, you specify DCMT for *application-name*. You associate the activity number with the DCMT commands you are grouping in the activity as follows.

#### Assigning DCMT Activity Numbers

DCMT provides the `#CTABGEN` macro for assigning activity numbers to DCMT commands. In the `#CTABGEN` macro, you associate an activity number with a DCMT command code.

**#CTABGEN Example**

In this example, `#CTABGEN` assigns the activity number of 14 to the DCMT commands as represented by their command codes -- N028 (VARY LTERM), N029 (VARY PTERM), and N030 (VARY LINE):

```
#CTABGEN (N028,14,N029,14,N030,14)
```
Release 10.2 DCMT Security

If you have implemented security classes for DCMT commands in Release 10.2, you need only reassemble #CTABGEN under Release 16.0 and define activities either specifically for DCMT or for the DEFAULT application, using RHDCSMIG output to generate CREATE RESOURCE ACTIVITY statements.

OCF/BCF Security

Application Name

When you create an activity name for a utility command, specify an application name of OCF for activities that are to be secured when running under a CV. To secure batch local mode utility activities, specify BCF for an application name. If the same command is to be secured in the same manner in both CV and batch, then two activity resources must be created.

Assigning OCF/BCF Activity Numbers

OCF/BCF security provides the #UTABGEN macro for assigning activity numbers to OCF/BCF utility commands. In the #UTABGEN macro, you associate an activity number with an OCF/BCF command code.

#UTABGEN Example

In this example, #UTABGEN assigns the activity number of 14 to the OCF/BCF commands FORMAT and PRINTPAGE as represented by their command codes:

#UTABGEN (FORMAT,14,PRINTPAGE,14)

Online Debugger Security

Application Name

When you create an activity name for the online debugger, you specify DBUG for application-name. You associate the activity number with the online debugger commands you are grouping in the activity as follows.

Assigning Online Debugger Activity Numbers
The CA IDMS/DC online debugger provides the #GTABGEN macro for assigning activity numbers to online debugger functions. In the #GTABGEN macro, you associate an activity number with an online debugger security Category.

**#GTABGEN Example**

Assume that you wish to assign the activity number of 20 to online debugger functions as represented by two online debugger security categories:

- AUPGMR (CA ADS user programs can be retrieved)
- USTGR (User storage can be retrieved)

To make this assignment, you would issue this #GTABGEN macro:

#GTABGEN(AUPGMR,20,USTGR,20)

**Note:** For #GTABGEN syntax and online debugger security categories, see Syntax for Assembler Macros.

---

**Implementing Multi-level Application Security**

**Multi-level Security**

You can secure applications at several levels.

For example, an online application with embedded SQL can be secured by the following:

- Task -- The ability to invoke the application
- Load module -- The ability to execute an application program
- Access module -- The ability to execute embedded SQL that accesses a database

**DCMT Example**

When you analyze your site's DCMT security requirements, keep in mind that you can implement security for DCMT commands at these levels:

- **At the task level**, you secure the DCMT task by assigning the task a Category using security administration statements.
- **At the program level**, you secure programs invoked for the DCMT task by assigning the programs a Category using security administration statements. Programs invoked for DCMT requests all have names that begin with RHDCMT (for example, RHDCMTPT or RHDCMTTI).
At the DCMT command level, you secure DCMT commands by means of the #CTABGEN macro. This macro is assembled into the program IDMSCTAB and allows you to apply discrete security to specific DCMT commands (such as, DCMT VARY PROGRAM) and also to individual command options (such as, DCMT VARY PROGRAM STORAGE PROTECT). The macro is used in conjunction with activity security in the security system to control access to specific DCMT functions.

Securing Database Resources

Database Security

Any user has the ability to access any database until it is secured. You can secure a database only by securing database resources. You can secure online access to databases at the task level, although this would involve securing the OCF task (online Command Facility) and other tasks. However, task security does not secure databases from batch access.

Even if the configuration of your external security system protects the database against local mode access, the database would not be protected from batch access through the central version unless the database is secured through CA IDMS centralized security.

Dictionaries and User catalog

The system dictionary, application dictionaries, and the user catalog are databases. If internal security is specified for one or more global resources, the user catalog contains security definitions. If internal security is specified for one or more database resources, the system dictionary contains security definitions, as does the application dictionary if the SQL Option is installed.

To secure dictionaries and the user catalog, you must secure database resources as described in this section.

For more information, see the following topics:
- About Database Resources (see page 88)
- Securing Common Database Resources (see page 92)
- Securing Access to Non-SQL-Defined Databases (see page 99)
- SQL Security Enforcement (see page 102)
- Securing SQL Access to Databases (see page 112)
- Securing the Dictionaries and the User catalog (see page 118)

About Database Resources

Before you implement a scheme for database security using information in this section, become familiar with the concepts discussed in this section.

Contents
Securing Database Resources in the SRTT

Database Resources

The security option in the SRTT for the DB resource type determines whether database resources other than DMCLs and database name tables (DBTB) are secured externally, internally, or not at all.

SRTT entries for resource types other than DB, DMCL, and DBTB are used in runtime security processing only if security for DB is external; in that case, the SRTT entry is used only to determine the external resource class and resource name to send with the security check request to the external system.

The following table lists CA IDMS database resources and their keyword equivalents for the #SECRRT RESTYPE parameter:

<table>
<thead>
<tr>
<th>Database resource</th>
<th>RESTYPE keyword to secure resource</th>
<th>RESTYPE keyword for external information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>DB</td>
<td>DB</td>
</tr>
<tr>
<td>DBADMIN privilege</td>
<td>DB</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Access module</td>
<td>DB</td>
<td>DACC</td>
</tr>
<tr>
<td>Area</td>
<td>DB</td>
<td>AREA</td>
</tr>
<tr>
<td>Run unit</td>
<td>DB</td>
<td>NRU</td>
</tr>
<tr>
<td>SQL-defined schema</td>
<td>DB</td>
<td>QSCH</td>
</tr>
<tr>
<td>Non-SQL-defined schema</td>
<td>DB</td>
<td>NSCH</td>
</tr>
<tr>
<td>Table</td>
<td>DB</td>
<td>TABL</td>
</tr>
<tr>
<td>Database name table</td>
<td>DBTB</td>
<td>DBTB</td>
</tr>
<tr>
<td>DMCL</td>
<td>DMCL</td>
<td>DMCL</td>
</tr>
</tbody>
</table>

DB Occurrence Overrides

You can specify DB occurrence overrides in the SRTT. For example, if security for databases is off but you add an internal security occurrence override for database PROD, the runtime system will route a security check on a database resource to internal security if the database name on the current security request begins with 'PROD'.

You cannot override the automatic assignment of the DB security option to the other database resources. For example, if security for DB is off, security for the AREA resource type is also off and the security option specified on the SRTT entry for AREA is ignored. However, external resource class and name information in an SRTT entry for a database resource type such as AREA is used if external security is specified on the entry for DB or on a DB occurrence override.
Database Security and Database Names

Segment Names and Database Names

The database name specified on a BIND RUN-UNIT statement or a CONNECT statement can be either a segment name or a database name defined in the database name table. If you secure all databases, a security check will be routed to the enforcing system on BIND RUN-UNIT statements and on database definition and access statements issued following a CONNECT.

**Note:** To issue the CONNECT statement itself under the central version, the user must have signon authority for the system with which the dictionary named in the statement is associated and authority to invoke the task or application from which the CONNECT is issued.

However, if you plan to leave some databases unsecured, you must consider how CA IDMS processes a database name before you build database security in the SRTT.

Role of the Database Name Table

If an application requests a bind to a database or a connection to a dictionary, CA IDMS searches the database name table for the name specified on the BIND or CONNECT. If it finds a match, CA IDMS determines the areas and files to be accessed based on the segments that are included in the database name. If it does not find a match in the database name table, CA IDMS searches for a matching segment name in the DMCL. If no match is found, an error results.

Securing Access to Individual Segments

To understand how access to segments is secured, consider this sample database name table:

<table>
<thead>
<tr>
<th>Database name</th>
<th>Segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSTEM</td>
</tr>
<tr>
<td></td>
<td>CATSYS</td>
</tr>
<tr>
<td></td>
<td>SYMSMG</td>
</tr>
<tr>
<td>DIRLDICT</td>
<td>DIRLNWK</td>
</tr>
<tr>
<td></td>
<td>CATSYS</td>
</tr>
<tr>
<td></td>
<td>SYMSMG</td>
</tr>
</tbody>
</table>

If the entry for DB is security 'OFF', you would obtain these results using occurrence overrides:

- **If you secure 'SYSTEM',** access to the SYSTEM segment is secured. Access to CATSYS and SYMSMG through dbname SYSTEM is secured, but access to these segments directly or through dbname DIRLDICT is not secured.

- **If you secure 'SYSTEM' and 'DIRLDICT',** access to the SYSTEM segment is secured. Access to CATSYS, SYMSMG, and DIRLNWK through dbnames is secured, but direct access to these segments is not secured.
If you secure 'SYSTEM', 'DIRLDICT', 'CATSYS', and 'SYSMSG', access to all segments but DIRLNWK is secured.

Therefore, to achieve complete database security using occurrence overrides, you must secure all segments to be protected and all dbname's that include one or more of those segments.

Securing the Database Name Table

To maintain database security that is based on occurrence overrides, you must secure database name tables that are included in DMCLs. If a database name table is not secure, a knowledgeable user could create or modify the definition of a database name that is not secured to include otherwise secure segments.

Note: For more information, see Securing Database Name Tables (see page 98).

Internal Security for Database Resources

Privileges on Common Database Resources

The following table presents the privileges in CA IDMS internal security that apply to use of database resources common to both SQL-defined and non-SQL-defined databases:

<table>
<thead>
<tr>
<th>Privilege</th>
<th>DB</th>
<th>AREA</th>
<th>DMCL</th>
<th>DBTABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ALTER</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DROP</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>X</td>
<td>(1)</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>USE</td>
<td>(1)</td>
<td>(1)</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DBAREAD</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBAWRITE</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBADMIN</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) Privilege applicable only to non-SQL-defined databases.

DBADMIN can be granted to any other user by a holder of SYSADMIN or DBADMIN. All other privileges are grantable if a holder of SYSADMIN or DBADMIN grants them using the WITH GRANT OPTION parameter. A grantable privilege means that the recipient of the privilege can grant it to another user.

Definition Privileges
CREATE, ALTER, DROP, and DISPLAY control the user's ability to manipulate the definition of an object. To issue any definition statement other than DISPLAY on the common database resources, the user must also hold DBADMIN authority on the dictionary to which the session is connected when the statement is issued, if DB security is enabled for the dictionary.

**USE Privilege**

The following table explains the type of access that the USE privilege authorizes:

<table>
<thead>
<tr>
<th>Resource</th>
<th>What USE privilege permits the user to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB</td>
<td>Associate a secured segment with an SQL schema</td>
</tr>
<tr>
<td>NSCH</td>
<td>Associate a secured non-SQL-defined schema with an SQL schema</td>
</tr>
<tr>
<td>AREA</td>
<td>Create an SQL table or index in a secured area</td>
</tr>
<tr>
<td>DMCL</td>
<td>Punch the load module of a secured DMCL and execute utilities on the journal files defined by that DMCL</td>
</tr>
<tr>
<td>DBTABLE</td>
<td>Punch the load module of a secured database name table and associate a database name table with a DMCL.</td>
</tr>
</tbody>
</table>

(1) NSCH is a common database resource in the sense that it represents a non-SQL-defined entity and is meaningful in SQL processing.

**DBREAD and DBWRITE Privileges**

The DBREAD and DBWRITE privileges are granted to permit users to execute utility functions on areas of the database. DBREAD privilege allows the user to execute utilities that require read-only access to an area. DBWRITE privilege allows the user to execute utilities that require read-write access to an area.

---

**Securing Common Database Resources**

**Contents**

- Securing Databases (see page 93)
- Securing DBADMIN (see page 95)
- Securing Areas (see page 95)
- Securing DMCLs (see page 96)
- Securing Database Name Tables (see page 98)

**About the DB Resource**

Security on the DB resource type in the SRTT automatically activates security on the following resources:

- Database
- DBADMIN
Area
Run unit
SQL schema
Non-SQL-defined schema
Table
Access module

Internal Security for Databases
If you secure databases internally, you grant privileges to allow users access to database resources.

External Security for Databases
If you secure databases externally, you specify rules in the external security system for accessing database resources. You must also provide external resource class and name information in the SRTT entry for the database resources represented by these resource type keywords:

- DB
- AREA
- NRU
- QSCH
- NSCH
- TABL
- DACC
- SACC

⚠️ Note: For more information, see #SECRRT (see page 160).

Securing Databases

About Databases

When you secure resource type DB, you control who can issue DDL SEGMENT statements and who can specify a segment in the DBNAME parameter of a CREATE SCHEMA statement. Until you secure resource type DB, any user can issue DDL SEGMENT statements and can specify a segment in the DBNAME parameter of a CREATE SCHEMA statement.
How to Secure Databases

To secure the DB resource internally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,             X
   RESTYPE=DB,            X
   SECBY=INTERNAL
```

To secure the DB resource externally, include an entry in the SRTT:

```
#SECRTT    TYPE=ENTRY,             X
   RESTYPE=DB,            X
   SECBY=EXTERNAL,        X
   Additional parameters required
```

⚠️ **Note:** For more information on #SECRTT, see #SECRTT (see page 160).

Database Occurrence Overrides

You can specify a security option for a particular occurrence of a database that differs from the option specified for DB in the SRTT. This allows you, for example, to secure databases internally but to leave security 'OFF' for specific databases.

In this example, internal security is activated in the SRTT for all databases in the system (including the system dictionary and the user catalog), but security is turned off for any databases with names that begin with 'TEST' or 'DEMO'.

```
#SECRTT    TYPE=ENTRY,             X
   RESTYPE=DB,            X
   SECBY=INTERNAL

#SECRTT    TYPE=OCCURRENCE,        X
   RESTYPE=DB,            X
   RESNAME='TEST',        X
   SECBY=OFF

#SECRTT    TYPE=OCCURRENCE,        X
   RESTYPE=DB,            X
   RESNAME='DEMO',        X
   SECBY=OFF
```

How to Grant Database Definition Privilege

To give physical database definition privileges, you issue a GRANT statement on the DB resource type, specifying the privilege or privileges and the name of the database. You can specify any combination of CREATE, ALTER, DROP, DISPLAY, and USE privileges, or you can specify all definition privileges (DEFINE). You must be connected to the system dictionary.

As a holder of SYSADMIN or DBADMIN privilege, you can specify WITH GRANT OPTION when you grant definition privileges to allow the recipient to grant the same privileges to another user.

⚠️ **Note:** For more information, see the following sections:
Securing DBADMIN

About DBADMIN

When you secure resource type DB, you control who can manipulate database definitions and database-related objects. Until you secure DBADMIN, any user can manipulate database definitions and database-related objects.

How to Secure DBADMIN

You secure DBADMIN by securing the DB resource.

⚠️ Note: For more information, see Securing Databases (see page 93).

How to Grant DBADMIN Privilege

You give DBADMIN privilege on a named database to a user or group with a GRANT DBADMIN statement. You must hold SYSADMIN or the appropriate DBADMIN privilege to grant DBADMIN privilege. You must be connected to the system dictionary.

⚠️ Note: For more information, see the following sections:

- GRANT Administration Privilege (see page 201)
- REVOKE Administration Privilege (see page 207)

Securing Areas

About Areas
When you secure resource type DB, you can control who can access an area through a CA IDMS utility and who can create tables and indexes to be stored in the area. Until you secure resource type DB, any user can access an area through a CA IDMS utility and create tables and indexes to be stored in the area.

**How to Secure Areas**

You secure areas by securing the DB resource.

![Note: For more information, see Securing Databases (see page 93).]

If you secure areas externally, you must also include an entry in the SRTT with external security information for resource type AREA.

**How to Grant Area Access and Use Privileges**

To give area access privileges, you issue a GRANT statement on the area resource type, specifying the privilege or privileges and identifying the area. You can specify any combination of DBAREAD, DBAWRITE, and USE privileges.

DBAREAD and DBAWRITE privileges allow read-only and read-write access to an area using CA IDMS Utilities. USE privilege allows creation of a table or index in the area.

As a holder of SYSADMIN or DBADMIN privilege, you can specify WITH GRANT OPTION when you grant these privileges to allow the user to grant the same privileges to another user. You must be connected to the system dictionary.

**More Information:**

- For more information, see **GRANT Area Access Privileges (see page 245).**
- For more information, see **REVOKE Area Access Privileges (see page 265).**
- For more information on utilities, see the **CA IDMS Administrating section.**
- For more information on creating tables and indexes, see the **CA IDMS SQL Reference section.**

**Securing DMCLs**

**About DMCLs**

When you secure the DMCL resource type, you can control who can do the following:

- Issue DDL DMCL statements
- Display or punch the DMCL load module
### Execute utilities which operate against DMCL journal files

Until you secure the DMCL resource type, any user can issue DDL DMCL statements if database security is not in effect, and any user perform the other previously listed functions.

### How to Secure DMCLs

If the system dictionary is secured, DBADMIN privilege on the system dictionary is required to manipulate DMCL definitions.

#### Note:
For more information, see Securing the Dictionaries and the User catalog (see page 118).

The following discussion applies to securing DMCLs explicitly.

To secure DMCLs internally, include an entry in the SRTT:

```
#SECRRT TYPE=ENTRY, X
RESTYPE=DMCL, X
SECBY=INTERNAL
```

To secure DMCLs externally, include an entry in the SRTT:

```
#SECRRT TYPE=ENTRY, X
RESTYPE=DMCL, X
SECBY=EXTERNAL, X
```

Additional parameters required

#### Note:
For more information, see #SECRRT (see page 160).

### How to Grant DMCL Definition and Use Privileges

To allow a user to create and maintain a DMCL definition, you issue a GRANT statement on the DMCL resource type, specifying the privilege or privileges and identifying the DMCL.

You can specify any combination of CREATE, ALTER, DROP, DISPLAY, and USE privileges, or you can specify all privileges (DEFINE).

#### Note:
The USE privilege allows the user to punch the DMCL load module and format journal files defined in the DMCL.

As a holder of the applicable SYSADMIN or DBADMIN privilege, you can specify WITH GRANT OPTION when you grant these privileges to allow the recipient to grant the same privileges to another user. You must be connected to the system dictionary.
More Information:

- For more information, see GRANT Physical Database Definition Privileges (see page 249).
- For more information, see REVOKE Physical Database Definition Privileges (see page 268).
- For more information on DMCL statements, see the CA IDMS Database Administering section.

Securing Database Name Tables

About Database Name Tables

When you secure the database name table resource type, you control who can issue DBTABLE definition statements and who can display or punch the DBTABLE load module. Until you secure the database name table resource type, any user can issue DBTABLE definition statements if database security is not in effect, and any user can display or punch the DBTABLE load module.

How to Secure Database Name Tables

If the system dictionary is secured, DBADMIN privilege on the system dictionary is required to manipulate database name table definitions.

⚠️ Note: For more information, see Securing the Dictionaries and the User catalog (see page 118).

The following discussion applies to securing database name tables explicitly.

To secure database name tables internally, include an entry in the SRTT:

```plaintext
#SECRTT    TYPE=ENTRY,          X
              RESTYPE=DBTB,       X
              SECBY=INTERNAL
```

To secure database name tables externally, include an entry in the SRTT:

```plaintext
#SECRTT    TYPE=ENTRY,          X
              RESTYPE=DBTB,       X
              SECBY=EXTERNAL,     X
Additional parameters required
```

⚠️ Note: For more information, see #SECRTT (see page 160).

How to Grant Database Name Table Definition Privilege
To allow a user to create and maintain a database name table definition, you issue a GRANT statement on the database name table resource type, specifying the privilege or privileges and identifying the database name table. You can specify any combination of CREATE, ALTER, DROP, DISPLAY, and USE privileges, or you can specify all privileges (DEFINE).

**Note:** The USE privilege allows the user to punch the database name table load module and to associate the database name table with a DMCL.

As a holder of the applicable SYSADMIN or DBADMIN privilege, you can specify WITH GRANT OPTION when you grant these privileges to allow the recipient to grant the same privileges to another user. You must be connected to the system dictionary.

**More Information:**
- For more information, see GRANT Physical Database Definition Privileges (see page 249).
- For more information, see REVOKE Physical Database Definition Privileges (see page 268).
- For more information on DBTABLE statements, see the CA IDMS Database Administering section.

### Securing Access to Non-SQL-Defined Databases

**Contents**
- Securing Run Units (see page 100)
- Schema and Subschema Security (see page 101)
- SQL Access to a Non-SQL-Defined Database (see page 102)
- Using Database Procedures (see page 102)

**How to Do It**

To secure access to a non-SQL-defined database, these resource types must be secured:

- **DB**
- **DBADMIN** (internal security)
- **AREA**
- **NRU**

Depending on your database definitions and runtime environment, you may also need to secure these resource types:

- **DBTB**
- **DMCL**
**Securing Run Units**

**About Run Units**

The primary access to data stored in non-SQL-defined databases is through a subschema using navigational DML. This type of database transaction is a run unit.

A run unit is started by a BIND RUN-UNIT command, which effectively opens a logical view of the database as defined by the subschema. Within the subschema, privacy locks and LRF path logic restrict the operations which can be performed on the data accessible through the subschema.

When you secure run units, you control who can access a non-SQL-defined database through navigational DML. Until you secure run units, any user can access a non-SQL-defined database through navigational DML.

**About the Run Unit Resource**

At bind time, the following elements are available for security checking purposes:

- The name of the database being accessed
- The name of the subschema
- The original compilation name of the program issuing the BIND

The combination of these three elements -- `database-name.subschema-name.program-name` -- identify the run unit resource on which authority is checked when access to a non-SQL-defined database using navigational DML is requested.

**How to Secure Run Units**

You secure run units by securing the DB resource.
If you secure run units externally, you must also include an entry in the SRTT with external security information for resource type NRU.

**How to Grant Execution Privilege on a Run Unit**

Run units must be categorized before you can grant execution privilege on them. For example, this statement assigns run units for a group of programs that access the PRODSCHM database to the Category EMPINFO:

```bash
create resource
    Category empinfo
```

**Note:** For more information, see `CREATE RESOURCE (see page 217)`.

To allow a user to execute a run unit, you issue a GRANT EXECUTE statement on the Category that contains the run unit. For example, this statement gives privilege on the EMPINFO Category to two groups of users:

```bash
grant execute on
    Category empinfo
    to hr_mgrs, corp_execs;
```

**Note:** For more information, see `GRANT Execution Privilege (see page 224)`.

**Runtime Run Unit Checking**

If the security option for DB is internal, BIND processing requests a check of privilege on the Category to which the run unit is assigned. The internal security system uses dynamic table support to maintain a sorted cache of run units and their associated Category.

If the security option for DB is external, the check request is routed to external security with the class and resource name information specified on the SRTT entry for NRU.

**Schema and Subschema Security**

You can control who can use CA IDMS compilers to define non-SQL-defined schemas and subschemas by implementing internal dictionary security.

**Note:** For more information on dictionary security, see `Securing Application Dictionary Resources (see page 121)`.
SQL Access to a Non-SQL-Defined Database

With the CA IDMS SQL Option, a user can access a non-SQL-defined database using SQL. If the database is secured, you allow SQL access to a non-SQL database by creating an SQL schema for a non-SQL schema (this requires CREATE privilege on the SQL schema, USE privilege on the non-SQL schema) and granting table access privileges to other users.

Note: For more information, see the following sections:

- Securing SQL schemas (see page 112)
- Securing Non-SQL-defined Schemas (see page 113)

Using Database Procedures

What You can Do

You can use a database procedure to enforce user-defined security rules for non-SQL-defined databases. The procedure can make use of centralized security to actually do the checking which in turn will invoke an external package if appropriate.

What is a Database Procedure

A database procedure is a user-written exit specified as part of the schema definition of the non-SQL-defined database. Both area- and record-level procedures are supported. They allow you to specify which non-SQL DML verbs will cause invocation of the procedure.

Advantages of Database Procedures

- You can choose resource names for records which are meaningful in your environment.
- You can selectively check privileges based on the record or area and the type of access being performed, limiting the additional overhead incurred to only those resources which require a higher degree of security checking.
- Within the procedure you can apply additional rules based on such information as the content of the record (the equivalent of row-level security), the program name being used, and the contents of the BIND PROCEDURE block as passed from the application program.
- Facilities for caching information are provided.

SQL Security Enforcement
Overview

The security scheme for SQL processing is both based upon, and compliant with, the ANSI SQL standard.

The major secured resource in SQL processing is the table resource type. CA IDMS internal security validates a user’s right to access a table according to the rules specified in the ANSI SQL standard. Additional facilities control table creation and alteration.

If resources accessed in SQL processing are secured externally, rules for accessing each resource type must be defined to the external security system. Reference to external security in the following discussion is limited to information on how CA IDMS centralized security processes a request for an externally secured resource.

SQL Resources

Table

The table resource type represents base tables, functions, procedures, table procedures and views.

A view is a logical table derived from information in one or more base tables, table procedures or views. CA IDMS centralized security does not distinguish between table-like objects, but special security considerations pertain to view access.

Note: For more information, see Securing Views (see page 109).

Access Module

The access module resource represents a set of precompiled SQL statements. The owner of the access module must hold all necessary table access privileges in order to execute the access module. The access module owner can grant execution privilege to other users if the owner holds grantable table access privileges.

Note: For a more detailed discussion of access module security, see Runtime Security for Access Modules (see page 107).
Schema

The schema resource is equivalent to the ANSI SCHEMA construct. All tables, functions, procedures, table procedures, views, and access modules are contained within a schema. The schema name becomes the high-level name qualifier for all subordinate entities.

The schema also designates ownership of resources. The owner of a schema owns all resources within the schema. Ownership is established when the schema is created.

CA IDMS Privileges

Privileges and Resources

The following tables summarizes the resources and CA IDMS privileges that apply to those resources in SQL processing if the resources are secured internally:

<table>
<thead>
<tr>
<th>Privilege</th>
<th>TABLE</th>
<th>SCHEMA</th>
<th>ACCESS MODULE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATE</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXECUTE</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CREATE</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ALTER</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DROP</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>(1)</td>
<td>(1)</td>
<td>(2)</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) Privilege to display the resource and privileges on it.

(2) Privilege to issue the EXPLAIN statement on the module and to display the resource and privileges on it.

All privileges are grantable when a holder of SYSADMIN or DBADMIN privilege grants them using the WITH GRANT OPTION parameter. This allows the recipient of the privilege to grant it to another user.

CA IDMS internal security specifically checks for grantability of privileges when it processes a security check on view and access module resource types.

⚠️ **Note:** For more information on runtime security checks on views and access modules, see the following sections:

- Securing Views (see page 109)
Access Privileges

SELECT, INSERT, UPDATE, and DELETE privileges control a user's ability to access data. These privileges are defined according to the ANSI SQL standard.

Definition Privileges

CREATE, ALTER, DROP, DISPLAY, and REFERENCES control the user's ability to manipulate the definition of an object or, in the case of REFERENCES, control a user's ability to reference a table in a referential constraint definition.

Access Module Execution Privilege

The EXECUTE privilege allows the user to execute an access module. The privilege to execute an access module can also be held through the Category mechanism.

If an access module has been assigned to a Category, a user must hold privilege on the Category to execute the access module. In this situation, an individual grant of execution privilege on the access module is ignored by the security system as long as the Category exists and the access module remains in it.

Security Checking for Interactive and Dynamic SQL

Dynamic Checking

When a user executes a tool which allows SQL statements to be entered explicitly (or implicitly as a result of information provided on a form), security checking is performed as each statement is processed. This is also true if a user-written program issues dynamic SQL statements.

The authorization ID against which the privileges are checked is the authorization ID of the executing user, except in certain instances associated with view access, as discussed in Securing Views (see page 109). Every privilege required to execute a given SQL statement is checked. The results of the check are cached for the life of the database transaction, to avoid repetitive authorization checks for similar access to the same table-like object.

Securing the Dynamic SQL Statement cache

If dynamic SQL statement caching is in effect, users and administrators should keep in mind that the SQL cache contains images of the source of dynamic SQL statements. These images can possibly contain confidential information. The source of the cached SQL statements is accessible through the table procedure SYSCA.DSCCACHE and any view (that is SYSCA.DSCCACHEV delivered during installation), that projects the STATEMENT column of this table procedure. It is recommended to secure access to the SYSCA.DSCCACHE table procedure and any view exposing the column STATEMENT of SYSCA.DSCCACHE.

External Security
If security on the database being accessed is controlled externally, the security checks are issued by CA IDMS as it executes the commands and the authorization permissions are cached for the life of the transaction or task whichever ends first.

**CA IDMS Internal Security**

If security on the database being accessed is controlled by CA IDMS internal security, security checks are issued by the access module compiler (AMC) as it compiles the dynamic SQL command.

### Security Checking for Precompiled SQL Statements

#### Precompiled SQL Statements

User-written programs or SQL routines may contain embedded SQL statements that are precompiled and included in an access module prior to runtime. Security checking for embedded SQL statements is performed in one of two ways, depending on how security on the database being accessed is controlled.

**External Security**

If external security is in effect for the database, dynamic security checking is performed on all SQL statements, precompiled or not. When the SQL session is started, a security check determines if external security is in force for the database to which the session is connected. If so, this information is cached for the duration of the session.

CA IDMS issues the security checks as it executes each statement and caches the information for the life of the database transaction or task. The name of the access module is passed as part of the security check and is used as an authorized program filter.

This method of security checking for SQL statements complies with government requirements.

**CA IDMS Internal Security**

If CA IDMS internal security is in effect for the database, security checking for precompiled SQL statements takes a pre-authorized approach that requires the owner of the access module to hold all privileges necessary to execute every SQL statement in the module. For example, the owner must hold the appropriate table access privilege for each table accessed by an SQL statement in the module.

If this condition is met, then the owner of the access module can execute it. The owner can give execution privilege on the access module to other users if the owner holds the necessary grantable privileges.

**Advantages of the Pre-authorized Approach**

The pre-authorized security approach for SQL statements minimizes the overhead of security checking at runtime.
It also eliminates the need to grant all users the privileges needed to execute the SQL statements in
the access module. Only the owner must have those privileges; other users simply require execution
privilege on the access module. This means that executing the program is the only way the users can
access the resources because they hold no privileges independent of the access module.

The pre-authorized security approach for SQL statements complies with the ANSI SQL standard.

Runtime Security for Access Modules

Overview

Each time a new copy of the access module is physically loaded by the runtime system, the privileges
of the access module owner are checked.

The result of the security check performed on a new copy of an access module is the status of the
access module. This information is cached in the PDE until a new copy is loaded (or until the system is
recycled).

The status of the access module is one of the following:

- Not runnable -- No one can execute the access module.
- Runnable -- The owner holds all necessary privileges to execute the access module.
- Runnable/grantable -- The owner holds all necessary privileges to execute the access module and
to grant execution privilege on the access module to other users.

Runtime checking is required because a grantable privilege needed to pass execution privilege to
users could be removed from the owner. Revocation of a privilege occurs independently of both CA
IDMS and CA IDMS inter security.

A detailed description of runtime procedures for access module security follows.

On a Load of an Access Module

On a load of any access module, the CA IDMS program load function issues two security check
requests to verify that the user has execution privilege for the access module. The first security check
is for load privilege on the access module. An access module is just like any other load module. When
loaded, you need to check to see if the user has the authority to load the program. The second
security check is issued to see if the requestor has authority to access the database using the SQL
statements that are stored in the access module.

- If the checks fail, an error is returned.
- If the checks succeed, normal load processing occurs.

If a new copy of the access module is being loaded and the security check on the user succeeds, CA
IDMS program load processing:

1. Calls a database routine to scan the module and return this information on the access
module:
1. Status (Not runnable, runnable, runnable/grantable).
   

   This information is cached.

2. The CA IDMS program load function requests a check to determine if the user is allowed to execute the access module based on its status:

   - If status is not runnable, then the check fails.
   - If status is runnable/grantable, then the check succeeds.
   - If status is runnable and the user (or a group to which the user belongs) is the owner of the access module, then the check succeeds; if status is runnable but the user is not the owner, the check fails.

3. Depending on the result of the preceding step, the program requesting the load receives a return code indicating one of these conditions:

   - Security violation -- The user does not have execution privilege on the access module.
   - Runnable -- The user has execution privilege on the access module.
   - Not grantable -- The user cannot execute the access module because the owner does not hold all required privileges.

On a CREATE or ALTER ACCESS MODULE Command

The access module compiler issues a security check to determine whether the user has the CREATE or ALTER privilege on the access module. If not, an error is issued and no further processing is done.

If the security check succeeds, the access module compiler creates the access module. It then calls the database routine used by the CA IDMS program loader function to check the access module owner’s privileges. The owner of the access module is the owner of the associated schema. The user who submits the CREATE or ALTER ACCESS MODULE statement is notified if the access module owner lacks any required privileges.

**Note:** The access module is stored whether or not the owner holds all required privileges. If you subsequently grant the owner any missing privileges, the access module will be runnable.

On Dynamic Compilation of an SQL Statement

There are three situations that require dynamic compilation of SQL statements:

- The statement is submitted through the Command Facility.
- A user-written program submits a dynamic SQL statement.
CA IDMS determines that database changes require an access module to be recompiled.

In these situations, the access module compiler checks privileges as the statements are being compiled. The result of this checking is based on the same criteria used for a load:

- If status is not runnable, then the check fails.
- If status is runnable/grantable, then the check succeeds.
- If status is runnable and the user (or a group to which the user belongs) is the owner of the access module, then the check succeeds; if status is runnable but the user is not the owner, the check fails.

**Securing Views**

**Views in Security Strategy**

There are special security considerations associated with creating and accessing views. A view is a logical table derived from one or more base tables, table procedures or views. You can use a view to restrict a user's access to specific columns and rows of the underlying tables.

The benefit of views from a security perspective is that you can give users access to the view without giving them equivalent privileges on the underlying tables.

Checking of view privileges is performed at runtime when either of these cases occurs:

- A dynamic SQL statement is executed (the result of the check is cached until end of transaction).
- A new copy of an access module is physically loaded (the result of the check is cached until another copy is loaded or the system is recycled).

In either case, the actual checks made are identical except for the authorization ID used for checking access to the view:

- The executing user, when statements are executed dynamically.
- The owner of the access module, when statements are precompiled.

**View ownership**

The method used by CA IDMS internal security to secure views employs information on view ownership. A table-like object, such as a view, is owned by the owner of the schema with which it is associated.

When CA IDMS internal security is in effect for views, a user gains access to the view only if:

- The user holds SELECT privilege on the view.
- The owner of the view holds grantable SELECT privilege on each table-like object referenced by the owner’s view.
This approach to view security complies with ANSI standards for SQL.

View example

In the view depicted by the following chart, SCHEMA_A.V1 is owned by user JOHN and references table SCHEMA_B.T2 owned by JANE and view SCHEMA_C.V2 owned by MARY. View V2 in turn references table SCHEMA_D.T3 owned by MIKE.

- SCHEMA_A.V1 (owner JOHN)
- SCHEMA_B.T2 (owner JANE)
- SCHEMA_C.V2 (owner MARY)
- SCHEMA_D.T3 (owner MIKE)

What CA IDMS Internal Security Checks

Using the preceding view example, assume that user FRED is issuing a dynamic SELECT against view V1. The security system will check to see the following:

- FRED, the executing user, holds SELECT privilege on view V1
- JOHN, the owner of view V1 holds:
  - Grantable SELECT privilege on table T2
  - Grantable SELECT privilege on view V2
- MARY, the owner of view V2 holds grantable SELECT privilege on table T3

The security system uses the schema name qualifier of the table-like object to determine the authorization ID to be checked. The authorization ID of the schema owner is stored in a row of the SYSTEM RESOURCEGROUP table. For example, to determine security requirements on view SCHEMA_A.V1, CA IDMS in effect asks the security system to check whether the owner of SCHEMA_A has grantable SELECT privilege on table T2 and view V2.

External Security Enforcement for Views

If external security is in effect, only the executing user's privilege to access the view is checked. Neither owner privileges nor authorities to access base tables are checked.
Securing SQL routines

SQL routines are SQL-invoked procedures or functions written in the SQL procedural language, that require a number of different objects to be built, processed and executed in the dictionary. The following table gives an overview of these objects showing the dictionary area containing the object, the dictionary entity, the entity subtype, and the name and description. The objects are created while processing the create command for an SQL routine.

<table>
<thead>
<tr>
<th>Dictionary Area</th>
<th>Entity Type</th>
<th>Name and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDLCAT</td>
<td>TABL</td>
<td>A table-like object (procedure or function) contained in the routine schema of the SQL catalog</td>
</tr>
<tr>
<td>DDLCATX</td>
<td>E</td>
<td>PROFCE or FUNCTION</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DDLCAT</td>
<td>TABL</td>
<td>LOCAL VARIA BLES OWNE R</td>
</tr>
<tr>
<td>DDLCATX</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DDLDML</td>
<td>MODULE</td>
<td>An IDD process module that contains the CA ADS source code, generated for the SQL routine. The name is constructed from ExternalName, as follows: PREMAP-ExternalName (1)</td>
</tr>
<tr>
<td></td>
<td>PROCE SS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DDLDCLOD</td>
<td>LOAD RCM MOD ULE</td>
<td>The RCM (Resource Control Module) module associated with the SQL routine. The name of the RCM is given by External Name (2)</td>
</tr>
<tr>
<td></td>
<td>DIALOG</td>
<td>The dialog module associated with the SQL routine. The name of the dialog is given by ExternalName</td>
</tr>
<tr>
<td></td>
<td>AM</td>
<td>The AM (Access Module) contained in the routine schema and associated with the SQL routine. The name of the AM is given by ExternalName (2)</td>
</tr>
<tr>
<td></td>
<td>AM LOAD AM MOD ULE</td>
<td></td>
</tr>
</tbody>
</table>

(1) The object is not accessed when the SQL routine is invoked.

(2) The object is only accessed when the AM needs to be recompiled.

To invoke an SQL routine a user must have the SELECT privilege on the SQL procedure or function and the EXECUTE privilege on the AM associated with the SQL routine. If CA ADS run time security is implemented then additional privileges might be required to execute the CA ADS dialog associated with the SQL routine.

To create an SQL routine the user must have the CREATE and SELECT privileges on the SQL procedure or function and the CREATE, ALTER and DROP privileges on the AM associated with the SQL routine.
To drop an SQL routine the user must have the DROP privileges on the SQL procedure or function and the DROP privilege on the AM associated with the SQL routine.

⚠️ **Note:** Dropping an SQL routine will remove all the associated dictionary objects. The optionally granted privileges will also have been removed, except for the privileges granted on the access module, which are preserved. A REVOKE command can be used to remove the privileges if needed.

### How to Grant Privileges to Invoke an SQL Procedure

Assume the SQL procedure GET_EMPLOYEE contained in the SQL schema HR_APPL with an external name GETEMPL needs to be called by users of the group HR_DEP.

```
grant select on HR_APPL.GET_EMPLOYEE to HR_DEP;
grant execute on access module HR_APPL.GETEMPL to HR_DEP;
```

### How to Grant Privileges to Define and Maintain an SQL Procedure

Assume the SQL procedure GET_EMPLOYEE contained in the SQL schema HR_APPL with an external name GETEMPL needs to be defined and maintained by users of the group HR_DEV.

```
grant define on HR_DEV.GET_EMPLOYEE to HR_DEV;
grant select on HR_DEV.GET_EMPLOYEE to HR_DEV;
grant define on access module HR_DEV.GETEMPL to HR_DEV;
grant execute on access module HR_APPL.GETEMPL to HR_DEV;
```

---

### Securing SQL Access to Databases

**Contents**

- Securing SQL schemas (see page 112)
- Securing Non-SQL-defined Schemas (see page 113)
- Securing Tables (see page 114)
- Securing Access Modules (see page 116)

**About SQL Access**

Users have the capability of accessing both SQL-defined and non-SQL-defined databases with SQL DML. Granting SQL access to a secured non-SQL-defined database is the same as granting access to a secured SQL-defined database with the additional step of granting USE privilege on non-SQL-defined schemas.

### Securing SQL schemas

**About SQL Schemas**

When you secure resource type DB, you control who can create an SQL schema. Until you secure resource type DB, any user can create an SQL schema.
How to Secure SQL Schemas

You secure SQL schemas by securing the DB resource.

⚠️ Note: For more information, see Securing Databases (see page 93).

If you secure SQL schemas externally, you must also include an entry in the SRTT with external security information for resource type QSCH.

How to Grant Definition Privileges on an SQL Schema

To allow a user to create an SQL schema, you issue a GRANT statement on the SQL schema resource type, specifying the privilege or privileges and identifying the SQL schema. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can specify all definition privileges (DEFINE).

As a holder of SYSADMIN or DBADMIN privilege, or as owner of the schema, you can specify WITH GRANT OPTION when you grant definition privileges to allow the recipient to grant the same privileges to another user.

More Information:

- For more information, see GRANT SQL Definition Privileges (see page 253).
- For more information, see REVOKE SQL Definition Privileges (see page 271).
- For more information on creating SQL schemas and transferring schema ownership, see the CA IDMS SQL Reference section.

Securing Non-SQL-defined Schemas

About Non-SQL-Defined Schemas

When you secure resource type DB, you control who can create an SQL schema for a non-SQL-defined schema. Until you secure resource type DB, any user can create an SQL schema for a non-SQL-defined schema.

How to Secure Non-SQL-defined Schemas

You secure non-SQL-defined schemas by securing the DB resource.

⚠️ Note: For more information, see Securing Databases (see page 93).
If you secure non-SQL-defined schemas externally, you must also include an entry in the SRTT with external security information for resource type NSCH.

### How to Grant USE Privilege on a Non-SQL-defined Schema

To allow a user to specify a non-SQL-defined schema when creating an SQL schema, you issue a GRANT statement on the non-SQL-defined schema specifying the USE privilege.

As a holder of SYSADMIN or DBADMIN privilege, you can specify WITH GRANT OPTION when you grant this privilege to allow the recipient to grant the same privilege to another user.

**More Information:**

- For more information, see GRANT Non-SQL Definition Privilege (see page 247).
- For more information, see REVOKE Non-SQL Definition Privilege (see page 267).
- For more information on creating SQL schemas, see the CA IDMS SQL Reference section.

### Granting SQL Access to Non-SQL-defined Databases

You allow SQL access to non-SQL-defined databases by creating an SQL schema for a non-SQL schema. This requires the CREATE privilege on the SQL schema and the USE privilege on the non-SQL-defined schema, as in this example:

```sql
grant create on schema qschtest
to dba;

grant use on nonsql schema v0001.nschtest
to dba;
```

The owner of the SQL schema for non-SQL schema has the authority to access the non-SQL-defined database using SQL. Other users require table access privileges.

### Securing Tables

#### About tables

When you secure resource type DB, you can control who can create and access a table-like object. Until you secure resource type DB, any user can create and access a table-like object.

#### How to Secure Tables

You secure tables by securing the DB resource.

**Note:** For more information, see Securing Databases (see page 93).
If you secure tables externally, you must also include an entry in the SRTT with external security information for resource type TABL.

**How to Grant Table Definition Privileges**

To allow a user to create a table-like object, you issue a GRANT statement on the table-like object, specifying the privilege or privileges and identifying the object. You can specify any combination of CREATE, ALTER, DISPLAY, and DROP privileges, or you can specify all definition privileges (DEFINE). You can also specify the REFERENCES privilege.

**Note:** REFERENCES privilege allows the user to create a constraint that names the table as the referenced table in the constraint.

As a holder of SYSADMIN or DBADMIN privilege or as owner of the table-like object, you can specify WITH GRANT OPTION when you grant definition privileges to allow the recipient to grant the same privileges to another user.

**Note:** For more information, see [GRANT SQL Definition Privileges (see page 253)](##) and see [REVOKE SQL Definition Privileges](##).

For more information on creating tables and constraints, see the *CA IDMS SQL Reference* section.

**How to Grant Table Access Privilege**

To allow a user to access a table-like object, you issue a GRANT statement on the table-like object, specifying the privilege or privileges. You can specify any combination of DELETE, INSERT, SELECT, and UPDATE privileges.

As a holder of SYSADMIN or DBADMIN privilege, or as owner of the table-like object, you can specify WITH GRANT OPTION when you grant access privileges to allow the recipient to grant the same privileges to another user.

**Note:** For more information, see the following sections:

- [GRANT Table Access Privileges (see page 257)](##)
- [REVOKE Table Access Privileges (see page 274)](##)
How to Grant All Table Privileges

You can grant all definition and access privileges on a table-like object with the GRANT ALL PRIVILEGES statement.

As a holder of SYSADMIN or DBADMIN privilege, or as owner of the table-like object, you can specify WITH GRANT OPTION when you grant all table privileges to allow the recipient to grant the same privileges to another user.

⚠️ Note: For more information, see the following sections:

- GRANT All Table Privileges (see page 242)
- REVOKE All Table Privileges (see page 263)

Securing Access to Table Definitions

You can allow limited access to table definitions by granting users privilege on SYSCA views.

SYSCA views restrict access to information in the SYSTEM tables to viewing definitions of only those tables on which the executing user holds SELECT privilege. SYSCA information on tables on which the executing user does not hold SELECT privilege. Therefore, a user who holds privilege on SYSCA views and not on SYSTEM tables must have the authority to retrieve data from a table in order to be able to view the definition of the table.

⚠️ Note: For more information on SYSCA views, see the CA IDMS SQL Reference section.

Securing Access Modules

About Access Modules

An access module is a set of compiled and optimized SQL statements. Certain characteristics of the access module are contained in its dictionary definition. The actual module to be loaded at runtime may be regenerated because of changed database characteristics and aspects of the access module definition.

When you secure resource type DB, you can control who can create and maintain an access module definition. Until you secure resource type DB, any user can create and maintain an access module definition.

How to Secure Access Modules

You secure access modules by securing the DB resource.
If you secure access modules externally, you must also include an entry in the SRTT with external security information for resource types DACC (access module definition) and SACC (loadable entity).

**How to Grant Access Module Definition Privilege**

To allow a user to create and maintain an access module definition, you issue a GRANT statement on the access module resource type, specifying the privilege or privileges and identifying the access module. You can specify any combination of CREATE, ALTER, DROP, and DISPLAY privileges, or you can specify all definition privileges (DEFINE).

As a holder of SYSADMIN or DBADMIN privilege, or as owner of the access module, you can specify WITH GRANT OPTION when you grant definition privileges to allow the recipient to grant the same privileges to another user.

**More Information:**

- For more information, see [GRANT SQL Definition Privileges (see page 253)](#).
- For more information, see [REVOKE SQL Definition Privileges (see page 271)](#).
- For more information on creating access modules, see the [CA IDMS SQL Reference section](#).

**How to Grant Access Module Execution Privilege**

Two security checks are involved in granting access module execution privilege. The categorization of access modules controls who can load an access module, it does not control who can execute the SQL statements in the access module. You have to think of an access module as any other load module. You need both the authority to load the module and the authority to perform the database access to the program.

As mentioned before, there are two levels of authority that must be granted to implement access module and DB security:

- A Category must be created and an access module added to the Category and execute authority granted. This controls the loading of the access module.

- Execute must be granted on the access module. This controls the database access for statements within the access module.

To allow a user to execute an individual access module, you issue a GRANT statement on the access module resource type, specifying the EXECUTE privilege and identifying the access module.

As holder of SYSADMIN or DBADMIN privilege, or as owner of the access module, you can specify WITH GRANT OPTION when you grant execution privilege to allow the recipient to grant the same privilege to another user.

**Note:** For more information, see the following sections:
Securing the Dictionaries and the User catalog

Activating Database Security

The dictionaries and the user catalog are CA IDMS databases. To secure these entities, you must activate database security for them.

The following discussion explains what to do when you secure dictionaries by defining occurrence overrides.

Note: You can secure all dictionaries by specifying security for all databases in an SRTT entry for the DB resource type.

For the purposes of discussion, the authorities that you give to allow access to a dictionary are the CA IDMS privileges you would grant if the dictionary is secured internally.

Securing the System Dictionary

To secure the system dictionary using occurrence overrides, you must secure the DB resource type for database name 'SYSTEM' and the names of the three segments that comprise the system dictionary.

In the following example, the first entry secures the name 'SYSTEM' which prevents access to the system dictionary through the database name defined for it at installation. This entry also prevents access to the SYSTEM segment. The next entries secure the SYSMSG segment, which contains messages, and the CATSYS segment, which is the catalog component of the dictionary.

```
#SECRTT  TYPE=OCCURRENCE, RESTYPE=DB, RESNAME='SYSTEM', SECBY=INTERNAL

#SECRTT  TYPE=OCCURRENCE, RESTYPE=DB, RESNAME='SYSMSG', SECBY=INTERNAL

#SECRTT  TYPE=OCCURRENCE, RESTYPE=DB, RESNAME='CATSYS', SECBY=INTERNAL
```

Securing the User catalog

To secure the user catalog with an occurrence override, specify the SYSUSER segment, as in this example:
If a database name has been defined for this segment, you must also include an entry specifying the database name.

**Securing Application Dictionaries**

If you activate database security with occurrence overrides, you must individually secure every segment in the application dictionary and every database name that includes a dictionary segment.

**Privileges for Secured Dictionaries**

After you have secured dictionaries and the user catalog, you grant privileges that permit appropriate access.

If security for the dictionary databases is *internal*, you grant CA IDMS privileges on the database resource types associated with the dictionary, including privileges on resources such as run units and areas that allow users to access the dictionary according to their needs.

If security for the dictionary databases is *external*, you define rules for each dictionary database and each of its associated database resources in the external system. You add SRTT entries with external class and resource name information for the dictionary database resources to be sent with security checks to the external security system.

For example, users who must execute the CA IDMS compilers such as the schema compiler require execute privilege on a Category containing compiler run unit resources. Users who must execute CA IDMS utilities require the appropriate privileges for area access.

**Granting Privileges on Run Units**

To grant blanket run unit access to an internally secured system dictionary, you first categorize all run units and then grant privilege on the category, as in this example:

```
create resource Category sysdict_general
    add rununit sysdict.*
;
grant execute
    on Category sysdict_general
    to general;
```

To categorize specific run units for CA IDMS compilers and tools that access the dictionary, you can specify as appropriate run units listed in the following installation source library members:

- SECASF
- SECCULP
- SECCMS
- SECID
- SECIT
For the purpose of using the CA IDMS Command Facility, there is no need to grant privileges on run units that access the SYSUSER segment.

**Example**

In this example, the system and application dictionaries have been secured. The first statement creates a Category of run units that access these dictionaries, and the second statement grants EXECUTE privilege on the Category:

```sql
create resource Category rununit_category
    add rununit appldict.idmsnwka.idmschem
    add rununit appldict.idmsnwka.idmsdddl
    add rununit appldict.idmsnwka.idmsubsc
    add rununit appldict.idmsnwkg.idmsrpts
    add rununit system.idmsnwka.idmsdddl
    add rununit system.idmsnwka.rhdcsgen
    add rununit system.idmsnwkg.idmsrpts
;
grant execute on Category rununit_category
to rununit_group
;
```

**Granting Privileges on Areas**

To allow execution of certain CA IDMS utilities against a secured database, you grant DBAREAD or DBAWRITE privilege on the area or areas to be accessed. The following table presents the installation names of the areas of the system dictionary and the user catalog:

<table>
<thead>
<tr>
<th>Database</th>
<th>Area (segment-name.area-name)</th>
</tr>
</thead>
<tbody>
<tr>
<td>System dictionary</td>
<td>SYSTEM.DDLDMU</td>
</tr>
<tr>
<td></td>
<td>SYSTEM.DDLDCRUN</td>
</tr>
<tr>
<td></td>
<td>SYSTEM.DDLDCLOG</td>
</tr>
<tr>
<td></td>
<td>SYSTEM.DDLDCSCR</td>
</tr>
<tr>
<td></td>
<td>SYSTEM.DDLDCLOD</td>
</tr>
<tr>
<td></td>
<td>SYSMSG.DDLDCMSG</td>
</tr>
<tr>
<td></td>
<td>CATSYS.DDLCAT</td>
</tr>
<tr>
<td></td>
<td>CATSYS.DDLCATX</td>
</tr>
<tr>
<td></td>
<td>CATSYS.DDLCATLOD</td>
</tr>
<tr>
<td>User catalog</td>
<td>SYSUSER.DDLSEC</td>
</tr>
</tbody>
</table>

**Granting Privileges on Non-SQL-defined Schemas**

When you secure the dictionaries and the user catalog, you control SQL access to these databases. You allow SQL access by creating SQL schemas for the non-SQL-defined schemas that describe these databases (IDMSNTWK for the dictionary and IDMSSECU for the user catalog) and granting table access privileges.
Similarly, security definitions for system and non-SQL-defined database resources are inaccessible through SQL unless you create an SQL schema for IDMSSECS, the non-SQL-defined schema for system resources security database.

**Granting Access to SYSTEM Tables**

When you secure a dictionary as a database, you secure tables associated with the SYSTEM schema in the catalog component of the dictionary. To allow access to the SYSTEM tables, you have these options:

- Grant access privileges on SYSTEM tables, individually or collectively.
- Grant access privileges on SYSCA views, which allow the executing user to view data in SYSTEM tables about only those tables on which the user holds SELECT privilege.

**Note:** For more information on SYSTEM tables and SYSCA views, see the CA IDMS SQL Reference section.

---

**Securing Application Dictionary Resources**

### Application Dictionary

An application dictionary contains definitions for application development objects such as dialogs and maps. An application dictionary may also contain non-SQL defined schemas and subschemas and, in its catalog component, SQL-defined entities.

### Securing the Dictionary As a Database

An application dictionary consists of one segment that is a non-SQL defined database and, if the SQL Option is installed, another segment (the catalog component) that is an SQL-defined database. A user can access the dictionary as a database, and, therefore, to secure the application dictionary, you should secure it as a database in CA IDMS centralized security.

**How You Do It**

To secure the application dictionary as a database, you secure the DB resource type or the occurrence of the DB resource that the dictionary represents. Then you categorize run units for the compilers and tools that access the dictionary and grant execution privilege on the Category. For more information, see [Securing the Dictionaries and the User catalog](http://wiki-dev.ca.com/display/IDMS/Securing+the+Dictionaries+and+the+User+catalog).

For more information, see the following topics:
Signon To the Dictionary

Using a Compiler or Tool

When a user invokes a compiler or tool, signon to the application dictionary is automatically initiated using the ID with which the requesting user is signed on to the system.

If the user of the compiler or tool is not signed on to the system, an actual system signon is attempted internally, and if it is successful, dictionary signon proceeds.

Thus, under centralized security, a user who is authorized to sign on to the system is authorized to access the dictionary that is current for the user session.

Current dictionary: You can enforce the specification of the current dictionary for a user’s session by including the DICTNAME attribute in the user profile with the OVERRIDE=NO parameter. This prevents the user from accessing a dictionary other than the one you specify in the DICTNAME attribute. For more information, see Securing User Profiles (see page 67).

Securing Secondary Signons

You can secure signon to a particular application dictionary by using DDDL to specify SECURITY FOR IDD SIGNON IS ON for the dictionary. In this situation, the user must be defined in the application dictionary with the ADD USER statement and authorized to sign on to the dictionary with the inclusion of IDD SIGNON IS ALLOWED in the USER statement.

This measure provides additional security only for IDD and does not affect security for other compilers that access the dictionary. Therefore, it is not a substitute for securing signon through centralized security.

Secondary Signon Processing

If a user issues an IDD signon statement that specifies the same ID as the user’s system signon ID, no password validation is done for signon to the dictionary. If a user issues an IDD signon statement that specifies a different ID from the user’s system signon ID, then the ID and password entered on the signon statement must match an ID and password defined in the dictionary with the ADD USER statement.

If the user is either not defined or not authorized, the secondary signon is rejected.

Compiler Security Within the Dictionary

Contents

- Checking Compiler Security (see page 124)
- Checking Registration Override Security (see page 124)
- Checking Verb Security (see page 125)
Checking Component Security (see page 126)

What is Compiler Security?

The compiler security described in this section is part of the IDD architecture and is not part of CA IDMS centralized security. Security checks access the definition of the user in the dictionary, not the user catalog.

When do Compilers Check Security?

The compilers perform security checking operations when any of the following is true of a DDL statement:

- The verb is SIGNON, VALIDATE, or GENERATE.
- The SET OPTIONS statement contains REGISTRATION OVERRIDE.
- The component type is SCHEMA.
- The component type is SUBSCHEMA.
- The statement is the first statement of the session.

Checks User’s Dictionary Description

In any of the preceding cases, the compiler determines whether the requested operation is secured within the dictionary.

If the operation is not secured, the compiler bypasses the security check and begins processing the statement.

If the operation is secured, the compiler checks the user’s description in the dictionary to determine whether the user is authorized to perform an operation.

If the user is authorized, the compiler processes the input statement; if not, the compiler issues an error message. All levels of security checking follow this procedure.

Types of Security

The compilers check four kinds of security:

- Compiler security
- Registration override security
- Verb security
- Component security

What Follows

Each kind of security is presented separately as follows; each topic includes the following information:
• When security is checked
• How security is turned on or off
• How the compiler determines who the issuing user is
• What constitutes an authorized user

Checking Compiler Security

Turning on Compiler Security

Compiler security is turned on or off through the IDD DDDL statement, SET OPTIONS FOR DICTIONARY SECURITY FOR IDMS IS ON/OFF. (Note that this IDD DDDL statement also turns verb security on or off: compiler security and verb security cannot be set independently.)

How the Compiler Checks the User

To determine who is issuing the statement, the compiler looks at the user name specified in the SIGNON statement. If the SIGNON statement is not issued or does not include the USER clause, the user name defaults to the following:

• The system signon ID of the user
• The ID in SET OPTIONS DEFAULT PREPARED BY/REVISED BY user-name

Definition of an Authorized User

An authorized user, for this function, is one whose description in the dictionary includes authority to use the compiler. Compiler authority is assigned through one of the following IDD DDDL USER statements (use MODIFY for existing user descriptions):

ADD USER NAME IS user-ID AUTHORITY FOR any verb user-ID IS ALL. IS IDMS. IS SCHEMA. IS SUBSCHEMA. assigns authority to use both compilers assigns authority to use both schema compiler only assigns authority to use the subschema compiler only

Checking Registration Override Security

When Compilers Check Registration Override

The schema and subschema compilers check registration override security when they encounter a SET OPTIONS statement containing a REGISTRATION OVERRIDE clause.

Turned on by REGISTRATION OVERRIDE
Unlike the other kinds of security, this one cannot be turned on or off; that is, the compiler always checks for an authorized user when it encounters a REGISTRATION OVERRIDE clause.

**How the Compilers Check the User**

To determine who is issuing the REGISTRATION OVERRIDE clause, the compiler looks at user names specified in the `user-specification` clause of the SET OPTIONS statement. If the SET OPTIONS statement does not include this clause, the user name defaults as described in the SET OPTIONS statement.

**Note:** For more information on the SET OPTIONS statement, see the *CA IDMS Database Administering section*.

**Description of an Authorized User**

An authorized user for the REGISTRATION OVERRIDE clause is one who has been defined in the dictionary and whose description includes all authorities. All authorities are assigned through the following IDD DDDL USER statement (use MODIFY for existing user descriptions):

```
ADD USER NAME IS user-id
    AUTHORITY IS ALL.
```

**Checking Verb Security**

**When Compilers Check Verb Security**

The schema and subschema compilers check verb security whenever a SCHEMA statement (schema compiler only) or SUBSCHEMA statement (subschema compiler only) is issued. Note that verb security is not checked for each component of a schema or subschema. Once a user passes security for a schema or subschema, all of its components are available to the user.

**Turning on Verb Security**

Verb security is turned on or off through the IDD DDDL statement, SET OPTIONS FOR DICTIONARY SECURITY FOR IDMS IS ON/OFF. (Note that this IDD DDDL statement also turns compiler security on or off: verb security and compiler security cannot be set independently.)

**How the Compilers Check the User**

To determine who is issuing the SCHEMA or SUBSCHEMA statement, the compiler looks at the following:

- The ID of user signed on to the dictionary
- The `user-specification` on the SCHEMA or SUBSCHEMA statement
- The `user-specification` on the SET OPTIONS statement
If any of these IDs is that of an authorized user, security is satisfied and the compiler processes the request.

**Description of an Authorized User**

An authorized user, for this function, is one who is defined in dictionary and whose description includes authority to issue the verb specified in the SCHEMA or SUBSCHEMA statement, *in conjunction with* the authority to use the compiler. Verb authority is assigned through IDD DDDL USER statements, such as those in the following examples:

- `ADD USER NAME IS KCO AUTHORITY FOR UPDATE IS IDMS.` assigns authority to use all verbs in each DDL compiler
- `ADD USER NAME IS GKD AUTHORITY FOR MODIFY IS IDMS.` assigns authority to use MODIFY, DISPLAY, and PUNCH in each DDL compiler
- `ADD USER NAME IS TWG AUTHORITY FOR DELETE IS SCHEMA.` assigns authority to use DELETE, DISPLAY, and PUNCH in the schema compiler only

**Implicit Subschema Updates Allowed**

While schema authority only allows the user to access the schema compiler, any subschema updates resulting from authorized schema updates are allowed (for example, deleting a set from the schema causes the set to be deleted from the schema's subschemas).

⚠️ **Note:** For more information on assigning verb authority, see the *CA IDMS IDD DDDL Reference section.*

## Checking Component Security

### When Compilers Check Component Security

The schema compiler checks the security of a specific schema whenever a SCHEMA statement (other than ADD SCHEMA) is issued for that schema; the subschema compiler checks security of a specific subschema whenever a SUBSCHEMA statement (other than ADD SUBSCHEMA) is issued for that subschema. Note that this security is not checked for each component of a schema or subschema; once a user passes security for a schema or a subschema, all of its components are available to the user. Component security applies to every existing schema and subschema, regardless of whether compiler security is on.

### Security Maintained Through PUBLIC ACCESS Clause

Security for a specific schema or subschema is set through the PUBLIC ACCESS clause of the SCHEMA or SUBSCHEMA statement. A schema or subschema is said to be unsecured if PUBLIC ACCESS IS ALLOWED FOR ALL is in effect; any other public access specification places some level of security on the schema or subschema. The following examples show how component security is set:
Description of an Authorized User

An authorized user for a specific schema or subschema is one who is defined in the dictionary and whose association with the schema or subschema includes the verb used in the SCHEMA or SUBSCHEMA statement being processed. This authority is assigned through the REGISTERED FOR subclause (in the user-options-specification) of the USER clause in a previously issued SCHEMA or SUBSCHEMA statement, as illustrated in the following examples:

ADD SUBSCHEMA NAME IS EMPSS01
   USER NAME IS KCO
   REGISTERED FOR ALL.

ADD SUBSCHEMA NAME IS EMPSS02
   USER NAME IS GKD
   REGISTERED FOR PUBLIC ACCESS.

ADD SCHEMA NAME IS EMPSCHM
   USER NAME IS TWG
   REGISTERED FOR DISPLAY.

Note: For more information on PUBLIC ACCESS and USER clauses, see the SCHEMA and SUBSCHEMA statement documentation in the CA IDMS Database Administering section.

Syntax for Assembler Macros

This section explains the syntax, parameters, command codes, examples, and usage of the following Macros:

- CTABGEN (see page 128)
- UTABGEN (see page 147)
- GTABGEN (see page 151)
- SECHECK (see page 153)
- SECRRT (see page 160)
- SECSGOF (see page 170)
- SECSGON (see page 173)
CTABGEN

CTABGEN assigns activity numbers to DCMT commands.

- **Syntax** (see page 128)
- **Parameters** (see page 128)
- **Usage** (see page 129)
- **DCMT Command Codes** (see page 131)

### Syntax

```
#CTABGEN ( command-security-specification ) .LOGIN= YES NO
```

**Expansion of Command-Security-Specification**

```
security-label, activity-number
command-code, security-label, activity-number
```

### Parameters

- **command-security-specification**
  Specifies the information for assigning activity numbers to DCMT commands.
  Expanded syntax for `command-security-specification` appears immediately following the command syntax.

- **LOGIN**
  Specifies whether DC/UCF writes input DCMT commands to the log file.
  If you do not specify the LOGIN parameter, the effect is the same as specifying LOGIN=NO.

- **YES**
  Specifies that all input DCMT commands are written to the log file.

- **NO**
  Specifies that most input DCMT commands are not written to the log.
  Some input DCMT commands are always written to the log because they provide important information on the activity of the command. For example, DCMT VARY MEMORY commands are always logged. The LOGIN option does not stop the system from logging these input DCMT commands.

- **security-label,activity-number**
  Associates a DCMT security label with an activity number.
  *Security-label* specifies a label with which you can associate one or more DCMT commands.
Security-label must be one alphabetic character (A through Z). You can define at most 26 security labels in the #CTABGEN macro. Activity-number specifies the activity number you are associating with security-label. Activity-number must be a numeric value in the range 1 through 256.

- **command-code**
  Specifies one of the following:
  - The 4-character command code that identifies a predefined group of DCMT commands. For example, N001 identifies the DCMT SHUTDOWN command group.
  - The 7-character command code that identifies a specific DCMT command option. For example, N001001 identifies the DCMT SHUTDOWN IMMEDIATE command.

You can specify any number of DCMT command codes in the #CTABGEN macro.

⚠️ **Note**: For a list of valid DCMT command codes, see [DCMT Command Codes](#).

- **security-label**
  Specifies the DCMT security label that you are associating with command-code.

- **activity-number**
  Specifies the DCMT activity number that you are associating with command-code.

## Usage

### Coding Considerations

- All lines except the first one must start in column 16.
- All lines except the last one must have a non-blank character in column 72.

### General

When you use the #CTABGEN macro, you can assign a DCMT activity number to a DCMT command or to a group of commands:

- You can associate a specific **DCMT activity number** (1 through 256) with the DCMT command.
- You can associate a site-defined **security label** (A through Z) with the DCMT command. Use of security labels makes it easier to maintain security definitions when several commands and/or command groups are assigned the same DCMT activity number. You define a security label in the #CTABGEN macro itself. You need only change the security label definition in the #CTABGEN macro to modify the security for all associated DCMT commands.

### Defining Labels
You make security label assignments in a parenthesized list. Use a comma to separate specifications and parenthesized lists. For example:

```
- - - - - - 1 - - - - - 2 - - - - - 3 - - - - - 4 - - - - - 5 - - - - - 6 - - - - - 7 -
#CTABGEN (A, 3, B, 10),
    (N001, A, N002, B, N003, 6)
```

You must define all security labels before beginning the list of DCMT security assignments.

**Specifying Command Groups and Command Codes**

If a 4-character command code is given for a group of commands and a 7-character command code is given for a particular command within that group, then the DCMT activity number for the particular command overrides the security for the group.

**No Effect of LOGIN on Response Messages**

Response messages issued for DCMT commands are not affected by the LOGIN specification. Response messages are written to the system log if LOG is given as a destination in the message definition.

**z/VSE Sites**

At z/VSE sites, the LOGIN parameter, if coded, must be coded following all other #CTABGEN parameters.

**Generating the #CTABGEN Macro**

The source file that contains the #CTABGEN macro can contain only one macro. The resulting object must be link edited as a stand-alone module IDMSCTAB.

To assemble and link edit the #CTABGEN macro, use the JCL or commands appropriate for your operating system.

⚠️ **Note:** For more information, see Appendix A: Security Macro JCL. (see page 310)

**Examples**

**Example 1**

Assign DCMT activity number 3 to DCMT SHUTDOWN (code N001) and ABORT (code N002) by means of security label A:

```
- - - - - - 1 - - - - - 2 - - - - - 3 - - - - - 4 - - - - - 5 - - - - - 6 - - - - - 7 -
#CTABGEN (A, 3),
    (N001, A, N002, A)
```

**Example 2**
Assign different discrete security to several DCMT commands:

```
#CTABGEN (A,3,B,10),                X
(N001,A,N002,A,N033,A),             X
(N011,B,N025,B),                    X
(N025007,50)
```

This #CTABGEN macro associates three different DCMT activity numbers (3, 10, and 50) with DCMT commands:

- **Number 3 is assigned indirectly**, by means of security label A, to the following:
  - SHUTDOWN (N001)
  - ABORT (N002)
  - VARY MEMORY (N033)
- **Number 10 is assigned indirectly**, by means of security label B, to the following:
  - VARY DATABASE PROGRAM (N011)
  - VARY PROGRAM (N025)
- **Number 50 is assigned directly** to one DCMT command: DCMT VARY PROGRAM STORAGE PROTECT ON (N025007). To use this DCMT command, a user would require DCMT activity number 50. To use any other DCMT VARY PROGRAM command, the user would require DCMT activity number 10.

### DCMT Command Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>DCMT Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>N001</td>
<td>SHUTDOWN</td>
</tr>
<tr>
<td>N001000</td>
<td>SHUTDOWN</td>
</tr>
<tr>
<td>N001001</td>
<td>SHUTDOWN IMMEDIATE</td>
</tr>
<tr>
<td>N002</td>
<td>ABORT</td>
</tr>
<tr>
<td>N002000</td>
<td>ABORT</td>
</tr>
<tr>
<td>N002001</td>
<td>ABORT DUMP</td>
</tr>
<tr>
<td>N003</td>
<td>DISPLAY DATABASE</td>
</tr>
<tr>
<td>N003000</td>
<td>DISPLAY DATABASE</td>
</tr>
<tr>
<td>N004</td>
<td>DISPLAY TRANSACTIONS</td>
</tr>
<tr>
<td>N004000</td>
<td>DISPLAY TRANSACTIONS</td>
</tr>
<tr>
<td>N004001</td>
<td>DISPLAY TRANSACTION run-unit-id</td>
</tr>
<tr>
<td>N005</td>
<td>DISPLAY AREAS</td>
</tr>
<tr>
<td>N005000</td>
<td>DISPLAY AREAS</td>
</tr>
<tr>
<td>N005001</td>
<td>DISPLAY AREA area-name</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>N005002</td>
<td>DISPLAY AREA area-name FILE</td>
</tr>
<tr>
<td>N005003</td>
<td>DISPLAY AREA area-name BUFFER</td>
</tr>
<tr>
<td>N005004</td>
<td>DISPLAY AREA area-name ALL</td>
</tr>
<tr>
<td>N005128</td>
<td>DISPLAY AREA area-name LOC</td>
</tr>
<tr>
<td>N006</td>
<td>DISPLAY BUFFERS</td>
</tr>
<tr>
<td>N006000</td>
<td>DISPLAY BUFFERS</td>
</tr>
<tr>
<td>N006001</td>
<td>DISPLAY BUFFER buffer-name AREA</td>
</tr>
<tr>
<td>N006002</td>
<td>DISPLAY BUFFER buffer-name FILE</td>
</tr>
<tr>
<td>N006003</td>
<td>DISPLAY BUFFER buffer-name</td>
</tr>
<tr>
<td>N006004</td>
<td>DISPLAY BUFFER buffer-name ALL</td>
</tr>
<tr>
<td>N006128</td>
<td>DISPLAY BUFFER buffer-name LOC</td>
</tr>
<tr>
<td>N006130</td>
<td>DISPLAY BUFFER buffer-name FILE BUFNO</td>
</tr>
<tr>
<td>N007</td>
<td>DISPLAY CENTRAL VERSION</td>
</tr>
<tr>
<td>N007000</td>
<td>DISPLAY CENTRAL VERSION</td>
</tr>
<tr>
<td>N008</td>
<td>DISPLAY DATABASE PROGRAMS</td>
</tr>
<tr>
<td>N008000</td>
<td>DISPLAY DATABASE PROGRAMS</td>
</tr>
<tr>
<td>N008001</td>
<td>DISPLAY DATABASE PROGRAM program-name</td>
</tr>
<tr>
<td>N009</td>
<td>VARY AREA</td>
</tr>
<tr>
<td>N009000</td>
<td>ACTIVE</td>
</tr>
<tr>
<td>N009001</td>
<td>RETRIEVAL &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N009002</td>
<td>UPDATE (ONLINE) &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N009003</td>
<td>OFFLINE &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N009004</td>
<td>TRANSIENT RETRIEVAL &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N009005</td>
<td>UPDATE (ONLINE) LOCKED &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N009006</td>
<td>PREFETCH ON</td>
</tr>
<tr>
<td>N009007</td>
<td>PREFETCH OFF</td>
</tr>
<tr>
<td>N009008</td>
<td>QUIESCE WAIT</td>
</tr>
<tr>
<td>N009011</td>
<td>OPEN UPDATE</td>
</tr>
<tr>
<td>N009012</td>
<td>OPEN</td>
</tr>
<tr>
<td>N009013</td>
<td>CLOSE</td>
</tr>
<tr>
<td>N009014</td>
<td>PURGE</td>
</tr>
<tr>
<td>N009015</td>
<td>QUIESCE &lt;NOWAIT&gt;</td>
</tr>
<tr>
<td>N009016</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>N009017</td>
<td>SHARED CACHE cache-name</td>
</tr>
<tr>
<td>N009018</td>
<td>SHARED CACHE NO</td>
</tr>
<tr>
<td>N009020</td>
<td>DATA SHARING ON</td>
</tr>
<tr>
<td>N009021</td>
<td>DATA SHARING OFF</td>
</tr>
<tr>
<td>N009022</td>
<td>ALLOCATE</td>
</tr>
<tr>
<td>N010</td>
<td>VARY BUFFER</td>
</tr>
<tr>
<td>N010000</td>
<td>PAGES</td>
</tr>
<tr>
<td>N010001</td>
<td>OPEN</td>
</tr>
<tr>
<td>N010002</td>
<td>CLOSE</td>
</tr>
<tr>
<td>N010003</td>
<td>MAXIMUM PAGES</td>
</tr>
<tr>
<td>N010004</td>
<td>INITIAL PAGES</td>
</tr>
<tr>
<td>N010005</td>
<td>ADDITIONAL PAGES</td>
</tr>
<tr>
<td>N010006</td>
<td>OPSYS</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>N01007</td>
<td>DC</td>
</tr>
<tr>
<td>N01010</td>
<td>PREFETCH ON</td>
</tr>
<tr>
<td>N01011</td>
<td>PREFETCH OFF</td>
</tr>
<tr>
<td>N01012</td>
<td>PREFETCH nnn</td>
</tr>
<tr>
<td>N011</td>
<td>VARY DATABASE PROGRAM</td>
</tr>
<tr>
<td>N01100</td>
<td>PRIORITY</td>
</tr>
<tr>
<td>N01101</td>
<td>LOCKS</td>
</tr>
<tr>
<td>N01102</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N01103</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N01104</td>
<td>STORAGE/LOCK/CALL/DBIO LIMIT</td>
</tr>
<tr>
<td>N012</td>
<td>VARY CENTRAL VERSION</td>
</tr>
<tr>
<td>N01200</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N01201</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N013</td>
<td>VARY TRANSACTION</td>
</tr>
<tr>
<td>N01300</td>
<td>ABORT</td>
</tr>
<tr>
<td>N01301</td>
<td>PRIORITY</td>
</tr>
<tr>
<td>N015</td>
<td>DISPLAY TASKS</td>
</tr>
<tr>
<td>N015001</td>
<td>DISPLAY TASKS</td>
</tr>
<tr>
<td>N015002</td>
<td>DISPLAY TASK task-name</td>
</tr>
<tr>
<td>N016</td>
<td>DISPLAY PROGRAMS</td>
</tr>
<tr>
<td>N016001</td>
<td>DISPLAY PROGRAMS</td>
</tr>
<tr>
<td>N016002</td>
<td>DISPLAY PROGRAM program-name</td>
</tr>
<tr>
<td>N017</td>
<td>DISPLAY TIME</td>
</tr>
<tr>
<td>N017001</td>
<td>DISPLAY TIME</td>
</tr>
<tr>
<td>N017002</td>
<td>DISPLAY TIME TASKS</td>
</tr>
<tr>
<td>N018</td>
<td>DISPLAY QUEUES</td>
</tr>
<tr>
<td>N018001</td>
<td>DISPLAY QUEUES</td>
</tr>
<tr>
<td>N018002</td>
<td>DISPLAY QUEUE queue-name</td>
</tr>
<tr>
<td>N019</td>
<td>DISPLAY DESTINATIONS</td>
</tr>
<tr>
<td>N019001</td>
<td>DISPLAY DESTINATIONS</td>
</tr>
<tr>
<td>N019002</td>
<td>DISPLAY DESTINATION destination-id</td>
</tr>
<tr>
<td>N020</td>
<td>DISPLAY LTERMINALS</td>
</tr>
<tr>
<td>N020001</td>
<td>DISPLAY LTERMINALS</td>
</tr>
<tr>
<td>N020002</td>
<td>DISPLAY LTERMINAL logical-terminal-id</td>
</tr>
<tr>
<td>N020003</td>
<td>DISPLAY LTERMINAL RESOURCES</td>
</tr>
<tr>
<td>N021</td>
<td>DISPLAY PTERMINALS/SNA PTERMINALS</td>
</tr>
<tr>
<td>N021001</td>
<td>DISPLAY PTERMINALS/SNA PTERMINALS</td>
</tr>
<tr>
<td>N021002</td>
<td>DISPLAY PTERMINAL physical-terminal-id</td>
</tr>
<tr>
<td>N021004</td>
<td>DISPLAY SNA PTERMINAL physical-terminal-id</td>
</tr>
<tr>
<td>N021004</td>
<td>DISPLAY SNA PTERMINAL physical-terminal-id</td>
</tr>
<tr>
<td>N022</td>
<td>DISPLAY MEMORY</td>
</tr>
<tr>
<td>N022001</td>
<td>hex-address</td>
</tr>
<tr>
<td>N022002</td>
<td>CSA</td>
</tr>
<tr>
<td>N022003</td>
<td>TCA</td>
</tr>
<tr>
<td>N022004</td>
<td>RCA</td>
</tr>
<tr>
<td>N022005</td>
<td>SCT</td>
</tr>
<tr>
<td>N022006</td>
<td>OPT</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>N022007</td>
<td>MAP</td>
</tr>
<tr>
<td>N022008</td>
<td>TDT</td>
</tr>
<tr>
<td>N022009</td>
<td>TDE</td>
</tr>
<tr>
<td>N022010</td>
<td>PDT</td>
</tr>
<tr>
<td>N022011</td>
<td>PDE</td>
</tr>
<tr>
<td>N022012</td>
<td>QDT</td>
</tr>
<tr>
<td>N022013</td>
<td>QDE</td>
</tr>
<tr>
<td>N022014</td>
<td>DDT</td>
</tr>
<tr>
<td>N022015</td>
<td>DDE</td>
</tr>
<tr>
<td>N022016</td>
<td>LTT</td>
</tr>
<tr>
<td>N022017</td>
<td>LTE</td>
</tr>
<tr>
<td>N022018</td>
<td>PLE</td>
</tr>
<tr>
<td>N022019</td>
<td>PTE</td>
</tr>
<tr>
<td>N022020</td>
<td>ID</td>
</tr>
<tr>
<td>N022021</td>
<td>PROGRAM</td>
</tr>
<tr>
<td>N022022</td>
<td>STR</td>
</tr>
<tr>
<td>N022023</td>
<td>MOD/EP</td>
</tr>
<tr>
<td>N022024</td>
<td>NUCLEUS</td>
</tr>
<tr>
<td>N022025</td>
<td>ESE</td>
</tr>
<tr>
<td>N022026</td>
<td>ERES</td>
</tr>
<tr>
<td>N022027</td>
<td>ACTIVE ERES</td>
</tr>
<tr>
<td>N022028</td>
<td>SVC</td>
</tr>
<tr>
<td>N022029</td>
<td>CCE</td>
</tr>
<tr>
<td>N022030</td>
<td>DMCL</td>
</tr>
<tr>
<td>N022031</td>
<td>BCR/BUFFER</td>
</tr>
<tr>
<td>N022032</td>
<td>JCB/JOURNAL</td>
</tr>
<tr>
<td>N022033</td>
<td>SEGMENT</td>
</tr>
<tr>
<td>N022034</td>
<td>DPR/AREA</td>
</tr>
<tr>
<td>N022035</td>
<td>FCB/FILE</td>
</tr>
<tr>
<td>N023</td>
<td>DISPLAY ACTIVE</td>
</tr>
<tr>
<td>N023001</td>
<td>TASKS</td>
</tr>
<tr>
<td>N023002</td>
<td>PROGRAMS</td>
</tr>
<tr>
<td>N023010</td>
<td>TASKS WAITING</td>
</tr>
<tr>
<td>N023013</td>
<td>STORAGE</td>
</tr>
<tr>
<td>N023020</td>
<td>TASKS HOLDING</td>
</tr>
<tr>
<td>N023027</td>
<td>REENTRANT PROGRAMS</td>
</tr>
<tr>
<td>N023039</td>
<td>XA PROGRAMS</td>
</tr>
<tr>
<td>N023040</td>
<td>XA REENTRANT PROGRAMS</td>
</tr>
<tr>
<td>N024</td>
<td>VARY TASK</td>
</tr>
<tr>
<td>N024001</td>
<td>ENABLE</td>
</tr>
<tr>
<td>N024002</td>
<td>DISABLE</td>
</tr>
<tr>
<td>N024003</td>
<td>SECURITY</td>
</tr>
<tr>
<td>N024004</td>
<td>PRIORITY</td>
</tr>
<tr>
<td>N024005</td>
<td>STALL</td>
</tr>
<tr>
<td>N024006</td>
<td>PROGRAM</td>
</tr>
<tr>
<td>N024007</td>
<td>RESOURCE INTERVAL timeout-interval/OFF</td>
</tr>
<tr>
<td>N024008</td>
<td>RESOURCE INTERVAL SYSTEM</td>
</tr>
<tr>
<td>N024009</td>
<td>RESOURCE PROGRAM</td>
</tr>
<tr>
<td>N024010</td>
<td>STALL OFF</td>
</tr>
<tr>
<td>N024011</td>
<td>STALL SYSTEM</td>
</tr>
<tr>
<td>N024012</td>
<td>SAVE</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>N024013</td>
<td>NOSAVE</td>
</tr>
<tr>
<td>N024014</td>
<td>LOCATION ANY</td>
</tr>
<tr>
<td>N024015</td>
<td>LOCATION BELOW</td>
</tr>
<tr>
<td>N024016</td>
<td>MAXIMUM CONCURRENT OFF</td>
</tr>
<tr>
<td>N024017</td>
<td>MAXIMUM CONCURRENT <em>task-count</em></td>
</tr>
<tr>
<td>N024018</td>
<td>STORAGE/LOCK/CALL/DBIO LIMIT</td>
</tr>
<tr>
<td>N024019</td>
<td>EXTERNAL WAIT <em>nnn</em></td>
</tr>
<tr>
<td>N024020</td>
<td>QUIESE WAIT <em>nnn</em></td>
</tr>
<tr>
<td>N024021</td>
<td>TRANSACTION SHARING ON/OFF</td>
</tr>
<tr>
<td>N024022</td>
<td>ON COMMIT ..</td>
</tr>
<tr>
<td>N024023</td>
<td>ON ROLLBACK ..</td>
</tr>
<tr>
<td>N024024</td>
<td>SNAP snap-options</td>
</tr>
<tr>
<td>N025</td>
<td>VARY PROGRAM</td>
</tr>
<tr>
<td>N025001</td>
<td>ENABLE</td>
</tr>
<tr>
<td>N025002</td>
<td>DISABLE</td>
</tr>
<tr>
<td>N025003</td>
<td>SECURITY</td>
</tr>
<tr>
<td>N025004</td>
<td>NEW COPY</td>
</tr>
<tr>
<td>N025005</td>
<td>PROGRAM CHECK THRESHOLD</td>
</tr>
<tr>
<td>N025006</td>
<td>DUMP THRESHOLD</td>
</tr>
<tr>
<td>N025007</td>
<td>STORAGE PROTECT ON</td>
</tr>
<tr>
<td>N025008</td>
<td>STORAGE PROTECT OFF</td>
</tr>
<tr>
<td>N025009</td>
<td>NEW COPY QUIESCE</td>
</tr>
<tr>
<td>N025010</td>
<td>NEW COPY IMMEDIATE</td>
</tr>
<tr>
<td>N025011</td>
<td>ADSO STATISTICS ON</td>
</tr>
<tr>
<td>N025012</td>
<td>ADSO STATISTICS OFF</td>
</tr>
<tr>
<td>N025013</td>
<td>MULTIPLE ENCLAVE ON</td>
</tr>
<tr>
<td>N025014</td>
<td>MULTIPLE ENCLAVE OFF</td>
</tr>
<tr>
<td>N025015</td>
<td>DEFINE &lt;keyword&gt;</td>
</tr>
<tr>
<td>N025016</td>
<td>DEFINE LANGUAGE</td>
</tr>
<tr>
<td>N025017</td>
<td>DEFINE ISA SIZE</td>
</tr>
<tr>
<td>N025018</td>
<td>DEFINE TYPE</td>
</tr>
<tr>
<td>N025019</td>
<td>DEFINE MPMODE</td>
</tr>
<tr>
<td>N025020</td>
<td>SNAP snap-options</td>
</tr>
<tr>
<td>N026</td>
<td>VARY TIME</td>
</tr>
<tr>
<td>N026001</td>
<td>STALL</td>
</tr>
<tr>
<td>N026002</td>
<td>RUNAWAY</td>
</tr>
<tr>
<td>N026003</td>
<td>TIMER</td>
</tr>
<tr>
<td>N026004</td>
<td>RESOURCE INTERVAL</td>
</tr>
<tr>
<td>N026005</td>
<td>RESOURCE PROGRAM</td>
</tr>
<tr>
<td>N026006</td>
<td>RECOVERY WAIT <em>nnn</em></td>
</tr>
<tr>
<td>N026007</td>
<td>QUIESCE WAIT <em>nnn</em></td>
</tr>
<tr>
<td>N027</td>
<td>VARY STORAGE POOL</td>
</tr>
<tr>
<td>N027001</td>
<td>VARY STORAGE POOL CUSHION</td>
</tr>
<tr>
<td>N027002</td>
<td>VARY STORAGE POOL RELOCATABLE THRESHOLD</td>
</tr>
<tr>
<td>N028</td>
<td>VARY LTERM</td>
</tr>
<tr>
<td>N028001</td>
<td>DESTINATION ONLINE</td>
</tr>
<tr>
<td>N028002</td>
<td>DESTINATION OFFLINE</td>
</tr>
<tr>
<td>N028003</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N028004</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N028005</td>
<td>TO</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>N02806</td>
<td>DISCONNECT</td>
</tr>
<tr>
<td>N02808</td>
<td>USERTRACE OFF</td>
</tr>
<tr>
<td>N02809</td>
<td>USERTRACE ON SAVE</td>
</tr>
<tr>
<td>N02810</td>
<td>USERTRACE ON WRAP</td>
</tr>
<tr>
<td>N02811</td>
<td>RESOURCES DELETE</td>
</tr>
<tr>
<td>N02812</td>
<td>COMMAND</td>
</tr>
<tr>
<td>N02813</td>
<td>TERMINAL</td>
</tr>
<tr>
<td>N02814</td>
<td>TCP/IP TRACE</td>
</tr>
<tr>
<td>N029</td>
<td>VARY PTERM</td>
</tr>
<tr>
<td>N02900</td>
<td>DEFAULT PRINT CLASS</td>
</tr>
<tr>
<td>N02901</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N02902</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N02903</td>
<td>TRACE OFF</td>
</tr>
<tr>
<td>N02904</td>
<td>TRACE (hh)</td>
</tr>
<tr>
<td>N02905</td>
<td>QUIESCE</td>
</tr>
<tr>
<td>N02906</td>
<td>CONNECT</td>
</tr>
<tr>
<td>N02907</td>
<td>DISCONNECT</td>
</tr>
<tr>
<td>N02908</td>
<td>ONLINE (&lt;\text{telephone-number}&gt;)</td>
</tr>
<tr>
<td>N02909</td>
<td>CONNECT (&lt;\text{telephone-number}&gt;)</td>
</tr>
<tr>
<td>N02910</td>
<td>(tcp/ip)-parameters</td>
</tr>
<tr>
<td>N030</td>
<td>VARY LINE</td>
</tr>
<tr>
<td>N03001</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N03002</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N03003</td>
<td>CONTROL UNIT ONLINE</td>
</tr>
<tr>
<td>N03004</td>
<td>CONTROL UNIT OFFLINE</td>
</tr>
<tr>
<td>N03005</td>
<td>QUIESCE</td>
</tr>
<tr>
<td>N03006</td>
<td>CONNECT</td>
</tr>
<tr>
<td>N03007</td>
<td>DISCONNECT</td>
</tr>
<tr>
<td>N03008</td>
<td>RLN ONLINE</td>
</tr>
<tr>
<td>N03009</td>
<td>RLN OFFLINE</td>
</tr>
<tr>
<td>N03010</td>
<td>RLN QUIESCE</td>
</tr>
<tr>
<td>N03011</td>
<td>RLN CONNECT</td>
</tr>
<tr>
<td>N03012</td>
<td>RLN DISCONNECT</td>
</tr>
<tr>
<td>N03013</td>
<td>RLN CONTROL UNIT ONLINE</td>
</tr>
<tr>
<td>N03014</td>
<td>RLN CONTROL UNIT OFFLINE</td>
</tr>
<tr>
<td>N03015</td>
<td>MASTER</td>
</tr>
<tr>
<td>N03016</td>
<td>SLAVE</td>
</tr>
<tr>
<td>N031</td>
<td>VARY QUEUE</td>
</tr>
<tr>
<td>N03101</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N03102</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N03103</td>
<td>THRESHOLD COUNT</td>
</tr>
<tr>
<td>N03104</td>
<td>MAX RECORDS</td>
</tr>
<tr>
<td>N03105</td>
<td>TASK CODE</td>
</tr>
<tr>
<td>N03106</td>
<td>DELETE</td>
</tr>
<tr>
<td>N032</td>
<td>VARY DESTINATION</td>
</tr>
<tr>
<td>N032001</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N032002</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>N032003</td>
<td>ADD TERMINAL</td>
</tr>
<tr>
<td>N032004</td>
<td>ADD OPERATOR</td>
</tr>
<tr>
<td>N032005</td>
<td>DELETE TERMINAL</td>
</tr>
<tr>
<td>N032006</td>
<td>DELETE OPERATOR</td>
</tr>
<tr>
<td>N033</td>
<td>VARY MEMORY</td>
</tr>
<tr>
<td>N033001</td>
<td>X 'hex-literal'</td>
</tr>
<tr>
<td>N033002</td>
<td>C 'character-literal'</td>
</tr>
<tr>
<td>N033003</td>
<td>PROGRAM</td>
</tr>
<tr>
<td>N034</td>
<td>VARY ACTIVE TASK</td>
</tr>
<tr>
<td>N034000</td>
<td>TERMINATE USERID user-id DUMP</td>
</tr>
<tr>
<td>N034001</td>
<td>MAX TASK</td>
</tr>
<tr>
<td>N034002</td>
<td>TERMINATE TASKID</td>
</tr>
<tr>
<td>N034003</td>
<td>TERMINATE TERMID</td>
</tr>
<tr>
<td>N034004</td>
<td>PRIORITY TASKID</td>
</tr>
<tr>
<td>N034005</td>
<td>PRIORITY TERMID</td>
</tr>
<tr>
<td>N034006</td>
<td>TERMINATE USERID</td>
</tr>
<tr>
<td>N034007</td>
<td>STORAGE/LOCK/CALL/DBIO LIMIT</td>
</tr>
<tr>
<td>N034008</td>
<td>TERMINATE TASKID task-id DUMP</td>
</tr>
<tr>
<td>N034009</td>
<td>TERMINATE TERMID logical-terminal-id DUMP</td>
</tr>
<tr>
<td>N035</td>
<td>HELP</td>
</tr>
<tr>
<td>N035000</td>
<td>HELP</td>
</tr>
<tr>
<td>N035001</td>
<td>TASKS</td>
</tr>
<tr>
<td>N035002</td>
<td>PROGRAMS</td>
</tr>
<tr>
<td>N035003</td>
<td>TIME</td>
</tr>
<tr>
<td>N035004</td>
<td>QUEUES</td>
</tr>
<tr>
<td>N035005</td>
<td>DESTINATIONS</td>
</tr>
<tr>
<td>N035006</td>
<td>TERMINALS</td>
</tr>
<tr>
<td>N035008</td>
<td>STORAGE</td>
</tr>
<tr>
<td>N035009</td>
<td>DATABASE</td>
</tr>
<tr>
<td>N035010</td>
<td>PRINTERS</td>
</tr>
<tr>
<td>N035011</td>
<td>MESSAGE</td>
</tr>
<tr>
<td>N035012</td>
<td>REPORTS</td>
</tr>
<tr>
<td>N035013</td>
<td>STATISTICS</td>
</tr>
<tr>
<td>N035014</td>
<td>LOADLIBS, LOADLISTS, and DICTIONARIES</td>
</tr>
<tr>
<td>N035015</td>
<td>SNAP</td>
</tr>
<tr>
<td>N035016</td>
<td>ADSO</td>
</tr>
<tr>
<td>N035017</td>
<td>LIMITS</td>
</tr>
<tr>
<td>N035018</td>
<td>LU</td>
</tr>
<tr>
<td>N035019</td>
<td>SNA</td>
</tr>
<tr>
<td>N035020</td>
<td>XA</td>
</tr>
<tr>
<td>N035021</td>
<td>JOURNALS</td>
</tr>
<tr>
<td>N035022</td>
<td>AREAS</td>
</tr>
<tr>
<td>N035023</td>
<td>MULTITASK</td>
</tr>
<tr>
<td>N035024</td>
<td>NUCLEUS</td>
</tr>
<tr>
<td>N035025</td>
<td>RUN UNITS</td>
</tr>
<tr>
<td>N035026</td>
<td>BUFFERS</td>
</tr>
<tr>
<td>N035027</td>
<td>SHUTDOWN and ABORT</td>
</tr>
<tr>
<td>N035028</td>
<td>DYNAMIC</td>
</tr>
<tr>
<td>N035029</td>
<td>FILES</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>N035030</td>
<td>SEGMENTS</td>
</tr>
<tr>
<td>N035031</td>
<td>DEADLOCKS</td>
</tr>
<tr>
<td>N035032</td>
<td>TRANSACTIONS</td>
</tr>
<tr>
<td>N035033</td>
<td>DBTABLE</td>
</tr>
<tr>
<td>N035034</td>
<td>DMCL</td>
</tr>
<tr>
<td>N035035</td>
<td>LOCKS</td>
</tr>
<tr>
<td>N035036</td>
<td>LOG</td>
</tr>
<tr>
<td>N035037</td>
<td>NODE</td>
</tr>
<tr>
<td>N035038</td>
<td>SYSGEN</td>
</tr>
<tr>
<td>N035039</td>
<td>DDS</td>
</tr>
<tr>
<td>N035040</td>
<td>DBGROUP</td>
</tr>
<tr>
<td>N035041</td>
<td>SHARED CACHE</td>
</tr>
<tr>
<td>N035042</td>
<td>RESOURCE</td>
</tr>
<tr>
<td>N035043</td>
<td>DATA SHARING</td>
</tr>
<tr>
<td>N035044</td>
<td>ID</td>
</tr>
<tr>
<td>N035045</td>
<td>DBNAME</td>
</tr>
<tr>
<td>N035046</td>
<td>SYSTRACE</td>
</tr>
<tr>
<td>N035047</td>
<td>SCRATCH</td>
</tr>
<tr>
<td>N035048</td>
<td>TCP/IP</td>
</tr>
<tr>
<td>N035049</td>
<td>CHANGE TRACKING</td>
</tr>
<tr>
<td>N035051</td>
<td>AUTOTUNE</td>
</tr>
<tr>
<td>N035052</td>
<td>MODID</td>
</tr>
<tr>
<td>N036</td>
<td>DISPLAY PRINTERS</td>
</tr>
<tr>
<td>N036001</td>
<td>DISPLAY PRINTERS</td>
</tr>
<tr>
<td>N036002</td>
<td>DISPLAY PRINTER printer-id</td>
</tr>
<tr>
<td>N037</td>
<td>DISPLAY CLASSES/REPORTS</td>
</tr>
<tr>
<td>N037001</td>
<td>DISPLAY CLASSES/REPORTS</td>
</tr>
<tr>
<td>N037002</td>
<td>DISPLAY CLASS/REPORTS CLASS printer-class</td>
</tr>
<tr>
<td>N037003</td>
<td>DISPLAY REPORTS DESTINATION printer-destination</td>
</tr>
<tr>
<td>N038</td>
<td>VARY PRINTER</td>
</tr>
<tr>
<td>N038001</td>
<td>DRAIN</td>
</tr>
<tr>
<td>N038002</td>
<td>REQUEUE</td>
</tr>
<tr>
<td>N038003</td>
<td>START</td>
</tr>
<tr>
<td>N038004</td>
<td>CANCEL</td>
</tr>
<tr>
<td>N038005</td>
<td>CLASSES</td>
</tr>
<tr>
<td>N038006</td>
<td>DESTINATION TO</td>
</tr>
<tr>
<td>N038007</td>
<td>DESTINATION ONLINE</td>
</tr>
<tr>
<td>N038008</td>
<td>DESTINATION OFFLINE</td>
</tr>
<tr>
<td>N039</td>
<td>VARY REPORT</td>
</tr>
<tr>
<td>N039001</td>
<td>DELETE</td>
</tr>
<tr>
<td>N039002</td>
<td>TO CLASS</td>
</tr>
<tr>
<td>N039003</td>
<td>TO DESTINATION</td>
</tr>
<tr>
<td>N039004</td>
<td>COPIES</td>
</tr>
<tr>
<td>N039005</td>
<td>FIRST</td>
</tr>
<tr>
<td>N039006</td>
<td>LAST</td>
</tr>
<tr>
<td>N039007</td>
<td>HOLD</td>
</tr>
<tr>
<td>N039008</td>
<td>RELEASE</td>
</tr>
<tr>
<td>N039009</td>
<td>KEEP</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>N040</td>
<td>DISPLAY REPLIES</td>
</tr>
<tr>
<td>N040000</td>
<td>DISPLAY REPLIES</td>
</tr>
<tr>
<td>N041</td>
<td>DISPLAY/VARY/WRITE STATISTICS</td>
</tr>
<tr>
<td>N041001</td>
<td>DISPLAY STATISTICS INTERVAL</td>
</tr>
<tr>
<td>N041002</td>
<td>VARY STATISTICS INTERVAL interval</td>
</tr>
<tr>
<td>N041003</td>
<td>VARY STATISTICS INTERVAL OFF</td>
</tr>
<tr>
<td>N041004</td>
<td>WRITE STATISTICS</td>
</tr>
<tr>
<td>N041005</td>
<td>DISPLAY STATISTICS SYSTEM</td>
</tr>
<tr>
<td>N041006</td>
<td>VARY STATISTICS TRANSACTION ON</td>
</tr>
<tr>
<td>N041007</td>
<td>VARY STATISTICS TRANSACTION OFF</td>
</tr>
<tr>
<td>N041008</td>
<td>DISPLAY STATISTICS ROLL</td>
</tr>
<tr>
<td>N041009</td>
<td>VARY STATISTICS ROLL TIME HH:MM</td>
</tr>
<tr>
<td>N041010</td>
<td>VARY STATISTICS ROLL FRE ddd</td>
</tr>
<tr>
<td>N041011</td>
<td>VARY STATISTICS ROLL TIME HH:MM FRE ddd</td>
</tr>
<tr>
<td>N041012</td>
<td>VARY STATISTICS NOROLL</td>
</tr>
<tr>
<td>N041013</td>
<td>WRITE STATISTICS ROLL</td>
</tr>
<tr>
<td>N042</td>
<td>DISPLAY JOURNAL</td>
</tr>
<tr>
<td>N042000</td>
<td>DISPLAY JOURNAL</td>
</tr>
<tr>
<td>N042001</td>
<td>DISPLAY JOURNAL journal-name</td>
</tr>
<tr>
<td>N043</td>
<td>VARY JOURNAL</td>
</tr>
<tr>
<td>N043000</td>
<td>VARY JOURNAL</td>
</tr>
<tr>
<td>N044</td>
<td>VARY UCF FETID</td>
</tr>
<tr>
<td>N044001</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N044002</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N044005</td>
<td>QUIESCE</td>
</tr>
<tr>
<td>N045</td>
<td>VARY UCF SYSTEM</td>
</tr>
<tr>
<td>N045001</td>
<td>ONLINE</td>
</tr>
<tr>
<td>N045002</td>
<td>QUIESCE</td>
</tr>
<tr>
<td>N045003</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N045004</td>
<td>NEW COPY</td>
</tr>
<tr>
<td>N046</td>
<td>VARY DYNAMIC</td>
</tr>
<tr>
<td>N046001</td>
<td>PROGRAM</td>
</tr>
<tr>
<td>N046002</td>
<td>TASK</td>
</tr>
<tr>
<td>N047</td>
<td>DISPLAY LINES</td>
</tr>
<tr>
<td>N047001</td>
<td>DISPLAY LINES</td>
</tr>
<tr>
<td>N047002</td>
<td>DISPLAY LINE</td>
</tr>
<tr>
<td>N048</td>
<td>DISPLAY/VARY DDS</td>
</tr>
<tr>
<td>N048001</td>
<td>DISPLAY DDS LINE</td>
</tr>
<tr>
<td>N048002</td>
<td>DISPLAY DDS PTERM</td>
</tr>
<tr>
<td>N048003</td>
<td>DISPLAY DDS</td>
</tr>
<tr>
<td>N048004</td>
<td>VARY PTERM WEIGHT</td>
</tr>
<tr>
<td>N049</td>
<td>DISPLAY/VARY DBTABLE</td>
</tr>
<tr>
<td>N049000</td>
<td>DISPLAY DBTABLE</td>
</tr>
<tr>
<td>N049001</td>
<td>VARY DBTABLE</td>
</tr>
<tr>
<td>N050</td>
<td>VARY LOADLIB</td>
</tr>
<tr>
<td>N050000</td>
<td>OFFLINE</td>
</tr>
<tr>
<td>N050001</td>
<td>ONLINE</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>N051</td>
<td>DISPLAY LOADLIBS/LOADLISTS/DICTIONARIES</td>
</tr>
<tr>
<td>N051001</td>
<td>DISPLAY LOADLIBS</td>
</tr>
<tr>
<td>N051002</td>
<td>DISPLAY LOADLIB</td>
</tr>
<tr>
<td>N051003</td>
<td>DISPLAY DICTIONARIES</td>
</tr>
<tr>
<td>N051004</td>
<td>DISPLAY DICTIONARY</td>
</tr>
<tr>
<td>N051005</td>
<td>DISPLAY LOADLIST</td>
</tr>
<tr>
<td>N051006</td>
<td>DISPLAY LOADLISTS</td>
</tr>
<tr>
<td>N052</td>
<td>DISPLAY ALL STORAGE POOLS</td>
</tr>
<tr>
<td>N052001</td>
<td>DISPLAY ALL STORAGE POOLS</td>
</tr>
<tr>
<td>N053</td>
<td>VARY SNAP</td>
</tr>
<tr>
<td>N053000</td>
<td>SYSTEM ON</td>
</tr>
<tr>
<td>N053001</td>
<td>SYSTEM OFF</td>
</tr>
<tr>
<td>N053002</td>
<td>SYSTEM PHOTO</td>
</tr>
<tr>
<td>N053003</td>
<td>SYSTEM NOPHOTO</td>
</tr>
<tr>
<td>N053004</td>
<td>TASK ON</td>
</tr>
<tr>
<td>N053005</td>
<td>TASK OFF</td>
</tr>
<tr>
<td>N053006</td>
<td>TASK PHOTO</td>
</tr>
<tr>
<td>N053007</td>
<td>TASK NOPHOTO</td>
</tr>
<tr>
<td>N053008</td>
<td>TASK TRACE ON</td>
</tr>
<tr>
<td>N053009</td>
<td>TASK TRACE OFF</td>
</tr>
<tr>
<td>N053010</td>
<td>TASK TRACE TASK</td>
</tr>
<tr>
<td>N053011</td>
<td>TASK TRACE LIMIT nnn</td>
</tr>
<tr>
<td>N053012</td>
<td>TASK TRACE LIMIT OFF</td>
</tr>
<tr>
<td>N054</td>
<td>DISPLAY SNAP</td>
</tr>
<tr>
<td>N054000</td>
<td>DISPLAY SNAP</td>
</tr>
<tr>
<td>N055</td>
<td>DISPLAY ALL PROGRAM POOLS</td>
</tr>
<tr>
<td>N055001</td>
<td>DISPLAY ALL PROGRAM POOLS</td>
</tr>
<tr>
<td>N056</td>
<td>DISPLAY/VARY ADSO STATISTICS</td>
</tr>
<tr>
<td>N056001</td>
<td>VARY ADSO STATISTICS ON</td>
</tr>
<tr>
<td>N056002</td>
<td>VARY ADSO STATISTICS ON ALL</td>
</tr>
<tr>
<td>N056003</td>
<td>VARY ADSO STATISTICS ON SELECTED</td>
</tr>
<tr>
<td>N056004</td>
<td>VARY ADSO STATISTICS OFF</td>
</tr>
<tr>
<td>N056005</td>
<td>VARY ADSO STATISTICS CHECKPOINT</td>
</tr>
<tr>
<td>N056006</td>
<td>DISPLAY ADSO STATISTICS</td>
</tr>
<tr>
<td>N056007</td>
<td>VARY ADSO RECORD COMPRESSION ON</td>
</tr>
<tr>
<td>N056008</td>
<td>VARY ADSO RECORD COMPRESSION OFF</td>
</tr>
<tr>
<td>N058</td>
<td>DISPLAY MESSAGE</td>
</tr>
<tr>
<td>N058000</td>
<td>DISPLAY MESSAGE</td>
</tr>
<tr>
<td>N059</td>
<td>DISPLAY/VARY LIMITS</td>
</tr>
<tr>
<td>N059001</td>
<td>DISPLAY LIMITS</td>
</tr>
<tr>
<td>N059002</td>
<td>VARY LIMITS</td>
</tr>
<tr>
<td>N060</td>
<td>DISPLAY LU</td>
</tr>
<tr>
<td>N060001</td>
<td>DISPLAY LUS</td>
</tr>
<tr>
<td>N060002</td>
<td>DISPLAY LU logical-unit-name</td>
</tr>
<tr>
<td>N061</td>
<td>VARY LU</td>
</tr>
<tr>
<td>N061001</td>
<td>VARY LU (all except VARY LU RESET)</td>
</tr>
<tr>
<td>N061002</td>
<td>VARY LU RESET</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>N062</td>
<td>DISPLAY NUCLEUS MODULE RELOAD TABLE</td>
</tr>
<tr>
<td>N062000</td>
<td>DISPLAY NUCLEUS MODULE RELOAD TABLE</td>
</tr>
<tr>
<td>N063</td>
<td>VARY NUCLEUS</td>
</tr>
<tr>
<td>N063001</td>
<td>VARY NUCLEUS MODULE NEW COPY</td>
</tr>
<tr>
<td>N063002</td>
<td>VARY NUCLEUS MODULE CANCEL</td>
</tr>
<tr>
<td>N063004</td>
<td>VARY NUCLEUS MODULE RELOAD</td>
</tr>
<tr>
<td>N064</td>
<td>DISPLAY/VARY DISTRIBUTED</td>
</tr>
<tr>
<td>N064001</td>
<td>DISPLAY DISTRIBUTED TRANSACTION</td>
</tr>
<tr>
<td>N064002</td>
<td>DISPLAY DISTRIBUTED TRANSACTION (X)ID</td>
</tr>
<tr>
<td>N064006</td>
<td>DISPLAY DISTRIBUTED RESOURCE MANAGER</td>
</tr>
<tr>
<td>N064007</td>
<td>DISPLAY DISTRIBUTED RESOURCE MANAGER name</td>
</tr>
<tr>
<td>N064010</td>
<td>VARY DISTRIBUTED TRANSACTION (X)ID</td>
</tr>
<tr>
<td>N064011</td>
<td>VARY DISTRIBUTED RESOURCE MANAGER name</td>
</tr>
<tr>
<td>N065</td>
<td>VARY DB WRITE DRIVER ONLINE</td>
</tr>
<tr>
<td>N065000</td>
<td>VARY DB WRITE DRIVER ONLINE</td>
</tr>
<tr>
<td>N066</td>
<td>VARY JOURNAL DRIVER ONLINE</td>
</tr>
<tr>
<td>N066000</td>
<td>VARY JOURNAL DRIVER ONLINE</td>
</tr>
<tr>
<td>N067</td>
<td>VARY DB WRITE DRIVER OFFLINE</td>
</tr>
<tr>
<td>N067000</td>
<td>VARY DB WRITE DRIVER OFFLINE</td>
</tr>
<tr>
<td>N068</td>
<td>VARY JOURNAL DRIVER OFFLINE</td>
</tr>
<tr>
<td>N068000</td>
<td>VARY JOURNAL DRIVER OFFLINE</td>
</tr>
<tr>
<td>N069</td>
<td>VARY JOURNAL FRAGMENT NUMBER</td>
</tr>
<tr>
<td>N069000</td>
<td>VARY JOURNAL FRAGMENT NUMBER</td>
</tr>
<tr>
<td>N072</td>
<td>DISPLAY RUN UNITS</td>
</tr>
<tr>
<td>N072000</td>
<td>DISPLAY RUN UNIT run-unit-n</td>
</tr>
<tr>
<td>N072002</td>
<td>DISPLAY RUN UNIT QUEUE</td>
</tr>
<tr>
<td>N072003</td>
<td>DISPLAY RUN UNIT LOADER</td>
</tr>
<tr>
<td>N072004</td>
<td>DISPLAY RUN UNIT MSGDICT</td>
</tr>
<tr>
<td>N072005</td>
<td>DISPLAY RUN UNIT SIGNON</td>
</tr>
<tr>
<td>N072006</td>
<td>DISPLAY RUN UNIT SYSTEM/DEST</td>
</tr>
<tr>
<td>N072009</td>
<td>DISPLAY RUN UNIT SECURITY</td>
</tr>
<tr>
<td>N072012</td>
<td>DISPLAY RUN UNIT SQL LOADER</td>
</tr>
<tr>
<td>N072013</td>
<td>DISPLAY RUN UNIT SQL SECURITY</td>
</tr>
<tr>
<td>N073</td>
<td>VARY RUN UNIT</td>
</tr>
<tr>
<td>N073002</td>
<td>VARY RUN UNIT QUEUE</td>
</tr>
<tr>
<td>N073003</td>
<td>VARY RUN UNIT LOADER</td>
</tr>
<tr>
<td>N073004</td>
<td>VARY RUN UNIT MSGDICT</td>
</tr>
<tr>
<td>N073005</td>
<td>VARY RUN UNIT SIGNON</td>
</tr>
<tr>
<td>N073006</td>
<td>VARY RUN UNIT SYSTEM/DESTINATION</td>
</tr>
<tr>
<td>N073009</td>
<td>VARY RUN UNIT SECURITY</td>
</tr>
<tr>
<td>N073012</td>
<td>VARY RUN UNIT SQL LOADER</td>
</tr>
<tr>
<td>N073013</td>
<td>VARY RUN UNIT SQL SECURITY</td>
</tr>
<tr>
<td>N074</td>
<td>VARY DATABASE READ DRIVER ON</td>
</tr>
<tr>
<td>N074000</td>
<td>VARY DATABASE READ DRIVER ON</td>
</tr>
<tr>
<td>N075</td>
<td>VARY DATABASE READ DRIVER OFF</td>
</tr>
<tr>
<td>N075000</td>
<td>VARY DATABASE READ DRIVER OFF</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>N076</td>
<td>DISPLAY SUBTASK/MT</td>
</tr>
<tr>
<td>N076001</td>
<td>DISPLAY MPMODE TABLE</td>
</tr>
<tr>
<td>N076002</td>
<td>DISPLAY SUBTASK</td>
</tr>
<tr>
<td>N076003</td>
<td>DISPLAY SUBTASKS</td>
</tr>
<tr>
<td>N076004</td>
<td>DISPLAY MT QUEUE DEPTH</td>
</tr>
<tr>
<td>N076005</td>
<td>DISPLAY SUBTASK EFFECTIVENESS</td>
</tr>
<tr>
<td>N077</td>
<td>VARY SUBTASK/MT QUEUE</td>
</tr>
<tr>
<td>N077001</td>
<td>VARY SUBTASK N RRS ENABLED</td>
</tr>
<tr>
<td>N077002</td>
<td>VARY SUBTASK N RRS DISABLED</td>
</tr>
<tr>
<td>N077003</td>
<td>VARY MT QUEUE DEPTH nnn</td>
</tr>
<tr>
<td>N078</td>
<td>DISPLAY STATISTICS AREA</td>
</tr>
<tr>
<td>N078000</td>
<td>DISPLAY STATISTICS AREAS</td>
</tr>
<tr>
<td>N078001</td>
<td>DISPLAY STATISTICS AREA area-name</td>
</tr>
<tr>
<td>N078002</td>
<td>DISPLAY STATISTICS AREA area-name FILE</td>
</tr>
<tr>
<td>N078003</td>
<td>DISPLAY STATISTICS AREA area-name BUFFER</td>
</tr>
<tr>
<td>N078004</td>
<td>DISPLAY STATISTICS AREA area-name ALL</td>
</tr>
<tr>
<td>N079</td>
<td>DISPLAY STATISTICS BUFFER</td>
</tr>
<tr>
<td>N079000</td>
<td>DISPLAY STATISTICS BUFFERS</td>
</tr>
<tr>
<td>N079001</td>
<td>DISPLAY STATISTICS BUFFER buffer-name AREA</td>
</tr>
<tr>
<td>N079002</td>
<td>DISPLAY STATISTICS BUFFER buffer-name FILE</td>
</tr>
<tr>
<td>N079003</td>
<td>DISPLAY STATISTICS BUFFER buffer-name</td>
</tr>
<tr>
<td>N079004</td>
<td>DISPLAY STATISTICS BUFFER buffer-name ALL</td>
</tr>
<tr>
<td>N080</td>
<td>DISPLAY STATISTICS FILE</td>
</tr>
<tr>
<td>N080000</td>
<td>DISPLAY STATISTICS FILES</td>
</tr>
<tr>
<td>N080001</td>
<td>DISPLAY STATISTICS FILE file-name AREA</td>
</tr>
<tr>
<td>N080002</td>
<td>DISPLAY STATISTICS FILE file-name</td>
</tr>
<tr>
<td>N080003</td>
<td>DISPLAY STATISTICS FILE file-name BUFFER</td>
</tr>
<tr>
<td>N080004</td>
<td>DISPLAY STATISTICS FILE file-name ALL</td>
</tr>
<tr>
<td>N081</td>
<td>DISPLAY FILES</td>
</tr>
<tr>
<td>N081000</td>
<td>DISPLAY FILES</td>
</tr>
<tr>
<td>N081001</td>
<td>DISPLAY FILE file-name AREA</td>
</tr>
<tr>
<td>N081002</td>
<td>DISPLAY FILE file-name</td>
</tr>
<tr>
<td>N081003</td>
<td>DISPLAY FILE file-name BUFFER</td>
</tr>
<tr>
<td>N081004</td>
<td>DISPLAY FILE file-name ALL</td>
</tr>
<tr>
<td>N081128</td>
<td>DISPLAY FILE file-name LOC</td>
</tr>
<tr>
<td>N082</td>
<td>VARY FILE</td>
</tr>
<tr>
<td>N082000</td>
<td>VARY FILE file-name ACTIVE</td>
</tr>
<tr>
<td>N082001</td>
<td>VARY FILE file-name INACTIVE</td>
</tr>
<tr>
<td>N082002</td>
<td>VARY FILE file-name OPEN</td>
</tr>
<tr>
<td>N082003</td>
<td>VARY FILE file-name CLOSE</td>
</tr>
<tr>
<td>N082004</td>
<td>VARY FILE file-name OPEN UPDATE</td>
</tr>
<tr>
<td>N082005</td>
<td>VARY FILE file-name ALLOCATE</td>
</tr>
<tr>
<td>N082006</td>
<td>VARY FILE file-name DEALLOCATE</td>
</tr>
<tr>
<td>N082007</td>
<td>VARY FILE file-name DEALLOCATE FORCE</td>
</tr>
<tr>
<td>N082008</td>
<td>VARY FILE file-name DATASPACE/MEMORY CACHE YES</td>
</tr>
<tr>
<td>N082009</td>
<td>VARY FILE file-name DATASPACE/MEMORY CACHE NO</td>
</tr>
<tr>
<td>N082010</td>
<td>VARY FILE file-name DSNAMES</td>
</tr>
<tr>
<td>N082011</td>
<td>VARY FILE file-name DISP SHR</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>N082012</td>
<td>VARY FILE file-name DISP OLD</td>
</tr>
<tr>
<td>N082015</td>
<td>VARY FILE file-name PREFETCH ON</td>
</tr>
<tr>
<td>N082016</td>
<td>VARY FILE file-name PREFETCH OFF</td>
</tr>
<tr>
<td>N082017</td>
<td>VARY FILE file-name SHARED CACHE cache-name</td>
</tr>
<tr>
<td>N082018</td>
<td>VARY FILE file-name SHARED CACHE NO</td>
</tr>
<tr>
<td>N083</td>
<td>DISPLAY LOG</td>
</tr>
<tr>
<td>N083001</td>
<td>DISPLAY LOG</td>
</tr>
<tr>
<td>N083002</td>
<td>DISPLAY LOG DRIVERS</td>
</tr>
<tr>
<td>N084</td>
<td>VARY LOG DRIVER</td>
</tr>
<tr>
<td>N084001</td>
<td>VARY LOG DRIVER ONLINE</td>
</tr>
<tr>
<td>N084002</td>
<td>VARY LOG DRIVER OFFLINE</td>
</tr>
<tr>
<td>N084003</td>
<td>VARY LOG</td>
</tr>
<tr>
<td>N085</td>
<td>VARY JOURNAL TRANSACTION LEVEL</td>
</tr>
<tr>
<td>N085000</td>
<td>VARY JOURNAL TRANSACTION LEVEL</td>
</tr>
<tr>
<td>N086</td>
<td>DISPLAY DEADLOCK</td>
</tr>
<tr>
<td>N086001</td>
<td>DISPLAY DEADLOCK DETECTION INTERVAL</td>
</tr>
<tr>
<td>N086002</td>
<td>DISPLAY DEADLOCK DETAILS</td>
</tr>
<tr>
<td>N087</td>
<td>VARY DEADLOCK</td>
</tr>
<tr>
<td>N087001</td>
<td>VARY DEADLOCK DETECTION INTERVAL</td>
</tr>
<tr>
<td>N087003</td>
<td>VARY DEADLOCK DETAILS OFF</td>
</tr>
<tr>
<td>N087004</td>
<td>VARY DEADLOCK DETAILS ON</td>
</tr>
<tr>
<td>N088</td>
<td>DISPLAY PHYSICAL DATABASE</td>
</tr>
<tr>
<td>N088000</td>
<td>DISPLAY PHYSICAL DATABASE</td>
</tr>
<tr>
<td>N088001</td>
<td>DISPLAY PHYSICAL DATABASE database-name</td>
</tr>
<tr>
<td>N089</td>
<td>VARY DMCL</td>
</tr>
<tr>
<td>N089000</td>
<td>VARY DMCL VALIDATE NEW COPY</td>
</tr>
<tr>
<td>N089001</td>
<td>VARY DMCL NEW COPY</td>
</tr>
<tr>
<td>N089002</td>
<td>VARY DMCL PREFETCH ON</td>
</tr>
<tr>
<td>N089003</td>
<td>VARY DMCL PREFETCH OFF</td>
</tr>
<tr>
<td>N089004</td>
<td>VARY DMCL PREFETCH TRACE ON</td>
</tr>
<tr>
<td>N089005</td>
<td>VARY DMCL PREFETCH TRACE OFF</td>
</tr>
<tr>
<td>N089006</td>
<td>VARY DMCL MEMORY CACHE STORAGE LIMIT nnn xB</td>
</tr>
<tr>
<td>N089007</td>
<td>VARY DMCL MEMORY CACHE STORAGE LIMIT OPSYS</td>
</tr>
<tr>
<td>N089008</td>
<td>VARY DMCL MEMORY CACHE LOCATION 64 BIT ONLY</td>
</tr>
<tr>
<td>N089009</td>
<td>VARY DMCL MEMORY CACHE LOCATION ANYWHERE</td>
</tr>
<tr>
<td>N090</td>
<td>DISPLAY SEGMENTS</td>
</tr>
<tr>
<td>N090000</td>
<td>DISPLAY SEGMENTS</td>
</tr>
<tr>
<td>N090001</td>
<td>DISPLAY SEGMENT segment-name</td>
</tr>
<tr>
<td>N091</td>
<td>VARY SEGMENT</td>
</tr>
<tr>
<td>N091000</td>
<td>ACTIVE</td>
</tr>
<tr>
<td>N091001</td>
<td>RETRIEVAL &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N091002</td>
<td>UPDATE (ONLINE) &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N091003</td>
<td>OFFLINE &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N091004</td>
<td>TRANSIENT RETRIEVAL &lt;additional parameters&gt;</td>
</tr>
<tr>
<td>N091011</td>
<td>OPEN UPDATE</td>
</tr>
<tr>
<td>N091012</td>
<td>OPEN</td>
</tr>
<tr>
<td>N091013</td>
<td>CLOSE</td>
</tr>
<tr>
<td>N091014</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>N091015</td>
<td>PURGE</td>
</tr>
<tr>
<td>N091016</td>
<td>QUIESCE</td>
</tr>
<tr>
<td>N091017</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>N091018</td>
<td>SHARED CACHE <code>cache-name</code></td>
</tr>
<tr>
<td>N091020</td>
<td>SHARED CACHE NO</td>
</tr>
<tr>
<td>N091021</td>
<td>DATA SHARING ON</td>
</tr>
<tr>
<td>N091022</td>
<td>DATA SHARING OFF</td>
</tr>
<tr>
<td></td>
<td>ALLOCATE</td>
</tr>
<tr>
<td>N092</td>
<td>DISPLAY/VARY RESOURCE TABLE/NODE</td>
</tr>
<tr>
<td>N092001</td>
<td>DISPLAY RESOURCE TABLE</td>
</tr>
<tr>
<td>N092002</td>
<td>DISPLAY NODE</td>
</tr>
<tr>
<td>N092003</td>
<td>VARY RESOURCE TABLE NEW COPY</td>
</tr>
<tr>
<td>N093</td>
<td>DISPLAY LOCKS</td>
</tr>
<tr>
<td>N093001</td>
<td>DISPLAY LOCKS AREAS</td>
</tr>
<tr>
<td>N093002</td>
<td>DISPLAY LOCKS AREA <code>area-name</code></td>
</tr>
<tr>
<td>N093003</td>
<td>DISPLAY LOCKS LTERMS</td>
</tr>
<tr>
<td>N093004</td>
<td>DISPLAY LOCKS LTERM <code>logical-terminal-id</code></td>
</tr>
<tr>
<td>N093005</td>
<td>DISPLAY LOCK STATISTICS</td>
</tr>
<tr>
<td>N093006</td>
<td>DISPLAY LOCK RECORD DATA</td>
</tr>
<tr>
<td>N093007</td>
<td>DISPLAY LOCK RECORD DATA MEMBER <code>member-name</code></td>
</tr>
<tr>
<td>N094</td>
<td>DISPLAY SYSGEN</td>
</tr>
<tr>
<td>N094001</td>
<td>DISPLAY SYSGEN REFRESH ALL</td>
</tr>
<tr>
<td>N094002</td>
<td>DISPLAY SYSGEN REFRESH LINES</td>
</tr>
<tr>
<td>N094003</td>
<td>DISPLAY SYSGEN REFRESH LINE <code>line-id</code></td>
</tr>
<tr>
<td>N094004</td>
<td>DISPLAY SYSGEN REFRESH STORAGE POOLS</td>
</tr>
<tr>
<td>N094005</td>
<td>DISPLAY SYSGEN REFRESH STORAGE POOL <code>nnn</code></td>
</tr>
<tr>
<td>N094006</td>
<td>DISPLAY SYSGEN REFRESH PROGRAM POOLS</td>
</tr>
<tr>
<td>N094007</td>
<td>DISPLAY SYSGEN REFRESH PROGRAM POOL XAPP/XARP</td>
</tr>
<tr>
<td>N095</td>
<td>VARY SYSGEN</td>
</tr>
<tr>
<td>N095001</td>
<td>VARY SYSGEN REFRESH ALL</td>
</tr>
<tr>
<td>N095002</td>
<td>VARY SYSGEN REFRESH LINES</td>
</tr>
<tr>
<td>N095003</td>
<td>VARY SYSGEN REFRESH LINE <code>line-id</code></td>
</tr>
<tr>
<td>N095004</td>
<td>VARY SYSGEN REFRESH STORAGE POOLS</td>
</tr>
<tr>
<td>N095005</td>
<td>VARY SYSGEN REFRESH STORAGE POOL <code>nnn</code></td>
</tr>
<tr>
<td>N095006</td>
<td>VARY SYSGEN REFRESH PROGRAM POOLS</td>
</tr>
<tr>
<td>N095007</td>
<td>VARY SYSGEN REFRESH PROGRAM POOL XAPP/XARP</td>
</tr>
<tr>
<td>N096</td>
<td>DISPLAY/VARY/TEST DBGROUP/SHARED CACHE/DATA SHARING</td>
</tr>
<tr>
<td>N096001</td>
<td>DISPLAY DBGROUP</td>
</tr>
<tr>
<td>N096002</td>
<td>DISPLAY DBGROUP DEBUG</td>
</tr>
<tr>
<td>N096005</td>
<td>DISPLAY SHARED CACHE</td>
</tr>
<tr>
<td>N096006</td>
<td>DISPLAY SHARED CACHE DEBUG</td>
</tr>
<tr>
<td>N096007</td>
<td>DISPLAY XES LIST &amp;vbar. LOCK <code>strname</code></td>
</tr>
<tr>
<td>N096008</td>
<td>DISPLAY XES LIST &amp;vbar. LOCK <code>strname</code> DEBUG</td>
</tr>
<tr>
<td>N096010</td>
<td>VARY DBGROUP <code>group-name</code> ON</td>
</tr>
<tr>
<td>N096011</td>
<td>VARY DBGROUP <code>group name</code> ACTIVE</td>
</tr>
<tr>
<td>N096012</td>
<td>VARY DBGROUP <code>group-name</code> JOIN</td>
</tr>
<tr>
<td>N096015</td>
<td>VARY DBGROUP <code>group-name</code> OFF</td>
</tr>
<tr>
<td>N096016</td>
<td>VARY DBGROUP <code>group name</code> INACTIVE</td>
</tr>
<tr>
<td>N096017</td>
<td>VARY DBGROUP <code>group-name</code> LEAVE</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>-------</td>
<td>--------------</td>
</tr>
<tr>
<td>N096020</td>
<td>VARY SHARED CACHE <em>cache-name</em> ON</td>
</tr>
<tr>
<td>N096021</td>
<td>VARY SHARED CACHE <em>cache-name</em> OFF</td>
</tr>
<tr>
<td>N096030</td>
<td>DISPLAY DATA SHARING SUMMARY</td>
</tr>
<tr>
<td>N096031</td>
<td>DISPLAY DATA SHARING XES LOCK</td>
</tr>
<tr>
<td>N096032</td>
<td>DISPLAY DATA SHARING XES LIST</td>
</tr>
<tr>
<td>N096033</td>
<td>DISPLAY DATA SHARING XCF GROUP</td>
</tr>
<tr>
<td>N096034</td>
<td>DISPLAY DATA SHARING ALL</td>
</tr>
<tr>
<td>N096035</td>
<td>DISPLAY DATA SHARING DEBUG</td>
</tr>
<tr>
<td>N096036</td>
<td>DISPLAY DATA SHARING DEFAULT CACHE</td>
</tr>
<tr>
<td>N096037</td>
<td>VARY DATA SHARING ON CONNECTIVITY LOSS option</td>
</tr>
<tr>
<td>N096090</td>
<td>TEST DBGROUP count <em>member-name</em></td>
</tr>
<tr>
<td>N096091</td>
<td>TEST SHARED CACHE <em>file-id</em></td>
</tr>
<tr>
<td>N096092</td>
<td>TEST XCFGRP <em>group-name</em> option</td>
</tr>
<tr>
<td>N096093</td>
<td>TEST XCFMSG function <em>message-type</em> option</td>
</tr>
<tr>
<td>N096094</td>
<td>TEST XESLOCK option count</td>
</tr>
<tr>
<td>N096095</td>
<td>TEST XESLIST option count</td>
</tr>
<tr>
<td>N097</td>
<td>TEST LRBK</td>
</tr>
<tr>
<td>N097001</td>
<td>TEST LRBK <em>run-unit-id</em></td>
</tr>
<tr>
<td>N098</td>
<td>QUIESCE</td>
</tr>
<tr>
<td>N098001</td>
<td>QUIESCE AREA</td>
</tr>
<tr>
<td>N098002</td>
<td>QUIESCE SEGMENT</td>
</tr>
<tr>
<td>N098003</td>
<td>QUIESCE DBNAME</td>
</tr>
<tr>
<td>N099</td>
<td>DISPLAY ID</td>
</tr>
<tr>
<td>N099001</td>
<td>DISPLAY ID</td>
</tr>
<tr>
<td>N099002</td>
<td>DISPLAY ID NAME</td>
</tr>
<tr>
<td>N100</td>
<td>DISPLAY CSAFLAGS &amp; VARY CSATST/CSALMGR/CSAHPSC/CSADBIO/CSACFIM</td>
</tr>
<tr>
<td>N100000</td>
<td>DISPLAY CSAFLAGS</td>
</tr>
<tr>
<td>N100001</td>
<td>VARY CSATST <em>nn</em> ON/OFF</td>
</tr>
<tr>
<td>N100002</td>
<td>VARY CSALMGR <em>n</em> ON/OFF</td>
</tr>
<tr>
<td>N100003</td>
<td>VARY CSAHPSC <em>n</em> ON/OFF</td>
</tr>
<tr>
<td>N100004</td>
<td>VARY CSADBIO <em>n</em> ON/OFF</td>
</tr>
<tr>
<td>N100005</td>
<td>VARY CSACFIM <em>n</em> ON/OFF</td>
</tr>
<tr>
<td>N100006</td>
<td>VARY CSATMGR <em>n</em> ON/OFF</td>
</tr>
<tr>
<td>N101</td>
<td>VARY ID</td>
</tr>
<tr>
<td>N101003</td>
<td>VARY ID xxx TERMINATE</td>
</tr>
<tr>
<td>N102</td>
<td>DISPLAY/VARY TRANSACTION SHARING</td>
</tr>
<tr>
<td>N102000</td>
<td>DISPLAY TRANSACTION SHARING</td>
</tr>
<tr>
<td>N102001</td>
<td>VARY TRANSACTION SHARING ON/OFF</td>
</tr>
<tr>
<td>N103</td>
<td>DISPLAY/VARY TRACE/SYSTRACE</td>
</tr>
<tr>
<td>N103001</td>
<td>DISPLAY SYSTRACE</td>
</tr>
<tr>
<td>N103002</td>
<td>VARY SYSTRACE OFF</td>
</tr>
<tr>
<td>N103003</td>
<td>VARY SYSTRACE ON</td>
</tr>
<tr>
<td>N103007</td>
<td>DISPLAY TRACE</td>
</tr>
<tr>
<td>N103008</td>
<td>VARY TRACE</td>
</tr>
<tr>
<td>N104</td>
<td>DISPLAY/VARY SCRATCH</td>
</tr>
<tr>
<td>N104001</td>
<td>DISPLAY SCRATCH</td>
</tr>
<tr>
<td>N104002</td>
<td>VARY SCRATCH</td>
</tr>
<tr>
<td>Code</td>
<td>DCMT Command</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>N105</td>
<td>VARY CHANGE TRACKING</td>
</tr>
<tr>
<td>N105002</td>
<td>FILE COUNT nnn/DELETE</td>
</tr>
<tr>
<td></td>
<td>ON/OFF/REFRESH/ACTIVE/INACTIVE/DISABLE/ENABLE</td>
</tr>
<tr>
<td>N106</td>
<td>DISPLAY CHANGE TRACKING</td>
</tr>
<tr>
<td>N107</td>
<td>DISPLAY TCP/IP</td>
</tr>
<tr>
<td>N107001</td>
<td>DISPLAY TCP/IP ALL</td>
</tr>
<tr>
<td>N107002</td>
<td>DISPLAY TCP/IP SUMMARY</td>
</tr>
<tr>
<td>N107003</td>
<td>DISPLAY TCP/IP STATISTICS</td>
</tr>
<tr>
<td>N107004</td>
<td>DISPLAY TCP/IP STACK TABLE</td>
</tr>
<tr>
<td>N107005</td>
<td>DISPLAY TCP/IP SERVICES</td>
</tr>
<tr>
<td>N107006</td>
<td>DISPLAY TCP/IP SOCKETS</td>
</tr>
<tr>
<td>N108</td>
<td>VARY TCP/IP</td>
</tr>
<tr>
<td>N108001</td>
<td>VARY TCP/IP STATUS</td>
</tr>
<tr>
<td>N108002</td>
<td>VARY TCP/IP TCP_NODELAY</td>
</tr>
<tr>
<td>N108003</td>
<td>VARY TCP/IP DEFAULT STACK</td>
</tr>
<tr>
<td>N108004</td>
<td>VARY TCP/IP INCLUDE/EXCLUDE STACK</td>
</tr>
<tr>
<td>N108005</td>
<td>VARY TCP/IP MAXIMUM SOCKETS</td>
</tr>
<tr>
<td>N108006</td>
<td>VARY TCP/IP MAXIMUM SOCKETS PER TASK</td>
</tr>
<tr>
<td>N108007</td>
<td>VARY TCP/IP SERVICES FILE</td>
</tr>
<tr>
<td>N108008</td>
<td>VARY TCP/IP STACK TABLE</td>
</tr>
<tr>
<td>N109</td>
<td>VARY JOURNAL FILE</td>
</tr>
<tr>
<td>N109000</td>
<td>VARY JOURNAL FILE journal-file-name ACTIVE</td>
</tr>
<tr>
<td>N109001</td>
<td>VARY JOURNAL FILE journal-file-name INACTIVE</td>
</tr>
<tr>
<td>N109005</td>
<td>VARY JOURNAL FILE journal-file-name ALLOCATE</td>
</tr>
<tr>
<td>N109006</td>
<td>VARY JOURNAL FILE journal-file-name DEALLOCATE</td>
</tr>
<tr>
<td>N109007</td>
<td>VARY JOURNAL FILE journal-file-name DEALLOCATE FORCE</td>
</tr>
<tr>
<td>N109010</td>
<td>VARY JOURNAL FILE journal-file-name DSNAME</td>
</tr>
<tr>
<td>N109011</td>
<td>VARY JOURNAL FILE journal-file-name DISP SHR</td>
</tr>
<tr>
<td>N109012</td>
<td>VARY JOURNAL FILE journal-file-name DISP OLD</td>
</tr>
<tr>
<td>N109013</td>
<td>VARY JOURNAL FILE journal-file-name ONLINE</td>
</tr>
<tr>
<td>N109014</td>
<td>VARY JOURNAL FILE journal-file-name OFFLINE (PERmanent)</td>
</tr>
<tr>
<td>N110</td>
<td>DISPLAY AUTOTUNE</td>
</tr>
<tr>
<td>N110001</td>
<td>AUTOTUNE [parameter-name]</td>
</tr>
<tr>
<td>N110002</td>
<td>AUTOTUNE *</td>
</tr>
<tr>
<td>N111</td>
<td>VARY AUTOTUNE</td>
</tr>
<tr>
<td>N111001</td>
<td>AUTOTUNE ... RESET</td>
</tr>
<tr>
<td>N111002</td>
<td>AUTOTUNE ... OFF</td>
</tr>
<tr>
<td>N112</td>
<td>DISPLAY MODID</td>
</tr>
<tr>
<td>N112001</td>
<td>[FROM/TO]</td>
</tr>
<tr>
<td>N112002</td>
<td>module-name</td>
</tr>
<tr>
<td>N113</td>
<td>VARY ZIIP</td>
</tr>
<tr>
<td>N113002</td>
<td>VARY ZIIP OFF</td>
</tr>
<tr>
<td>N113003</td>
<td>VARY ZIIP ON</td>
</tr>
</tbody>
</table>
UTABGEN

Contents

- UTABGEN Syntax (see page 147)
- UTABGEN Parameters (see page 147)
- UTABGEN Usage (see page 148)
- UTABGEN Utility Command Codes (see page 149)

Assigns activity numbers to utility commands.

UTABGEN Syntax

```
#UTABGEN ( command-security-specification )
```

Expansion of Command-Security-Specification

```
security-label, activity-number
```

```
command-code, security-label
```

```
BOTH, BCF, OCF,
```

UTABGEN Parameters

- **security-label,activity-number**
  Defines a security label and associates it with a BCF or OCF activity number.
  A security label can be used to classify utility commands by assigning a security label to one or more utility command codes. All commands with the same security label will then be associated with the security label's activity number.
  A security label must be one alphabetic character (A through Z). You can define a maximum of 26 security labels in the #UTABGEN macro. Valid activity-numbers are in the range of 0-255.

  **Note:** An activity number of zero means no security.

- **BOTH, BCF, OCF,**
  Specifies whether the current group of command code security assignments (within the parentheses), will apply to both BCF and OCF commands, just BCF commands, or just OCF commands. BOTH is the default.
Note: The terms BCF and OCF are used to distinguish between operations processed inside the CV from those processed in the batch address space. This means that the term BCF applies to local mode batch only, while the term OCF applies to both OCF and batch to CV (that is, central mode batch).

- **command-code**
  Defines a code from the predefined utility command code table that represents a utility command. For example, FORMAT identifies the Format utility. PRINTSPACE identifies the Print Space utility.

  Note: A list of valid utility commands follows.

- **security-label**
  Specifies a previously defined security label that you are associating with `command-code`.

- **activity-number**
  Specifies the BCF/OCF activity number you are associating with `command-code`. Valid activity-numbers are in the range of 0-255.

  Note: An activity number of zero means no security.

**UTABGEN Usage**

*Coding Considerations*

All lines except the first one must start in column 16.

All lines except the last one must have a non-blank character in column 72.

*General*

When you use the `#UTABGEN` macro, you can assign an OCF/BCF activity number to one or more Utility commands.

- You can associate a specific OCF/BCF activity number (0 through 255) with a utility command.

- You can associate a security label (A through Z) with a utility command.

An activity number of zero turns off security for that security-label or command-code. Coding zero is a useful way to turn off security, without deleting the command-code from the `#UTABGEN` source definition.

Only commands that are being secured need to be coded. If omitted, they default to an activity code of zero.
Security labels must be defined before they can be assigned to command codes.

Use of security labels makes it easier to maintain security definitions when several commands are assigned the same OCF/BCF activity number. You define a security label in the #UTABGEN macro itself. You need to only change the security label definition in the #UTABGEN macro to modify the security for all associated DCMT commands.

*Generating the #UTABGEN Macro*

The source file that specifies the #UTABGEN macro can only contain one macro. Once assembled, the resulting object must be link edited as the stand-alone module IDMSUTAB.

*Example 1:*

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
#UTABGEN (A,3,B,10), (OCF,FORMAT,A,LOCK,A,UNLOCK,A), (BCF,ARCHIVEJOURNAL,B,ARCHIVELOG,B), (FIXPAGE,50)
END
```

This example shows an activity code of 3 being assigned to security-label A and activity code 10 being assigned to security-level B.

OCF indicates the commands that follow (within the parentheses) will be assigned activity codes only when running in the online command facility OCF or as part of the batch command facility IDMSBCF running in central mode. Commands FORMAT, LOCK and UNLOCK are associated with security-label A. Since security-label A is currently assigned to the OCF/BCF activity code 3, the Format, Lock, and Unlock commands are assigned activity code 3.

BCF indicates that the commands within that group will only be secured when running as part of the batch command facility IDMSBCF running in local mode only. In this example, the Archive Journal and Archive Log commands will be assigned to the activity code 10, via the security-label B.

FIXPAGE is not qualified so the activity code 50 will be assigned to the Fix Page utility in both OCF and BCF.

*Example 2:*

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
#UTABGEN (FORMAT,14,FIXPAGE,14)
END
```

In this example, the activity number of 14 is assigned the utility command codes FORMAT and FIXPAGE. Because the codes are not identified as being either OCF or BCF, the commands associated with these codes will be secured in both online and batch, and both will use the same activity number.

**UTABGEN Utility Command Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Utility Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARCHIVEJOURNAL</td>
<td>Archive Journal</td>
</tr>
<tr>
<td>Code</td>
<td>Utility Command</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>ARCHIVETRACE</td>
<td>ARCHIVE TRACE</td>
</tr>
<tr>
<td>ARCHIVELOG</td>
<td>Archive Log</td>
</tr>
<tr>
<td>BACKUP</td>
<td>Backup</td>
</tr>
<tr>
<td>BUILD</td>
<td>Build Indexes/Constraints</td>
</tr>
<tr>
<td>CLEANUP</td>
<td>Cleanup Segment/Area</td>
</tr>
<tr>
<td>CONVERTCATALOG</td>
<td>Convert catalog</td>
</tr>
<tr>
<td>CONVERTPAGE</td>
<td>Convert Page</td>
</tr>
<tr>
<td>EXPANDPAGE</td>
<td>Expand Page</td>
</tr>
<tr>
<td>FASTLOAD</td>
<td>Fastload</td>
</tr>
<tr>
<td>FIXARCHIVE</td>
<td>Fix Archive</td>
</tr>
<tr>
<td>FIXPAGE</td>
<td>Fix Page</td>
</tr>
<tr>
<td>FORMAT</td>
<td>Format Area/Segment/File</td>
</tr>
<tr>
<td>INSTALLSTAMPS</td>
<td>Install Stamps</td>
</tr>
<tr>
<td>LOAD</td>
<td>Load</td>
</tr>
<tr>
<td>LOCK</td>
<td>Lock Area/Segment</td>
</tr>
<tr>
<td>MAINTAINASF</td>
<td>Maintain ASF</td>
</tr>
<tr>
<td>MAINTAININDEX</td>
<td>Maintain Index</td>
</tr>
<tr>
<td>MERGEARCHIVE</td>
<td>Merge Archive</td>
</tr>
<tr>
<td>PRINTINDEX</td>
<td>Print Index</td>
</tr>
<tr>
<td>PRINTJOURNAL</td>
<td>Print Journal</td>
</tr>
<tr>
<td>PRINTLOG</td>
<td>Print Log</td>
</tr>
<tr>
<td>PRINTPAGE</td>
<td>Print Page</td>
</tr>
<tr>
<td>PRINTSPACE</td>
<td>Print Space</td>
</tr>
<tr>
<td>PRINTTRACE</td>
<td>PRINT TRACE</td>
</tr>
<tr>
<td>PUNCHLOADMODULE</td>
<td>Punch Load Module</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Reload</td>
</tr>
<tr>
<td>REORG</td>
<td>Unload and reload all or part of a database</td>
</tr>
<tr>
<td>RESTORE</td>
<td>Restore</td>
</tr>
<tr>
<td>RESTRUCTURE</td>
<td>Restructure</td>
</tr>
<tr>
<td>RESTRUCTURECONNECT</td>
<td>RestructureConnect</td>
</tr>
<tr>
<td>ROLLBACK</td>
<td>RollBack</td>
</tr>
<tr>
<td>ROLLFORWARD</td>
<td>RollForward</td>
</tr>
<tr>
<td>SETOPTIONS</td>
<td>Set BCF/OCF Options</td>
</tr>
<tr>
<td>SYNCHRONIZESTAMPS</td>
<td>Synchronize stamps</td>
</tr>
<tr>
<td>TUNEINDEX</td>
<td>Tune Index</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Unload</td>
</tr>
</tbody>
</table>
GTABGEN

Contents

- GTABGEN Syntax (see page 151)
- GTABGEN Parameters (see page 151)
- GTABGEN Usage (see page 152)

Assigns activity numbers to online debugger security categories.

GTABGEN Syntax

```
#GTABGEN debugger-security-specification
```

Expansion of Debugger-Security-Specification

```
( security-label, activity-number ),
( security-category, security-label activity-number )
```

GTABGEN Parameters

- **security-label,activity-number**
  Associates an online debugger security label with an activity number.
  *Security-label* specifies a label with which you can associate one or more online debugger security categories.
  *Security-label* must be one alphabetic character (A through Z). You can define at most 26 security labels in the #GTABGEN macro.
  *Activity-number* specifies the activity number you are associating with *security-label*. *Activity-number* must be a numeric value in the range 1 through 256.

- **security-category**
  Specifies the online debugger security category with which you are associating the activity number or security label.
  *Security-category* can be any value listed in the "Category" column of the following table, which presents each online debugger security category and its description:
CA IDMS - 19.0
Category Description
ALLR

Any entities named in the following categories can be retrieved.

ALLU

Any entities named in the following categories can be updated.(1)

ASYSPGR CA ADS runtime system programs (ADSOMAIN, ADSORUN1, and ADSODBUG) can be
retrieved.
ASYSPG CA ADS runtime system programs can be updated.(1)
U
ASYSTGR Storage acquired by the CA ADS runtime system can be retrieved. System storage includes
the control block for the online work area (OWA), the online terminal block (OTB), the
online terminal block extension (OTBX), and the variable dialog block (VDB).
ASYSTG Storage acquired by the CA ADS runtime system can be updated.(1)
U
AUPGM CA ADS user programs can be retrieved. These include the fixed dialog block (FDB), the
R
application definition block (ADB), and the task application table (TAT).
AUPGM CA ADS user programs can be updated.(1)
U
SHSTGR Shared storage can be retrieved.
SHSTGU Shared storage can be updated.(1)
UPGMR User programs, subschemas, maps, and tables can be retrieved.
UPGMU User programs, subschemas, maps, and tables can be updated.(1)
USTGR

User storage can be retrieved.

USTGU

User storage can be updated.(1)

(1) Update mode also implies that programs can be retrieved. In retrieval mode you can list
and set breakpoints.

security-label
Specifies the online debugger security label that you are associating with security-category.
This occurrence of security-label must match a security label that is specified in a prior
parenthesized list in the macro.
activity-number
Specifies the online debugger activity number that you are associating with security-category.

GTABGEN Usage
Generating the #GTABGEN Macro
To assemble and link edit the #GTABGEN macro, use the JCL or commands appropriate to your
operating system.

10-Jan-2018

152/367


In the following sample #GTABGEN macro, two security labels are defined:

- A, which is assigned activity number 20
- B, which is assigned activity number 30

UPGMU, the security category that allows updates of user programs, is assigned label A (and, therefore, activity number 20). USTGU, the category that allows updates of storage, is assigned security label B (and, therefore, activity number 30). ALLR, the category that allows retrieval only for all programs and storage is assigned activity number 50.

```
#GTABGEN(A,20,B,30),
(UPGMU,A,USTGU,B,ALLR,50)
```

SECHECK

Contents

- SECHECK Authorization (see page 153)
- SECHECK Syntax (see page 153)
- SECHECK Parameters (see page 154)
- SECHECK Usage (see page 159)

Checks a user's authority to access a resource.

SECHECK Authorization

No authorization required.

SECHECK Syntax

```
#SECHECK
SRB= request-block-area-name request-block-address
.RESTYPE= resource-type-variable resource-type-address 'resource-type-name'
.RESTYPE= resource-name-variable resource-name-address 'resource-name'
.AUTHRTY= ( authority-name authority-bit-indicator )
```
SECHECK Parameters

- **SRB=**
  Specifies the area containing the security request block (SRB) associated with this request. This area is mapped by DSECT #SECRB.
  The location of the SRB is required because most of the other parameters coded on the #SECHECK will be stored in the SRB.

- **request-block-area-name**
  Specifies the symbolic name of the area containing the SRB.

- **request-block-address**
  Specifies the register containing the address of the SRB.

- **,RESTYPE=**
  Specifies the type of the resource being checked. The name of the resource type must be one- to four-characters. At runtime the resource type must be defined to CA IDMS centralized security.

⚠️ **Note:** For a list of valid CA IDMS resource types, see #SECRTT (see page 160).
#SECHECK does not validate the resource type. However, the security system will fail the request at runtime if the type is not valid.

- **resource-type-variable**
  Specifies a user-defined field containing the name of the resource type. The field must be at least four bytes in length. The name of the resource type must be left-justified and padded with blanks.

- **resource-type-address**
  Specifies a register containing the address of the resource type name.

- `'resource-type-name'`
  Specifies a literal that is the name of the resource type.

- `.RESNAME=`
  Specifies the name of the resource being checked.

- **resource-name-variable**
  Specifies a user-defined field containing the name of the resource. The resource name must be left-justified and padded with blanks or binary zeros.

- **resource-name-address**
  Specifies a register containing the address of the resource name.

- `'resource-name'`
  Specifies a literal that is the name of the resource.

- `.AUTHRTY=`
  Specifies the authorities the user must hold to gain access to the resource being checked.

- **authority-name**
  Specifies the name of the authority. If you specify only one authority, you may omit parentheses. Valid authority names are as follows:

<table>
<thead>
<tr>
<th>Authority Name</th>
<th>Authority Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER</td>
<td>DROP</td>
</tr>
<tr>
<td>CREATE</td>
<td>EXECUTE</td>
</tr>
<tr>
<td>DBADMIN</td>
<td>INSERT</td>
</tr>
<tr>
<td>DBAREAD</td>
<td>REFERENCES</td>
</tr>
<tr>
<td>DBAWRITE</td>
<td>SELECT</td>
</tr>
<tr>
<td>DCADMIN</td>
<td>SIGNON</td>
</tr>
<tr>
<td>DELETE</td>
<td>SYSADMIN</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>UPDATE</td>
</tr>
<tr>
<td></td>
<td>USE</td>
</tr>
</tbody>
</table>

- **authority-bit-indicator**
  Specifies a user-defined six-byte field or a register containing the address of such a field. You must set the bits in the field corresponding to the required authorities.

- `.CATEGORY=category-indicator`
  Specifies whether the security system should update the category table from the catalog before proceeding with the security check.
If "category-indicator" is 0, the security system will unconditionally update the category table from the catalog before making the security check. If "category-indicator" is a non-zero value, the security system will update the category table from the catalog before making the security check only if the category has not already been retrieved. "category-indicator" must be one of the following:

- The name of a user-defined halfword field containing the indicator
- A register containing the indicator value
- A literal representing the indicator

If the category is zero, the security system will retrieve the assigned value from the security database and return it in the SRB. "CATEGORY" is valid only for resource types which can be secured by categories. If you specify "CATEGORY" for the resource, do not specify the APPLFNC parameter.

- ',DBNAME='
  Specifies the database name for the load area in which resource types such as database resources and load modules for CA ADS dialogs reside. The runtime security system ignores the contents of this field for other resource types.
  **Resource type DB**: Use the RESNAME parameter to supply database name if the security check is for resource type DB.

- 'database-name-variable'
  Specifies the name of a user-defined field that contains the database name. The database name must be left-justified and padded with blanks or binary zeros.

- 'database-name-address'
  Specifies the register with the address of the user-defined field that contains the database name.

- 'database-name'
  Supplies the database name as a character string literal.

- ',VERSION='
  Specifies the version for load modules (resource type SLOD) and non-SQL schemas (resource type NSCH). The value in this parameter must be in the form Vnnnn, where nnnn is the version in character format.
  If you specify VERSION= for the resource, do not specify the DDNAME parameter.

- 'version-number-variable'
  Specifies the name of a user-defined field that contains the version number.

- 'version-number-address'
  Specifies the register with the address of the user-defined field that contains the version number.

- 'version-number'
  Supplies the version number as a character string literal.

- 'DDNAME='
  Specifies the ddname defining the operating system library in which a program (resource type SPGM) resides.
  If you specify DDNAME for the resource, do not specify the VERSION= parameter.
- **ddname-variable**
  Specifies the name of a user-defined field that contains the ddname.
  The ddname must be left-justified and padded with blanks.

- **ddname-address**
  Specifies the register with the address of the user-defined field that contains the ddname.

- `'ddname'`
  Supplies the ddname as a character string literal.

- `SCHEMA=`
  Specifies the name of the schema for SQL tables (resource type TABL) and access modules (resource type DACC and SACC).
  **Resource type SCHEMA:** Use the RESNAME parameter to specify schema name if the security check is for a schema (resource types QSCH and NSCH).

- **schema-name-variable**
  Specifies the name of a user-defined field that contains the schema name.
  Schema name must be left-justified and padded with blanks.

- **schema-name-address**
  Specifies the register with the address of the user-defined field that contains the schema name.

- `'schema-name'`
  Supplies the schema name as a character string literal.

- `SUBSCHEMA=`
  Specifies the subschema name for native run units (resource type NRU).

- **subschema-name-variable**
  Specifies the name of a user-defined field that contains the subschema name.
  Subschema name must be left-justified and padded with blanks.

- **subschema-name-register**
  Specifies the register with the address of the user-defined field that contains the subschema name.

- `'subschema-name'`
  Supplies the subschema name as a character string literal.

- `APPLFNC=`
  Specifies the number associated with an application function. Each application can have up to 256 functions, numbered 1 through 256. Function numbers must be unique within a given application but need not be unique across applications.
  APPLFNC is valid for activities (resource type ACTI) only. If you specify APPLFNC for the resource, do not specify the CATEGRY parameter.

- **application-function-indicator**
  Supplies the application function number.
  **Application-function-indicator** can be one of the following:
  - User-supplied halfword field
- Register containing the function number
- Numeric literal

- **,.RESLIST=**
  Specifies a list of resources to be checked on this call.
  The resource list contains one entry for each resource being checked. All resources in the list must be of the same resource type.

  **Note:** For more information on the format of the entries in a resource list, see DSECT #SECRLST (see page 348) in Security Database Information and DSECTs.

- **resource-list-indicator**
  Specifies one of the following:
  - A user-defined field that contains the name associated with the first entry in the resource list.
  - The register with the address of the first entry in the resource list.

- **,.RESLCNT=**
  Specifies the number of entries in the list specified by RESLIST.

- **entry-count-indicator**
  Specifies one of the following:
  - A user-defined halfword field that contains the entry count.
  - The register that contains the entry count.
  - The entry count represented by a numeric literal.

- **,.RGSV=**
  Specifies that one or more registers are to be saved across the call. This parameter is valid in system mode only.

- **,.register-number**
  Specifies a register.
  *Register-number* must be a numeric literal.

- **,.CALL=**
  Controls the expansion of the #SECHECK macro.
  If you omit the CALL parameter, the effect is the same as specifying CALL=YES.

  - **YES**
    Causes #SECHECK to generate both the code to complete the SRB and invoke the security system.

  - **NO**
    Causes #SECHECK to generate the code to fill in the SRB fields, but not to build the parameter list or the call.
- **ONLY**
  Causes #SECHECK to generate only the code needed to invoke the security system.

- **CLEAR=**
  Specifies whether you want the SRB to be initialized.
  If you omit the CLEAR= parameter, the effect is the same as specifying CLEAR=YES.

- **YES**
  Causes #SECHECK to clear the SRB to binary zeros before the macro expansion begins to assign values.

- **NO**
  Indicates that the SRB should not be initialized.

- **PLIST=**
  Specifies the address of the area in which to build the parameter list.

- **SYSPLIST**
  Supplies the default name for the area that contains the parameter list.

- **parameter-list-area-indicator**
  Overrides the default area name SYSPLIST by specifying one of these:
  - The name of a user-defined fullword-aligned field that contains the address of the parameter list
  - A register with the address of the parameter list

### SECHECK Usage

**Copying the Security Request Block**

To issue the #SECHECK macro, you must copy the Security Request Block. This block is mapped by the #SECRB DSECT.

⚠️ **Note:** For more information, see the documentation of #SECRB (see page 346).

**Multiple Security Checks in a Single Request**

The #SECHECK function supports multiple security checks in a single request if all the resources are of the same type. For example, all authorities needed by an access module can be validated in a single request.

An application can request that authorization for a list of resources be checked. This reduces the number of calls to the security manager.

**Validating Parameters**
#SECHECK does not validate parameters based on resource type. All parameters supplied on the call are stored in the SRB. The security system ignores information which is irrelevant to the resource type.

## #SECHECK Return Codes

The return code for a security check is stored in register 15 and the SRBXR15 field of the SRB. The following table lists the possible return codes provided by the internal security system in response to a #SECHECK macro:

<table>
<thead>
<tr>
<th>Code</th>
<th>Resource grouping</th>
<th>Meaning Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>Request was successful; access allowed</td>
</tr>
<tr>
<td>04</td>
<td></td>
<td>Resource occurrence or object (user, group, etc.) not found</td>
</tr>
<tr>
<td>08</td>
<td></td>
<td>User not authorized; access denied</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>Interface/parameter list error</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>Resource access threshold violation</td>
</tr>
</tbody>
</table>

When using the multiple security check option, register R15 contains return code 0 only if the user has access to all the resources. The security system places the return codes for the individual resources in the area (RTNADDR) supplied by the caller.

### Return Codes for a List of Resources

To check the return code for each entry in a list of resources supplied with a security request, you must copy #SECRLST. When using this option, determine the results from all list entries if the return code is not 12 (invalid request). R15=0 does not necessarily mean the user is authorized for all entries in the list.

### External Return Codes

If the external security system return code is 4, the issuer of #SECHECK will receive a return code of 8. Other non-zero codes from the external system will be passed through as is. The CA IDMS internal security system is not used as a fallback system for external security.

---

**SECRRTT**

**Contents**

- SECRRTT Authorization (see page 161)
- SECRRTT Syntax (see page 161)
- SECRRTT Parameters (see page 162)
- SECRRTT Usage (see page 169)

Generates the table used to route security check requests for each resource type to the internal or external security system.
**SECRTT Authorization**

Authorization to assemble the `#SECRTT` macro, if any, is defined in an external security system.

**SECRTT Syntax**

*Syntax for TYPE=INITIAL*

```
#SECRTT TYPE=INITIAL

.envname= environment-name

.sgnret= time-interval

.sysprof= (OFF NULL USER GROUP SYSTEM DEFAULT)

.usprof= (OFF NULL USER GROUP SYSTEM)

.df1tsgn= YES

.df1tuid= (VTAMNODE PTERMID LTERMID)

.extruid= user-identifier

.maxresn= resource-entries

.svcnum= svc-number
```

*Syntax for TYPE=ENTRY and TYPE=OCCURRENCE*

```
#SECRTT TYPE=ENTRY OCCURRENCE.RESNAME= 'resource-name'.

.restype= resource-type-name

.secb= EXTERNAL

.extcls= resource-class-name
```

10-Jan-2018 161/367
SECRTT Parameters

- **TYPE=**
  Specifies the type of action to result from assembling the macro. In a series of #SECRRT macros, the first of the series must specify TYPE=INITIAL and the last must specify TYPE=FINAL.

  - **INITIAL**
    Specifies that entries in the SRTT for all CA IDMS-defined resources are to be initialized:
    For each resource type, the initial values are the following:
    - SECBY=OFF
    - EXTNAME=(RESNAME)
    - EXTCLS=blanks

- **ENVNAME=environment-name**
  Specifies a name for the environment that uses the SRTT. Environment-name can be used in external resource name construction. Environment-name must be one to eight characters in length.

- **NULL**
  Specifies that there is no name for the environment that uses the SRTT.

- **SGNRETN**
  Specifies whether CA IDMS should retain signon information originating from external request units (ERUs). This option will provide performance improvements in environments which process large numbers of short-lived ERUs and external security systems.

  - **time-interval**
    Specifies the time in minutes that CA IDMS should retain signon information for external request units after the last session has been ended by signoff.
    You can specify the CA IDMS command, DCUF SHOW USERS ALL, to show the retained users signons with an LTERMID of *NONE*.
    **Note:** If a user signs on to the CA IDMS CV through a VTAM or TSO UCF connection and this is the last (or only) session, a FULL signoff will be performed and the retained signon information and control blocks will be freed from the CA IDMS CV.
- **OFF**
  Specifies that a full signoff, which frees all retained control blocks, will be performed at the end of the last (or only) session for the user. OFF is the default.

- **SYSPROF=**
  Specifies the default SYSTEM profile and whether SYSTEM profiles should be processed for external run units.
  - **OFF**
    Specifies that no SYSTEM profile should be processed.
    **Note:** If SYSTEM profiles are OFF, they will be off for all tasks including external run units, regardless of the setting of the second subparameter.
  - **NULL**
    Specifies that there is no default SYSTEM profile.
  - **USER**
    Specifies that the default SYSTEM profile name is the user-id.
  - **GROUP**
    Specifies that the default SYSTEM profile name is the name of the user's default group.
  - **SYSTEM**
    Specifies that the default SYSTEM profile name is the SYSTEM ID defined in SYSGEN.
  - **profile-name/DEFAULT**
    Specifies the name of the default profile. The profile name must be 1 to 18 characters.
    - **ON**
      Indicates that profiles should be processed for external run units. The default profile, if any, is specified by the first subparameter.
    - **OFF**
      Indicates that profiles should not be processed for external run units. The default is OFF.

- **USRPROF=**
  Specifies the default USER profile and whether USER profiles should be processed for external run units.
  - **OFF**
    Specifies that no USER profile should be processed.
    **Note:** If USER profiles are OFF, they will be off for all tasks including external run units, regardless of the setting of the second subparameter.
  - **NULL**
    Specifies that there is no default USER profile.
  - **USER**
    Specifies that the default USER profile name is the user-id.
- **GROUP**
  Specifies that the default USER profile name is the name of the user's default group.

- **SYSTEM**
  Specifies that the default USER profile name is the SYSTEM ID defined in SYSGEN.

- **profile-name/DEFAULT**
  Specifies the name of the default profile. The profile name must be 1 to 18 characters.
  - **ON**
    Indicates that profiles should be processed for external run units. The default profile, if any, is specified by the first subparameter.
  - **OFF**
    Indicates that profiles should not be processed for external run units. The default if OFF.

- **DFLTSGN=**
  Specifies whether CA IDMS should perform a signon using a specific name if a security check is issued and the terminal operator has not signed on. The name to use for the default signon is defined by the DFLTUID parameter.
  - **YES**
    Enables default signon.
  - **NO**
    Disables this option.

- **DFLTUID=**
  Specifies the default signon CA IDMS is to use when the DFLTSGN parameter is enabled, a security check is issued and the terminal operator has not signed on. Specify a user-identifier or a list of up to three ID options in parentheses. If DFLTSGN=YES, and you don't specify DFLTUID parameters, the default is as follows: (VTAMNODE, PTERMID, LTERMID).
  - **user-identifier**
    Specifies the default signon as an unquoted literal from 1 to 18 characters in length.

- **VTAMNODE**
  Specifies that for VTAM terminals, the VTAM node name is used as the default signon.

- **PTERMID**
  Specifies that the PTERM ID is used as the default signon, if the PTERM is available and the option has not been satisfied by the VTAMNODE parameter (non-VTAM terminals, or VTAMNODE not specified for VTAM terminals).

- **LTERMID**
  Specifies that the LTERM ID is used as the default signon, if the option has not been satisfied by the VTAMNODE or PTERMID parameters.

- **EXTRUID=**
  Specifies the extract user ID that can be used at sites that do not have an external security system. **User-identifier** is an unquoted literal from 1- to 18-characters.
- **MAXRESN=max-resource-entries**
  Specifies maximum number of entries in the #SECRTT global table. If the default of 150 entries is exceeded, the assembly of the #SECRTT fails with condition code of 12 and an assembler error message displays: "12, SRTT GLOBAL TABLE OVERFLOW. GENERATION ABORTED". When this error message is received, review the #SECRTT entries. Check the wildcards to ensure they are valid and used properly. When wildcards are used properly, they reduce the number of entries in #SECRTT global table.

  **Important!** Excessive entries require CPU time to resolve each security check.

  150
  This is the default. It can be increased if necessary.

- **SVCNUM=svc-number**
  Specifies the installed SVC number. This parameter is required. If svc-number is not specified, the system defaults to 175.

- **ENTRY**
  Specifies that the user-supplied values apply to all occurrences of the resource type identified in the RESTYPE parameter.
  For each resource type whose default values you want to replace in SRTT, you must issue a #SECRTT macro with TYPE=ENTRY.

- **OCCURRENCE**
  Specifies that the user-supplied values apply to one occurrence of the resource type identified in the RESTYPE parameter.

  **Note:** TYPE=OCCURRENCE is valid only for resource types DB, SPGM, and TASK.

  EXTCLS= and EXTNAME= specifications are ignored if TYPE=OCCURRENCE. Therefore, if you specify TYPE=OCCURRENCE and SECBY=EXTERNAL to secure an occurrence override externally, be sure to specify EXTCLS= and EXTNAME= on the TYPE=ENTRY macro for the resource type. This information will be used for checks on the occurrence override.

  **RESNAME='resource-name'**
  Names the occurrence of the resource to which the user-supplied values in the macro apply. You must enclose the resource name in quotes.
If TYPE=OCCURRENCE, the value in resource-name is treated as a wildcarded name. Thus, if RESTYPE=SPGM and RESNAME='RHDC', the scope of the override is all program names that begin with 'RHDC'.

If you do not want wildcarding to take effect -- that is, you want to limit the scope of the override to only one resource-name -- then include a blank character at the end of the resource-name. Thus, if RESTYPE=SPGM and RESNAME='TEST01 ', the scope of the override is the program 'TEST01' only.

RESTYPE=resource-type-name

Specifies the resource type you are defining in the SRTT.

Resource-type-name must be 1 to 4 characters in length and may identify a resource type defined by CA IDMS or a user-defined resource type.

The following table lists valid resource type names for CA IDMS resources:

<table>
<thead>
<tr>
<th>Global resources</th>
<th>SYSADMIN privilege</th>
<th>User</th>
<th>Group</th>
<th>User profile</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SYSA</td>
<td>USER</td>
<td>GROU</td>
<td>UPRF</td>
</tr>
<tr>
<td>System resources</td>
<td>DCADMIN privilege</td>
<td>DCA</td>
<td>SYST</td>
<td>SPRF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>System</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>System profile</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Signon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Activity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Load module</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Queue</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Access module</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Program</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database resources</td>
<td>DBADMIN</td>
<td>DB</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Database</td>
<td>DB</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Area</td>
<td>DB (AREA)(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rununit</td>
<td>DB (NRU)(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Schema (SQL)</td>
<td>DB (QSCH)(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Non-SQL schema</td>
<td>DB (NSCH)(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Access module</td>
<td>DB (DACC)(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Table</td>
<td>DB (TABL)(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DMCL</td>
<td>DMCL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Database name table</td>
<td>DBTB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) Resource type is secured when DB is secured. Note: DBADMIN privilege is secured when you activate security for DB.

- **SECBY**
  Specifies the security option for the resource type identified in the RESTYPE parameter.
- **EXTERNAL**
  Specifies that security-checking for the resource type is performed using definitions in an external security system.
  If you specify SECBY=EXTERNAL, you must include the EXTCLS and EXTNAME parameters in the macro.

- **INTERNAL**
  Specifies that security-checking for the resource type is performed using security definitions in CA IDMS.
  SECBY=INTERNAL is valid for any CA IDMS resource type (see the following table). It is not valid for a user-defined resource type.

- **OFF**
  Specifies that no security-checking is performed for the resource type; the resource type is unsecured.

- **EXTCLS=**
  Maps the CA IDMS resource type specified in the RESTYPE parameter to the resource class you have defined for this type in the external security system.
  EXTCLS is required when TYPE=ENTRY and SECBY=EXTERNAL for the entry or for any occurrence override of the entry.
  If EXTCLS is specified, the information is recorded in the SRTT but used only when security enforcement is external.

  - **resource-class-variable**
    Specifies a variable containing the name of the external resource class.

  - **resource-class-name**
    Specifies the name of the external resource class.

- **EXTNAME=**
  Using a set of predefined keywords, specifies the fields to be included in the external resource name. The order in which you specify the keywords is the order in which the fields will be included in the external resource name.
  Since EXTNAME defines the format of the resource name for external security requests, the format you specify here must match the naming conventions for the corresponding resource class in the external security system.

  - **Note:** For more information on constructing external resource names, see Using External Security (see page 35).

  EXTNAME is required when TYPE=ENTRY and SECBY=EXTERNAL for the entry or for any occurrence override of the entry.
  If EXTNAME is specified, the information is recorded in the SRTT but used only when security enforcement is external.

- **ACTivity**
  Includes in the external resource name the activity number supplied by the application.
  When formatted for an external security request, this field will be a 4- to 8-character string that is the concatenation of the following:
- Either the application name or the first 5 characters of the application name (if the full name exceeds 5 characters).

- The 3-digit activity number in displayable format.

- **APPLname**
  Includes the full application name, as supplied on the current security request, in the external resource name.

- **DBNAme**
  Includes the database name, as supplied on the current security request in the external resource name.

- **DDNAme**
  Includes the ddname, as supplied on the current security request, in the external resource name. The ddname defines the operating system library in which the program (resource type SPGM) resides.

- **ENVIr**
  Includes the environment name in the external resource name.

- **RESName**
  Includes the resource name as specified on the current security request in the external resource name.

- **RESType**
  Includes the resource type, as supplied on the RESTYPE= parameter for this SRTT entry, in the external resource name.

- **SCHema**
  Includes the schema name, as supplied on the current security request, in the external resource name. The schema name qualifies the names of SQL tables (resource type TABL) and access modules (resource types DACC and SACC).

- **SSNAme**
  Includes the subschema name, as supplied on the current security request, in the external resource name.

- **SYSTem**
  Includes the name of the CA IDMS system in the external resource name.

- **VERSion**
  Includes the version number for load modules (resource type SLOD) and non-SQL schemas (resource type NSCH), as supplied on the current security request, in the external resource name.

- **FINAL**
  Indicates the end of SRTT specifications.
  You can specify TYPE=FINAL only once. SRTT entries will be generated from the series of #SECRTT macros beginning with the one that specifies TYPE=INITIAL.
SECRTT Usage

User-Defined Resource Types

Resource-type-name can be a user-defined resource type. The valid SECBY specifications for a user-defined resource type are EXTERNAL or OFF.

The following short resource type names are reserved by CA IDMS for future use. If you specify one of these short resource type names in a #SECRTT assembly, an error message will be returned.

<table>
<thead>
<tr>
<th>Resource Type Name</th>
<th>Resource Type Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDA</td>
<td>DPAN</td>
</tr>
<tr>
<td>NSUB</td>
<td>DPGM</td>
</tr>
<tr>
<td>DMSG</td>
<td>DREC</td>
</tr>
<tr>
<td>DATT</td>
<td>DSYS</td>
</tr>
<tr>
<td>DCLA</td>
<td>DUSR</td>
</tr>
<tr>
<td>DUDE</td>
<td>DDES</td>
</tr>
<tr>
<td>DAPP</td>
<td>DLIN</td>
</tr>
<tr>
<td>DIAL</td>
<td>DLTE</td>
</tr>
<tr>
<td>DELE</td>
<td>DPTE</td>
</tr>
<tr>
<td>DFIL</td>
<td>DQUE</td>
</tr>
<tr>
<td>DLOD</td>
<td>DTSK</td>
</tr>
<tr>
<td>DMAP</td>
<td>DACT</td>
</tr>
<tr>
<td>DMOD</td>
<td></td>
</tr>
</tbody>
</table>

Order of EXTNAME Specification

The order of keywords that you specify in the EXTNAME parameter determines the order of fields in the external resource name format. For example, suppose that you specify for RESTYPE=TASK the following parameter:

```plaintext
EXTNAME=(RESTYPE, ENVIR, SYSTEM, RESNAME)
```

The external resource name format for a task will be the following:

```plaintext
TASK.environment-name.system-name.task-identifier
```

Generating the #SECRTT Macro

To assemble and link edit the #SECRTT macro, you can use the appropriate JCL or commands.

**Note:** For more information, see the Security Macro JCL (see page 310) section.

However, it is recommended that you use SMP to assemble this macro.

**Note:** For more information, see "System Modification" in the CA IDMS Installation Manual for your operating system.

Using a Single Resource Class
In the following example, each resource type is assigned to the class IDMS. In each case, the resource type is part of the external resource name format:

```
#SECR TT  RESTYPE=TASK,SECBY=EXTERNAL,  X
  EXTCLS='IDMS', EXTNAM E=(RESTYPE,RESNAME)
#SECR TT  RESTYPE=SPGM,SECBY=EXTERNAL,  X
  EXTCLS='IDMS', EXTNAM E=(RESTYPE,DDNAME,RESNAME)
#SECR TT  RESTYPE=ACTI,SECBY=EXTERNAL,  X
  EXTCLS='IDMS',EXTNAM E=(RESTYPE,RESNAME)
```

---

**SECSGOF**

**Contents**

- SECSGOF Authorization (see page 170)
- SECSGOF Syntax (see page 170)
- SECSGOF Parameters (see page 171)
- SECSGOF Usage (see page 172)

#SECSGOF signs the user off the DC system.

**SECSGOF Authorization**

Authorization requirements for use of #SECSGOF, if any, are maintained in the external security system.

**SECSGOF Syntax**

```
#SECSGOF

  RB= request-block-area-name

  .FROMLTE= lte-area-name
           lte-address

  .RGSV= ( register-number )

  .CALL= YES
         NO
         ONLY

  .CLEAR= Yes
          No

  .PLIST= SYSPLIST
          parameter-list-area-name
          parameter-list-address
```
SECSGOF Parameters

- **RB=**
  Specifies the area containing the Security Request Block (SRB) associated with this request. This area is mapped by #SECRB.
  The location of the SRB is required because most of the other parameters coded on #SECSGOF will be stored in the SRB.
  This parameter is not required if CALL=ONLY.

- **request-block-area-name**
  Identifies the symbolic name of the area containing the SRB.

- **request-block-address**
  Identifies the register containing the address of the SRB.

- **,FROMLTE=**
  Specifies the address of the Logical Terminal Element (LTE) for which signoff will be done.
  If you do not specify FROMLTE=, signoff processing will use the task's primary LTE (that is, TCELTEA).

- **lte-area-name**
  Identifies the symbolic name of a user-defined area containing the LTE address.

- **lte-address**
  Identifies the register containing the LTE address.

- **,RGSV=**
  Specifies that one or more registers are to be saved across the call. This parameter is valid in system mode only.

- **register-number**
  Specifies a register.
  *Register-number* must be a numeric literal.
  *Register-number* can be specified in the form Rn, where n is a numeric literal. This assumes that the symbol Rn has been equated to the corresponding register number; for example:

  ```
  R0   EQU   0
  R1   EQU   1
  R2   EQU   2
  .    
  .    
  R15  EQU  15
  ```

  These assignments can be made with the macro #REGEQU or explicitly coded in the program.

- **,CALL=**
  Controls the expansion of the #SECSGOF macro.
  If you omit the CALL parameter, the effect is the same as specifying CALL=YES.

- **YES**
  Causes #SECSGOF to generate both the code to complete the SRB and the code to invoke the security system.
• **NO**
  Causes #SECSGOF to generate the code to fill in the SRB fields, but not to build the parameter list or the call.

• **ONLY**
  Causes #SECSGOF to generate only the code needed to invoke the security system.

• **CLEAR=**
  Specifies whether you want the SRB to be initialized.
  If you omit the CLEAR= parameter, the effect is the same as specifying CLEAR=YES. Exception: If CALL=ONLY, the CLEAR parameter defaults to NO.

• **Yes**
  Causes #SECSGOF to clear the SRB to binary zeros before the macro expansion begins to assign values.

• **No**
  Indicates that the SRB should not be initialized.

• **PLIST=**
  Specifies the area in which to build the parameter list. The parameter list is a three-fullword storage area.

• **SYSPLIST**
  Supplies the default name for the area that contains the parameter list.

• **parameter-list-area-name**
  Identifies the symbolic name of a user-defined fullword-aligned field that contains the parameter list.

• **parameter-list-address**
  Identifies the register containing the address of the parameter list.

### SECSGOF Usage

**Substituting for RHDCSNOF**

You can issue a #SECSGOF request instead of linking to RHDCSNOF. However, only RHDCSNOF frees all resources associated with the user session.

**Copying the Security Request Block**

To issue the #SECSGOF macro, you must copy the Security Request Block. This block is mapped by the #SECRB DSECT.

⚠️ **Note:** For more information, see the documentation of #SECRB (see page 346).
#SECSGOF Return Codes

The return code for a security check is stored in register 15 and the SRBXR15 field of the SRB. The following table lists the possible return codes provided by the security system in response to a #SECSGOF macro:

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Request was successful</td>
</tr>
<tr>
<td>12</td>
<td>Interface/parameter list error</td>
</tr>
</tbody>
</table>

### SECSONGON

**Contents**

- SECSONGON Authorization (see page 173)
- SECSONGON Syntax (see page 173)
- SECSONGON Parameters (see page 174)
- SECSONGON Usage (see page 177)

#SECSGON initiates a signon to the CA IDMS system and attempts to authenticate the user and validate the password.

### SECSONGON Authorization

Authorization requirements for use of #SECSGON, if any, are maintained in the external security system.

### SECSONGON Syntax

```plaintext
#SECSGON

RB= request-block-area-name
    request-block-address

.USERID= user-identifier-area-name
    user-identifier-address
    'user-identifier'

.PASSWRD= password-area-name
    password-address
    'password'

.NEWPASS= new-password-area-name
    new-password-address
    'new-password'

.PASSTYP= CLEAR
        ENCRYPT

.SUPOFFM= YES
        NO

.PASSCHK= YES
```
SECSGON Parameters

- **RB=**
  Specifies the area containing the Security Request Block (SRB) associated with this request. This area is mapped by #SECRB. The location of the SRB is required because most of the other parameters coded on #SECSGON will be stored in the SRB. This parameter is not required if CALL=ONLY.

- **request-block-area-name**
  Identifies the symbolic name of the area containing the SRB.

- **request-block-address**
  Identifies the register containing the address of the SRB.

- **USERID=**
  Specifies the area containing the user ID being signed on. The user ID must be 1 to 18 characters. If less than 18 characters, it must be left-justified and padded with spaces or binary zeros.

- **user-identifier-area-name**
  Identifies the symbolic name of the area containing the user ID.

- **user-identifier-address**
  Identifies the register containing the address of the user ID.

- **'user-identifier'**
  Supplies the user ID as a literal character string.

- **PASSWRD=**
  Specifies the area containing the password. The password must be one to eight characters. If less than eight characters, it must be left-justified and padded with spaces or binary zeros. PASSWRD is ignored if PASSCHK=NO is specified.
- **password-area-name**
  Identifies the symbolic name of the area containing the password.

- **password-address**
  Identifies the register containing the address of the password.

- **'password'**
  Supplies the password as a literal character string.

- **,NEWPASS=**
  Specifies the area containing the new password.
  The password must be one to eight characters. If less than eight characters, it must be left-justified and padded with spaces or binary zeros.
  NEWPASS is ignored if PASSCHK=NO is specified.

- **new-password-area-name**
  Identifies the symbolic name of the area containing the password.

- **new-password-address**
  Identifies the register containing the address of the password.

- **'new-password'**
  Supplies the password as a literal character string.

- **,PASSTYP=**
  Specifies whether the value supplied for PASSWRD is encrypted or in clear text.
  If you omit PASSTYP and specify CLEAR=YES, the effect is the same as coding PASSTYP=CLEAR.
  If you omit PASSTYP and specify CLEAR=NO, PASSTYP retains its prior setting.

- **CLEAR**
  Specifies that the password is not encrypted. CA IDMS will encrypt the password if signon security is internal.

- **ENCRYPT**
  Specifies that the password supplied on this request is already encrypted.

- **,SUPOFFM=**
  Specifies whether a message should be issued if a user is currently signed on to the terminal associated with this request and must be signed off.
  If you omit SUPOFFM and specify CLEAR=YES, the effect is the same as specifying SUPOFFM=NO.
  If you omit SUPOFFM and specify CLEAR=NO, SUPOFFM retains its prior setting.

- **YES**
  Specifies that the signoff message should be suppressed.

- **NO**
  Specifies that the signoff message should be issued.
• PASSCHK=
  Indicates whether password validation should be performed as part of this signon.
  Password validation may be bypassed only if the requestor is in system mode. If a user-mode
  program issues a #SECSGON with PASSCHK=NO, the request will be rejected as invalid (return
  code 12). If you omit PASSCHK and specify CLEAR=YES, the effect is the same as specifying
  PASSCHK=YES. If you omit PASSCHK and specify CLEAR=NO, PASSCHK retains its prior setting.

• YES
  Specifies that password validation should be performed as part of this request.

• NO
  Specifies that no password validation should be performed as part of this request.

• TOLTE=
  Specifies the address of the Logical Terminal Element (LTE) for which signon will be done.
  If you do not specify TOLTE=, signon processing will use the task's primary LTE (that is, TCELTEA).

• lte-area-name
  Identifies the symbolic name of a user-defined area containing the LTE address.

• lte-address
  Identifies the register containing the LTE address.

• RGSV=
  Specifies that one or more registers are to be saved across the call. This parameter is valid in
  system mode only.

• register-number
  Specifies a register.
  Register-number must be a numeric literal.
  Register-number can be specified in the form Rn, where n is a numeric literal. This assumes that
  the symbol Rn has been equated to the corresponding register number; for example:

  ```
  R0   EQU   0
  R1   EQU   1
  R2   EQU   2
  .
  .
  R15  EQU  15
  ```

  These assignments can be made with the macro #REGEQU or explicitly coded in the program.

• CALL=
  Controls the expansion of the #SECSGON macro.
  If you omit the CALL parameter, the effect is the same as specifying CALL=YES.

• YES
  Causes #SECSGON to generate both the code to complete the SRB and the code to invoke the
  security system.

• NO
  Causes #SECSGON to generate the code to fill in the SRB fields, but not to build the parameter list
  or the call.
ONLY
Causes #SECSGON to generate only the code needed to invoke the security system.

,CLEAR=
Specifies whether you want the SRB to be initialized.
If you omit the CLEAR= parameter, the effect is the same as specifying CLEAR=YES. Exception: If CALL=ONLY, the CLEAR parameter defaults to NO.

Yes
Causes #SECSGON to clear the SRB to binary zeros before the macro expansion begins to assign values.

No
Indicates that the SRB should not be initialized.

,PLIST=
Specifies the area in which to build the parameter list. The parameter list is a three-fullword storage area.

SYSPLIST
Supplies the default name for the area that contains the parameter list.

parameter-list-area-name
Identifies the symbolic name of a user-defined fullword-aligned field that contains the parameter list.

parameter-list-address
Identifies the register containing the address of the parameter list.

SECSGON Usage

Substituting for RHDCSNON

All processing performed by RHDCSNON, including user profile processing, is also performed by #SECSGON, except:

- Signon CLIST processing
- Issuing the message DC258003, USER IS SIGNED ON

Copying the Security Request Block

To issue the #SECSGON macro, you must copy the Security Request Block. This block is mapped by the #SECRB DSECT.

Note: For more information, see the documentation of #SECRB (see page 346).

#SECSGON Return Codes
The return code for a signon request is stored in register 15 and the SRBXR15 field of the SRB. The following table lists the possible return codes provided by the security system in response to a #SECSGON macro:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Request was successful; access allowed</td>
</tr>
<tr>
<td>04</td>
<td>User identifier not found</td>
</tr>
<tr>
<td>08</td>
<td>User not authorized; access denied</td>
</tr>
<tr>
<td>12</td>
<td>Interface/parameter list error</td>
</tr>
<tr>
<td>16</td>
<td>Password missing or invalid</td>
</tr>
<tr>
<td>20</td>
<td>Password has expired (this condition can result only if signon is externally secured)</td>
</tr>
</tbody>
</table>

**Note:** If MULTIPLE signon is enabled, and a user attempts to change the password while that user is signed on to another terminal, SRBXR15 will return a value of 8 (ACCESS DENIED) but signon will complete successfully. SRB field SRBX0 contains reason code SRBXNPPW (password cannot be changed).

**Notes on Security Statement Syntax**

**What is Authorization**

You can execute a security statement if you have the required authorization. The required authorization is expressed in terms of the privilege or privileges a user must hold. This appears immediately preceding the syntax diagram for each security statement documented in the following sections.

**Holders of SYSADMIN**

A user who holds SYSADMIN privilege is authorized to execute all security statements. SYSADMIN is identified as the required authorization in documentation of security statement syntax when it is the only privilege that qualifies the user to execute the statement.

For more information, see the following topics:

- Resource Identifiers (see page 178)
- Expansion of Table-Name (see page 180)
- Expansion of Authorization Identifier (see page 181)
- Syntactic Limits (see page 182)

**Resource Identifiers**

**Contents**

- Forming Identifiers (see page 179)
Delimited Identifiers (see page 180)

What is an Identifier

Identifiers are the smallest lexical units used to name resources in the CA IDMS environment.

The following are examples of identifiers:

- Area-name
- User-identifier
- Task-identifier

Qualifying Identifiers

Identifiers for some resources may need to be qualified by other identifiers. For example, an area name is always qualified by the name of the segment with which it is associated:

segment-name.area-name

Syntax diagrams in the sections that follow indicate when an identifier may be qualified and whether the qualifier is required or optional.

Forming Identifiers

Valid Characters

An identifier consists of a combination of the following:

- Letters (A through Z and a through z) -- See Delimited Identifiers (see page 180) for information on the significance of lowercase and uppercase letters
- Digits (0 through 9)
- At sign (@)
- Dollar sign ($)
- Pound sign (#)
- Underscore (_)

The first character of an identifier must be a letter, @, $, or #.

Maximum Length

The maximum length of a given identifier is presented in Syntactic Limits (see page 182).
Delimited Identifiers

Why Delimit Identifiers

You delimit an identifier in double quotation marks to do the following:

- **Allow the use of special characters and blanks.** An identifier enclosed in quotation marks can consist of any combination of characters. For example, this is a valid identifier:
  
  "&ATM+F(0517). MA"

  To include a double quotation mark as part of the identifier itself, use two consecutive double quotation marks. For example:
  
  "M1K""L96ZZ".

- **Make case significant.** When you enclose an identifier in quotation marks, CA IDMS does not convert lowercase letters to uppercase. Lowercase letters in quotation marks are not equal to uppercase letters or to lowercase letters that are not in quotation marks. In the following example, the identifiers on the left all identify the same table; the identifier on the right identifies a different table:

  | employee | "employee" |
  | EMPLOYEE | "EMPLOYEE" |

Placement of Quotation Marks

If one or more parts of a qualified identifier require quotation marks, place the quotation marks only around the individual parts. Do not include two identifiers in one set of quotation marks. For example, both parts of the following qualified identifier require quotation marks:

"temp-tab-1"."Commission to Date"

When you calculate the length of an identifier, do not include delimiting quotation marks.

⚠️ **Note:** The delimiter rules also apply to passwords using special characters or blanks even though passwords are not delimiters.

Expansion of Table-Name

**Contents**

- Expansion of Table-Name Syntax (see page 181)
- Expansion of Table-Name Parameters (see page 181)

Represents a qualified or unqualified table, view, function, procedure, or table procedure identifier in an SQL statement.
Expansion of Table-Name Syntax

Expansion of table-name

- **schema-name**
  Specifies the schema with which the table, view, procedure or table procedure identified by table-identifier, view-identifier, procedure-identifier, or table-procedure-identifier is associated.

- **table-identifier**
  Identifies either a base table defined in the dictionary or a temporary table defined during the current transaction.

- **view-identifier**
  Identifies a view defined in the dictionary.

- **procedure-identifier**
  Identifies a procedure defined in the dictionary.

- **function-identifier**
  Identifies a function defined in the dictionary.

- **table-procedure-identifier**
  Identifies a table procedure defined in the dictionary.

Expansion of Table-Name Parameters

- **schema-name**
- **table-identifier**
- **view-identifier**
- **procedure-identifier**
- **function-identifier**
- **table-procedure-identifier**

Expansion of Authorization Identifier

Contents

- Expansion of Authorization Identifier Syntax (see page 181)
- Expansion of Authorization Identifier Parameters (see page 182)
- Expansion of Authorization Identifier Usage (see page 182)

Represents a user identifier or a group identifier in an authorization statement.

Expansion of Authorization Identifier Syntax
Expansion of Authorization Identifier Parameters

- **user-identifier**
  Identifies a user.

- **group-identifier**
  Identifies a group.

Expansion of Authorization Identifier Usage

*User Catalog Definition*

The CREATE/ALTER/DROP statements for USER manipulate the definitions of *user-identifier* in the user catalog. Similarly, the CREATE/ALTER/DROP statements for GROUP manipulate the definitions of *group-identifier* in the user catalog.

When you specify **authorization-identifier** in any other security statement (or in the ADD USER parameter of CREATE/ALTER GROUP), *authorization-identifier* must be defined in the user catalog if either of these is true:

- The identifier is a *user-identifier* and security on the specified resource is enforced by CA IDMS internal security.
- The identifier is a *group-identifier*.

If authentication of users is handled by an external security system, *user-identifier* need not be defined in the user catalog.

*Authorizing a User to Update a Table*

In the following GRANT statement, the authorization identifier is the user identifier RES:

```
grant update
  on table employee
  to res;
```

*Revoking Execution Privileges from a Group*

In the following GRANT statement, the authorization identifier is the group identifier ACCT_GRP_1:

```
revoke execute
  on category emp_update
  from acct_grp_1;
```

Syntactic Limits

This table lists the maximum size or value, as applicable, of parameters you use in security syntax statements:
### Syntax for Securing Global Resources

This section explains the syntax, parameters, command codes, examples, and usage of the following resources:

- ALTER GROUP (see page 183)
- ALTER USER (see page 185)
- ALTER USER PROFILE (see page 187)
- CREATE GROUP (see page 190)
- CREATE USER (see page 191)
- CREATE USER PROFILE (see page 194)
- DROP GROUP (see page 196)
- DROP USER (see page 198)
- DROP USER PROFILE (see page 200)
- GRANT Administration Privilege (see page 201)
- GRANT Definition Privileges (see page 203)
- REVOKE Administration Privilege (see page 207)
- REVOKE Definition Privileges (see page 208)

### ALTER GROUP

Modifies the definition of a group by adding users, dropping users, or changing the description of the group.
ALTER GROUP Authorization

To issue an ALTER GROUP statement, you must hold one of the following privileges:

- SYSADMIN
- ALTER privilege on the group

ALTER GROUP Syntax

```
ALTER GROUP group-identifier

DESCRIPTION 'description'

ADD DROP USER user-identifier
```

ALTER GROUP Parameters

- `group-identifier`
  Identifies the group to be modified.
  `Group-identifier` must be a group that has been defined in the user catalog with the CREATE GROUP statement.

- `DESCRIPTION 'description'`
  Supplies a description of `group-identifier`.
  Description can be at most 40 characters in length.

- `ADD`
  Adds the specified users to `group-identifier`.

- `DROP`
  Drops the specified users from `group-identifier`.

- `USER user-identifier`
  Identifies the user to be added to or dropped from `group-identifier`.
  `User-identifier` must be a user that has been defined in the user catalog with the CREATE USER statement.
ALTER GROUP Usage

No Nesting of Groups
You cannot add a group to another group. You can add only users to a group.

Adding Users to a Group
The following ALTER GROUP statement adds three users to the hr_corp group:

```
alter group hr_corp
   add user sam, flo, guy;
```

Dropping a User from a Group
In the following ALTER GROUP statement, user sue is dropped from the hr_corp group:

```
alter group hr_corp
   drop user sue;
```

ALTER USER

Contents

- ALTER USER Authorization (see page 185)
- ALTER USER Syntax (see page 185)
- ALTER USER Parameters (see page 186)
- ALTER USER Usage (see page 187)
- ALTER USER More Information (see page 187)

Modifies the definition of a user in the user catalog.

ALTER USER Authorization
To issue an ALTER USER statement, you must hold one of the following privileges:

- SYSADMIN
- ALTER privilege on the user

ALTER USER Syntax

```
ALTER USER user-identifier

   DESCRIPTION 'description'

   GROUP PUBLIC
```

ALTER USER Parameters

- **user-identifier**
  Identifies the user to be modified.
  *User-identifier* must be a user that has been defined in the user catalog with the CREATE USER statement.

- **DESCRIPTION 'description'**
  Supplies a description of the user.
  *Description* can be at most 40 characters in length.

- **GROUP**
  Specifies the default group for *user-identifier*.

- **PUBLIC**
  Identifies the group PUBLIC.
  All users automatically belong to group PUBLIC.

- **group-identifier**
  Identifies a group.
  *Group-identifier* must be a group that has been defined in the user catalog with the CREATE GROUP statement.

- **NAME 'user-name'**
  Supplies the name associated with *user-identifier*.
  *User-name* can be at most 32 characters in length. This parameter is available for documentation; its value is not used in CA IDMS security processing.

- **PASSWORD password**
  Supplies the password associated with *user-identifier*.
  *Password* can be at most eight characters in length.

- **NULL**
  Removes the password defined for *user-identifier* in the user catalog.

- **PROFILE**
  Specifies the user profile associated with the *user-identifier*.

- **profile-name**
  Identifies a user profile.
- **NULL**
  Removes the profile associated with *user-identifier*.
  If NULL is specified, no user profile attributes will be established when the user executes CA IDMS software.

**ALTER USER Usage**

*Null Password*

If *password* is NULL and CA IDMS internal security is used to validate users, the user will be able to sign on to a system without supplying a password.

*Changing the Default Group and User Profile*

The following ALTER USER statement specifies a new default group and user profile for user sue:

```
alter user sue
  group hr_corp
  profile hr_prof;
```

**ALTER USER More Information**

- For more information about users, see Create User (see page 191) and DROP USER (see page 198).
- For more information about user profiles, see CREATE USER PROFILE (see page 194).
- For more information about the ALTER privilege, see GRANT Definition Privileges (see page 203).

**ALTER USER PROFILE**

*Contents*

- ALTER USER PROFILE Authorization (see page 187)
- ALTER USER PROFILE Syntax (see page 188)
- ALTER USER PROFILE Parameters (see page 188)
- ALTER USER PROFILE Usage (see page 189)
- ALTER USER PROFILE More Information (see page 190)

Modifies the definition of a user profile in the user catalog.

**ALTER USER PROFILE Authorization**

To issue an ALTER USER PROFILE statement, you must hold one of the following privileges:
ALTER USER PROFILE Syntax

```
ALTER USER PROFILE profile-name

ATTRIBUTES attribute-specification

attribute-keyword = 'prefix &USER &GROUP &SYSTEM suffix'

attribute-value

NULL

OVERRIDE YES NO
```

ALTER USER PROFILE Parameters

- **profile-name**
  Identifies the profile to be modified.
  *Profile-name* must be a profile that has been defined in the user catalog with the CREATE USER PROFILE statement.

- **ATTRIBUTES attribute-specification**
  Supplies one or more attributes of the profile.
  An attribute is a keyword and an associated value for the keyword.

- **attribute-keyword =**
  Specifies the attribute keyword whose value is to be added, replaced, or removed.
  An identifier of not more than eight characters can be used as an attribute keyword.

- **&USER**
  Supplies the attribute value. It is a substitution parameter.
  The value of &USER is equal to the user ID of the user on whose behalf the user profile is invoked.

- **&GROUP**
  Represents the current group. It is a substitution parameter. The value of &GROUP is equal to the name of the default group for the current user.
- **&SYSTEM**
  Supplies the attribute value. It is a substitution parameter.
  The value of &SYSTEM is equal to the SYSTEM ID value for the current system in the SYSTEM statement of system generation, or 'BATCH' in the case of local mode execution.

- **prefix**
  Supplies a prefix for the value in the substitution parameter.

- **suffix**
  Supplies a suffix for the value in the substitution parameter.

- **attribute-value**
  Provides the value portion of the attribute specification.
  Attribute-value may be at most 32 characters in length and must be enclosed in single quotation marks if it contains embedded blanks or special characters other than @, $, and #.

- **NULL**
  Removes the attribute from profile-name.

- **OVERRIDE**
  Specifies whether the user can modify the attribute specification with a DCUF SET PROFILE command.
  YES allows the user to override the attribute specification. NO prevents the user from overriding the attribute specification.
  If OVERRIDE is not specified and the attribute keyword already exists in the profile, the OVERRIDE value remains the same. If OVERRIDE is not specified and the attribute keyword does not exist in the profile, the OVERRIDE value defaults to YES.

**ALTER USER PROFILE Usage**

**Nesting Profiles**

The attribute keyword INCLUDE specifies that attribute-value identifies another profile. The attributes of the included, or nested, profile are added to those of the current profile.

Up to 10 levels of nesting are supported.

**Substitution Parameters**

The value of a substitution parameter in attribute-specification:

- Must not exceed 32 characters, including a prefix and suffix
- Must not contain special characters other than @, $, and #

The following ALTER USER PROFILE statement defines two attributes in an existing profile:

```
alter user profile corp_dev
  attributes
depth='5621',
prtdest='wwuav';
```
ALTER USER PROFILE More Information

- For more information about creating a user profile, see CREATE USER PROFILE (see page 194).
- For more information about dropping a user profile, see DROP USER PROFILE (see page 200).
- For more information about attributes, see section "System Profiles" in the CA IDMS System Tasks and Operator Reference section.

CREATE GROUP

Contents

- CREATE GROUP Authorization (see page 190)
- CREATE GROUP Syntax (see page 190)
- CREATE GROUP Parameters (see page 190)
- CREATE GROUP Usage (see page 191)
- CREATE GROUP More Information (see page 191)

Creates the definition of a group of users.

CREATE GROUP Authorization

To issue a CREATE GROUP statement, you must hold one of the following privileges:

- SYSADMIN
- CREATE privilege on the group

CREATE GROUP Syntax

```
CREATE GROUP group-identifier
  DESCRIPTION 'description'
  ADD USER user-identifier
```

CREATE GROUP Parameters

- `group-identifier`
  Identifies the group to be created.
  `Group-identifier` can be at most 18 characters in length. It must be unique among the set of authorization identifiers (users and groups) in the user catalog.
CREATE GROUP Usage

No Nesting of Groups

You cannot add a group to another group. You can only add users to a group.

This statement creates a group of users for data communications administrators:

```plaintext
create group dca_group
description 'Authorization group for DCAs'
add user dca_s15, dca_sys16, dca_sys17, dca_sys18, dca_sys19;
```

CREATE GROUP More Information

- For more information on groups, see ALTER GROUP (see page 183) and DROP GROUP (see page 196).
- For more information on creating a user, see CREATE USER (see page 191).
- For more information on the CREATE privilege, see GRANT Definition Privileges (see page 203).

CREATE USER

Contents

- CREATE USER Authorization (see page 192)
- CREATE USER Syntax (see page 192)
- CREATE USER Parameters (see page 192)
- CREATE USER Usage (see page 193)
CREATE USER More Information (see page 194)

Creates the definition of a user in the user catalog.

CREATE USER Authorization

To issue a CREATE USER statement, you must hold one of the following privileges:

- SYSADMIN
- CREATE privilege on the user

CREATE USER Syntax

```
CREATE USER user-identifier
  DESCRIPTION 'description'
  GROUP PUBLIC group-identifier
  NAME 'user-name'
  PASSWORD password NULL
  PROFILE profile-name NULL
```

CREATE USER Parameters

- **user-identifier**
  Identifies the user to be created. *User-identifier* can be at most 18 characters in length. It must be unique among the set of authorization identifiers (users and groups) in the user catalog.

  **Note:** For more information on identifiers, see Notes on Security Statement Syntax (see page 178).

- **DESCRIPTION 'description'**
  Supplies a description of the user. *Description* can be at most 40 characters in length. This parameter is available for documentation; its value is not used in CA IDMS security processing.

- **GROUP**
  Specifies the default group of *user-identifier*. 
- **PUBLIC**
  Identifies the group PUBLIC.

- **group-identifier**
  Identifies a group.
  If group-identifier has been defined in the user catalog, the user is added to the group. If group-identifier has not been defined in the user catalog, an error message is issued and the user is not added to the group.

  **Note:** For more information on identifiers, see Notes on Security Statement Syntax (see page 178).

- **NAME 'user-name'**
  Supplies a name to be associated with user-identifier in the user catalog. User-name can be at most 32 characters in length. This parameter is available for documentation; its value is not used in CA IDMS security processing.

- **PASSWORD password**
  Supplies a password for user-identifier. Password may be at most eight characters in length.

- **NULL**
  Specifies that no password is defined for user-identifier in the user catalog.

- **PROFILE**
  Specifies a user profile to be associated with user-identifier.

- **profile-name**
  Identifies a user profile. Profile-name must be a user profile defined in the user catalog with the CREATE USER PROFILE statement.

- **NULL**
  Specifies that no profile is associated with user-identifier.
  If NULL is specified, no user profile attributes will be established when the user executes CA IDMS software.

---

**CREATE USER Usage**

**Null Password**

If password is NULL and CA IDMS internal security is used to validate users, the user will be able to sign on to a system without supplying a password.

**Creating a Named User**

This statement creates a user ID for an individual:
create user tim
  name 'Thomas McNall'
  password qwerty
  profile dev_prof;

Creating a Generic User

This statement creates a user ID for a given role:

create user dca s18
  description 'Administrator for System 18'
  group dca_group
  password s18pass;

CREATE USER More Information

- For more information on users, see ALTER USER (see page 185) and DROP USER (see page 198).
- For more information on user profiles, see CREATE USER PROFILE (see page 194).
- For more information on the CREATE privilege, see GRANT Definition Privileges (see page 203).

CREATE USER PROFILE

Contents

- CREATE USER PROFILE Authorization (see page 194)
- CREATE USER PROFILE Syntax (see page 194)
- CREATE USER PROFILE Parameters (see page 195)
- CREATE USER PROFILE Usage (see page 196)
- CREATE USER PROFILE More Information (see page 196)

Creates the definition of a user profile in the user catalog.

CREATE USER PROFILE Authorization

To issue a CREATE USER PROFILE statement, you must hold one of the following privileges:

- SYSADMIN

- CREATE privilege on the user profile

CREATE USER PROFILE Syntax

CREATE USER PROFILE profile-name

            ATTRIBUTEs attribute-specification
CREATE USER PROFILE Parameters

- **profile-name**
  Identifies the profile to be created. *Profile-name* can be at most 18 characters in length.

- **ATTRIBUTEs attribute-specification**
  Specifies one or more attributes of the profile. An attribute is a keyword and an associated value for the keyword.

- **attribute-keyword**
  Specifies the attribute keyword. An identifier of not more than eight characters can be used as an attribute keyword.

  **Note:** Certain keywords have special significance to the CA IDMS runtime environment. For a list of these keywords and discussion of their meaning, see "System Profiles" in the CA IDMS System Tasks and Operator Reference section.

- **&USER**
  Is a substitution parameter that supplies the attribute value. The value of &USER is equal to the user ID of the user on whose behalf the user profile is invoked.

- **&GROUP**
  Is a substitution parameter representing the current group. The value of &GROUP is equal to the name of the default group for the current user.

- **&SYSTEM**
  Is a substitution parameter that supplies the attribute value. The value of &SYSTEM is equal to the SYSTEM ID value for the current system in the SYSTEM statement of system generation, or 'BATCH' in the case of local mode execution.

- **prefix**
  Supplies a prefix for the value in the substitution parameter.

- **suffix**
  Supplies a suffix for the value in the substitution parameter.

- **attribute-value**
  Supplies the value portion of the attribute specification. *Attribute-value* may be at most 32 characters in length and must be enclosed in single quotation marks if it contains embedded blanks or special characters other than @, $, and #.
• OVERRIDE
  Specifies whether the user can modify the attribute specification with a DCUF SET PROFILE command.
  YES allows the user to override the attribute specification. NO prevents the user from overriding the attribute specification.

CREATE USER PROFILE Usage

Nesting Profiles

The attribute keyword INCLUDE specifies that attribute-value identifies another profile. The attributes of the included, or nested, profile are added to those of the current profile.

Up to 10 levels of nesting are supported.

Substitution Parameters

The value of a substitution parameter in attribute-specification:

• Must not exceed 32 characters, including a prefix and suffix
• Must not contain special characters other than @, $, and #

The following CREATE USER PROFILE statement creates a profile with two attributes:

create user profile corp_dev
attributes
dept='1237',
prtdest='gdnc03';

CREATE USER PROFILE More Information

• For more information on altering a user profile, see ALTER USER PROFILE (see page 187).
• For more information on dropping a user profile, see DROP USER PROFILE (see page 200).
• For more information on attributes, see section "System Profiles" in the CA IDMS System Tasks and Operator Reference section.

DROP GROUP

Contents

• DROP GROUP Authorization (see page 197)
• DROP GROUP Syntax (see page 197)
• DROP GROUP Parameters (see page 197)
• DROP GROUP Usage (see page 197)
• DROP GROUP More Information (see page 198)

Deletes the definition of a group from the user catalog.

DROP GROUP Authorization

To issue a DROP GROUP statement, you must hold one of the following privileges:

• SYSADMIN

• DROP privilege on the group

DROP GROUP Syntax

DROP GROUP group-identifier

DROP GROUP Parameters

• group-identifier

Identifies the group to be dropped.

Group-identifier must be a group that has been defined in the user catalog with the CREATE GROUP statement.

DROP GROUP Usage

Group PUBLIC

You cannot drop the group PUBLIC.

Implicitly Revoking Privileges

When you drop a group, you automatically revoke all privileges that have been granted to the group. Thus, a user who was in the dropped group no longer holds privileges received as a result of membership in the group.

Issuing DROP GROUP Before and After SDEL Execution

The first time you issue a DROP GROUP statement for a group identifier, the identifier is flagged for logical deletion. To delete all privileges associated with each logically deleted group, you execute the SDEL task in each system of the domain and against each dictionary in the system that contains security definitions. (The system dictionary and each application dictionary with an SQL catalog component contain security definitions.) Then you reissue the DROP GROUP statement to physically delete the group from the user catalog.
**Note:** If the SDEL task has been defined as an autotask and is invoked at startup, all dictionaries are processed if a logically deleted group is found. If SDEL is invoked manually, each dictionary must be processed separately.

For more information on the SDEL task, see the *CA IDMS System Tasks and Operator Reference section*.

If you physically delete a group before running SDEL on all systems, follow the following steps:

1. Create the group again.
2. Drop the group once.
3. Run SDEL on all systems and dictionaries.
4. Drop the group a second time.

The following DROP GROUP statement drops the definition of the corp_admin group and implicitly revokes all privileges that have been granted to the group:

```sql
drop group corp_admin;
```

**DROP GROUP More Information**

- For more information on creating and altering a group, see CREATE GROUP (see page 190) and Alter Group.

- For more information on granting and revoking privileges, see descriptions of GRANT and REVOKE statements in the following sections:
  - GRANT Definition Privileges (see page 203)
  - REVOKE Definition Privileges (see page 208)
  - Syntax for Securing System Resources (see page 211)
  - Syntax for Securing Database Resources (see page 238)

**DROP USER**

**Contents**

- DROP USER Authorization (see page 199)
- DROP USER Syntax (see page 199)
- DROP USER Parameters (see page 199)
- DROP USER Usage (see page 199)
- DROP USER More Information (see page 200)
Deletes the definition of a user from the user catalog and drops the user from all groups of which the user is a member.

To complete a physical deletion from the user catalog, you must issue the DROP USER statement two times, as described in the "Usage" section following the parameter descriptions.

**DROP USER Authorization**

To issue a DROP USER statement, you must hold one of the following privileges:

- SYSADMIN
- DROP privilege on the user

**DROP USER Syntax**

```
DROP USER user-identifier
```

**DROP USER Parameters**

- `user-identifier`
  Identifies the user to be dropped.

  *User-identifier* must be a user that has been defined in the user catalog with the CREATE USER statement.

**DROP USER Usage**

*Issuing DROP USER Before and After SDEL Execution*

The first time you issue a DROP USER statement for a user identifier, the identifier is flagged for logical deletion. To delete all privileges associated with each logically deleted user, you execute the SDEL task in each system of the domain and against each dictionary in the system that contains security definitions. (The system dictionary and each application dictionary with an SQL catalog component contain security definitions.) Then you reissue the DROP USER statement to physically delete the user from the user catalog.

⚠️ **Note:** If the SDEL task has been defined as an autotask and is invoked at startup, all dictionaries are processed if a logically deleted user is found. If SDEL is invoked manually, each dictionary must be processed separately.

For more information on the SDEL task, see the *CA IDMS System Tasks and Operator Reference section*.
If you physically delete a user before running SDEL on all systems, follow these steps:

1. Create the user again.
2. Drop the user once.
3. Run SDEL on all systems and dictionaries.
4. Drop the user a second time.

*Implicitly Revoking Privileges*

When you drop a user, you implicitly revoke all privileges that have been granted to the user.

The following DROP USER statement removes the definition of user sue from the user catalog and removes user sue from any group to which user sue was assigned:

drop user sue;

**DROP USER More Information**

- For more information on creating and altering a user, see CREATE USER (see page 191) and ALTER USER.
- For more information on granting and revoking privileges, see descriptions of GRANT and REVOKE statements in the following sections:
  - GRANT Definition Privileges (see page 203)
  - REVOKE Definition Privileges (see page 208)
  - Syntax for Securing System Resources (see page 211)
  - Syntax for Securing Database Resources (see page 238)

**DROP USER PROFILE**

**Contents**

- DROP USER PROFILE Authorization (see page 201)
- DROP USER PROFILE Syntax (see page 201)
- DROP USER PROFILE Parameters (see page 201)
- DROP USER PROFILE Usage (see page 201)
- DROP USER PROFILE More Information (see page 201)

Deletes the definition of a user profile from the user catalog.
DROP USER PROFILE Authorization

To issue a DROP USER PROFILE statement, you must hold one of the following privileges:

- SYSADMIN
- DROP privilege on the user profile

DROP USER PROFILE Syntax

```sql
DROP USER PROFILE profile-name
```

DROP USER PROFILE Parameters

- `profile-name` Identifies the profile to be dropped.
  `Profile-name` must be a profile that has been defined in the user catalog with the CREATE USER PROFILE statement.

DROP USER PROFILE Usage

Users Associated With Profiles That are Dropped

If you drop a profile specified in a user definition, no user profile attributes will be associated with the user session.

The following DROP USER PROFILE statement removes the definition of the hr_prof profile from the user catalog:
```
drop user profile hr_prof;
```

DROP USER PROFILE More Information

- For more information about creating and altering a user profile, see CREATE USER PROFILE (see page 194) and ALTER USER PROFILE.
- For more information about defining users, see CREATE USER (see page 191) and ALTER USER.

GRANT Administration Privilege

Contents

- GRANT Administration Privilege Authorization (see page 202)
• GRANT Administration Privilege Syntax (see page 202)
• GRANT Administration Privilege Parameters (see page 202)
• GRANT Administration Privilege Usage (see page 202)
• GRANT Administration Privilege More Information (see page 203)

Gives one or more users SYSADMIN privilege.

GRANT Administration Privilege Authorization

To grant SYSADMIN privilege, you must hold SYSADMIN privilege.

GRANT Administration Privilege Syntax

►►─── GRANT SYSADMIN ─────────────────────────────────────────────────────────►

┌─────────────── , ───────────────┐
►─── TO ─▼─┬─ PUBLIC ────────────────────┬─┴─────────────────────────────────►◄

GRANT Administration Privilege Parameters

• SYSADMIN
  Specifies that you are giving SYSADMIN privilege to the users or groups identified in the TO parameter.

• TO
  Specifies the users or groups to whom you are giving SYSADMIN privilege.

• PUBLIC
  Specifies all users.

Important: If you grant SYSADMIN to PUBLIC, any user can administer the security system.

• authorization-identifier
  Identifies a user or group.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

GRANT Administration Privilege Usage

Ultimate Authority to Grant and Revoke
SYSADMIN controls access to all resources. A user with SYSADMIN privilege can grant and revoke any privilege on any resource in the domain.

**Caution in Granting SYSADMIN**

A user to whom you grant SYSADMIN privilege can administer the security system. If you grant SYSADMIN to PUBLIC, any user can administer the security system.

⚠️ **Note:** If resource type SYSA is unsecured in the SRTT, any user can administer the security system.

You can decentralize security administration by granting DCADMIN and DBADMIN privileges to users.

⚠️ **Note:** For more information on the DCADMIN and DBADMIN administration privileges, see the following sections:

- Securing Database Resources (see page 88)
- Syntax for Securing Database Resources (see page 238)

**Granting SYSADMIN to the Security Administrator**

The following GRANT statement gives SYSADMIN privilege to group secadmin:

```
grant sysadmin
to secadmin;
```

**GRANT Administration Privilege More Information**

For more information about revoking SYSADMIN privilege, see REVOKE Administration Privilege (see page 207).

**GRANT Definition Privileges**

**Contents**

- GRANT Definition Privileges Authorization (see page 204)
- GRANT Definition Privileges Syntax (see page 204)
- GRANT Definition Privileges Parameters (see page 204)
- GRANT Definition Privileges Usage (see page 206)
- GRANT Definition Privileges More Information (see page 206)
Gives one or more users or groups the privilege of performing definition functions on users, groups, or user profiles in the user catalog.

**GRANT Definition Privileges Authorization**

To grant a definition privilege on a user catalog resource, you must hold one of the following privileges:

- The corresponding grantable privilege (you can grant the privilege, but you cannot specify WITH GRANT OPTION)
- SYSADMIN privilege

**GRANT Definition Privileges Syntax**

```
GRANT Definition Privileges Syntax

►►─── GRANT ─┬─ DEFINE ───────────┬────────────────────────────────────────────►
│ ┌────── , ───────┐ │
└─▼─┬─ ALTER ──┬─▼─┘
├─ CREATE ─┤
├─ DISPLAY ─┤
└─ DROP ─┘

►─── ON ─┬─ GROUP ──┬─ PUBLIC ──┬─ authorization-identifier ─┘
├─ USER ──┤
│ USER PROFILE profile-name ┘

►─┬─────────────────────┬────────────────────────────────────────────────────►◄
└─ WITH GRANT OPTION ┘
```

**GRANT Definition Privileges Parameters**

- **DEFINE**
  Gives the ALTER, CREATE, and DROP privileges on the resource identified in the ON parameter to the users or groups identified in the TO parameter.

- **ALTER**
  Gives the ALTER privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The ALTER privilege on a resource allows the user to modify the resource definition.

- **CREATE**
  Gives the CREATE privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The CREATE privilege on a resource allows the user to define the resource.
DISPLAY
Gives the DISPLAY privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The DISPLAY privilege allows the user to issue a DISPLAY RESOURCE statement on the named resource. The grantable DISPLAY privilege allows a user to issue a DISPLAY PRIVILEGES statement on the named resource.

DROP
Gives the DROP privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The DROP privilege on a resource allows the user to delete the definition of the resource.

ON
Specifies the resource to which the specified definition privileges apply.

GROUP group-identifier
Identifies a group.
You can wildcard group-identifier.

Note: For more information, see Using a Wildcard (see page 57).

USER user-identifier
Identifies a user.
You can wildcard user-identifier.

Note: For more information, see Using a Wildcard (see page 57).

USER PROFILE profile-name
Identifies a user profile.
You can wildcard profile-name.

Note: For more information, see Using a Wildcard (see page 57).

TO
Specifies the users to whom you are giving definition privileges.

PUBLIC
Specifies all users.

authorization-identifier
Identifies a user or group.
• WITH GRANT OPTION
  Gives the privilege of granting the specified definition privileges to the users or groups identified in the TO parameter.
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

GRANT Definition Privileges Usage

Granting Definition Privilege with a Wildcard

By wildcarding the resource name when you grant a definition privilege, you allow a user to define multiple resources that are named with the same beginning characters.

For example, if you grant DEFINE privilege on user profile HR_* to group HR_ADMIN (see the following example), you allow the human resources administrative group to create, alter, and drop profiles that begin with the characters 'HR_'.

The DEFINE Keyword

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges on a resource to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all definition privileges that have been previously granted on the resource from the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges on the resource from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user. This is an efficient technique for granting all but one definition privilege.

The following GRANT statement gives the HR_ADMIN group the privilege to define user profiles that begin 'HR_' :

grant define
  on user profile hr_*
  to hr_admin;

GRANT Definition Privileges More Information

For more information about revoking definition privileges, see REVOKE Administration Privilege (see page 207).
REVOKE Administration Privilege

Contents

- REVOKE Administration Privilege Authorization (see page 207)
- REVOKE Administration Privilege Syntax (see page 207)
- REVOKE Administration Privilege Parameters (see page 207)
- REVOKE Administration Privilege Usage (see page 208)
- REVOKE Administration Privilege More Information (see page 208)

Revoke SYSADMIN privilege from a user or group.

REVOKE Administration Privilege Authorization

To revoke SYSADMIN privilege, you must hold SYSADMIN privilege.

REVOKE Administration Privilege Syntax

```
REVOKE SYSADMIN
FROM PUBLIC
```

REVOKE Administration Privilege Parameters

- **SYSADMIN**
  Specifies that you are revoking SYSADMIN privilege from the users or groups named in the FROM parameter.

- **FROM**
  Specifies the users or groups from whom you are revoking SYSADMIN privilege.

- **PUBLIC**
  Specifies all users.

- **authorization-identifier**
  Identifies a user or group.

Note: Expanded syntax for **authorization-identifier** is presented in the section Notes on Security Statement Syntax (see page 178).
REVOKE Administration Privilege Usage

Revoking SYSADMIN from All Users

If the SYSA resource type is secured in the SRTT and SYSADMIN is revoked from all users and groups that hold SYSADMIN privilege, no user has the privilege to grant SYSADMIN and, therefore, no user has overall authority to administer the security system.

In this situation, you must turn off security for the SYSA resource type in the SRTT. With SYSA security off, any user can grant SYSADMIN privilege. Once you have granted the privilege to the appropriate users, you reactivate security for SYSA.

Revoking SYSADMIN from PUBLIC

This statement revokes SYSADMIN privilege from the group PUBLIC:

```
revoke sysadmin
from public;
```

REVOKE Administration Privilege More Information

- For more information about grant SYSADMIN privilege, see GRANT Administration Privilege (see page 201).
- For more information about the role of the SYSADMIN user, see Securing Global Resources (see page 61).

REVOKE Definition Privileges

Contents

- REVOKE Definition Privileges Authorization (see page 208)
- REVOKE Definition Privileges Syntax (see page 209)
- REVOKE Definition Privileges Parameters (see page 209)
- REVOKE Definition Privileges Usage (see page 210)
- REVOKE Definition Privileges More Information (see page 211)

Revoke from one or more users or groups the privilege of performing definition functions on a user, group, or user profile in the user catalog.

REVOKE Definition Privileges Authorization

To revoke a definition privilege on a global resource, you must hold one of the following privileges:

- SYSADMIN privilege
- The corresponding grantable privilege

**REVOKE Definition Privileges Syntax**

![Diagram of REVOKE Definition Privileges Syntax]

**REVOKE Definition Privileges Parameters**

- **DEFINE**
  Revokes the ALTER, CREATE, and DROP privileges on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **ALTER**
  Revokes the ALTER privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **CREATE**
  Revokes the CREATE privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **DISPLAY**
  Revokes the DISPLAY privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **DROP**
  Revokes the DROP privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **ON**
  Specifies the resource to which the definition privileges apply.

- **GROUP group-identifier**
  Identifies a group.

- **USER user-identifier**
  Identifies a user.

- **USER PROFILE profile-name**
  Identifies a user profile.

- **FROM**
  Specifies the users or groups from whom you are revoking definition privileges.
PUBLIC
Specifies all users.

authorization-identifier
Identifies a user or group.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

REVOKE Definition Privileges Usage

The DEFINE keyword

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges on a resource to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all definition privileges that have been previously granted on the resource from the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges on the resource from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user. This is an efficient technique for granting all but one definition privilege.

Using a Wildcard When Revoking Definition Privilege

If you use a wildcard in the resource name in a REVOKE statement, the wildcarded name must match the wildcarded name used in a previous GRANT statement. Similarly, if definition privilege was granted on a wildcarded resource name, it can be revoked only by specifying the same name in the REVOKE statement.

For example, assume that this grant has been made:

grant define
  on user profile hr*
  to user1, user2, user3;

- To revoke the privilege, you must specify HR* in the REVOKE statement.
- If you revoke DEFINE on user profile HRA from USER1, it has no effect on the privilege granted in the preceding example.

The following statement revokes the privilege to define certain user profiles from the hr_corp group:

revoke define
  on user profile hr_*
  from hr_corp;
Syntax for Securing System Resources

This section explains the syntax, parameters, command codes, examples, and usage of the following Resources:

- ALTER RESOURCE (see page 213)
  - Authorization (see page 213)
  - Syntax (see page 213)
  - Parameters (see page 213)
  - Usage (see page 215)
  - Examples (see page 216)
  - More information (see page 216)
- CREATE RESOURCE (see page 217)
  - Authorization (see page 217)
  - Syntax (see page 217)
  - Parameters (see page 217)
  - Usage (see page 220)
  - Examples (see page 221)
  - More information (see page 221)
- DROP RESOURCE (see page 221)
  - Authorization (see page 221)
  - Syntax (see page 222)
  - Parameters (see page 222)
  - Usage (see page 222)
  - Examples (see page 222)
  - More information (see page 223)
- GRANT Administration Privilege (see page 223)
  - Authorization (see page 223)
  - Syntax (see page 223)
  - Parameters (see page 223)
  - Usage (see page 224)
  - Examples (see page 224)
  - More information (see page 224)
- GRANT Execution Privilege (see page 224)
  - Authorization (see page 224)
  - Syntax (see page 225)
  - Parameters (see page 225)
Usage (see page 226)
Examples (see page 226)

**GRANT Signon Privilege** (see page 227)
- Authorization (see page 227)
- Syntax (see page 227)
- Parameters (see page 227)
- Usage (see page 228)
- Examples (see page 228)
- More information (see page 228)

**GRANT System Definition Privileges** (see page 229)
- Authorization (see page 229)
- Syntax (see page 229)
- Parameters (see page 229)
- Usage (see page 231)
- Examples (see page 231)
- More information (see page 231)

**REVOKE Administration Privilege** (see page 232)
- Authorization (see page 232)
- Syntax (see page 232)
- Parameters (see page 232)
- Example (see page 232)
- More information (see page 233)

**REVOKE Execution Privilege** (see page 233)
- Authorization (see page 233)
- Syntax (see page 233)
- Parameters (see page 233)
- Examples (see page 234)
- More information (see page 234)

**REVOKE Signon Privilege** (see page 234)
- Authorization (see page 235)
- Syntax (see page 235)
- Parameters (see page 235)
- Example (see page 235)
- More information (see page 235)

**REVOKE System Definition Privileges** (see page 236)
- Authorization (see page 236)
- Syntax (see page 236)
- Parameters (see page 236)
- Usage (see page 237)
- Examples (see page 237)
- More information (see page 238)
ALTER RESOURCE

Modifies the definition of an activity or a category in the system dictionary.

Authorization

To modify a system, activity, or category definition, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

```
ALTER RESOURCE
ACTIVITY application-name.activity-name NUMBER activity-number
CATEGORY category-name ADD category-component
DROP category-component
```

Expansion of Category-Component

```
ACCESS MODULE dictionary-name.schema-name.access-module-name
LOAD MODULE dictionary-name.Vnnnn.load-module-name
PROGRAM file-name.program-name
QUEUE queue-name
RUNUNIT database-name.subschema-name.program-name
TASK task-code
```

Parameters

- **ACTIVITY**
  Specifies that you are modifying the security definition of an activity.

- **application-name**
  Identifies the application that includes the activity to be altered.

- **activity-name**
  Identifies the application activity to be altered.
  The activity must have been previously defined with a CREATE RESOURCE statement.

- **NUMBER activity-number**
  Specifies the security classification number to be assigned to the activity.
  Activity-number must be in the range 1 to 256 and must be unique for the application.
• CATEGORY category-name
  Specifies that you are modifying the definition of a category.
  Category-name must have been previously defined with a CREATE RESOURCE statement.

• ADD
  Specifies that you are adding a component to the category.

• DROP
  Specifies that you are dropping a component from the category.

• category-component
  Identifies the component to be added to or dropped from the category.
  Expanded syntax for category-component directly follows syntax for ALTER RESOURCE.
  A component that has been assigned to a category cannot be assigned to a second category.
  You can wildcard the name of the category component.
  Note: For more information on wildcarding, see Using a Wildcard (see page 57).

• ACCESS MODULE
  Specifies that the category component is an access module.
  If you drop an access module from a category, users with execution privilege on the category may
  no longer be able to execute the access module.

• dictionary-name
  Identifies the dictionary where the access module is stored (DDLCATLOD area).

• schema-name
  Identifies the schema with which the access module is associated.

• access-module-name
  Identifies the access module.

• LOAD MODULE
  Specifies that the category component is a load module.
  If you drop a load module from a category, users with execution privilege on the category may no
  longer be able to execute the load module.

• dictionary-name
  Identifies the dictionary where the load module is stored (DDLDCLOD area).

• Vnnnn
  Specifies the version of the load module.
  Nnnn identifies the version number of the load module. Leading zeros must be included.

• load-module-name
  Identifies the load module.
  Load-module-name must match the name of an entity defined in the dictionary by means of a CA
  IDMS or CA ADS compiler.

• PROGRAM
  Specifies that the category component is a program.
  If you drop a program from a category, users with execution privilege on the category may no
  longer be able to execute the program.
- **file-name**
  Supplies the external file name of the load library where the program is stored.
  File-name is either 'CDMSLIB' or 'Vnnnn' where nnnn identifies the version number (2 - 9999) of the program. Nnnn must include leading zeroes.

- **program-name**
  Identifies the program.

- **QUEUE**
  Specifies that the category component is a queue.
  If you drop a queue from a category, users with execution privilege on the category may no longer be able to create or access the queue.

- **queue-name**
  Identifies the queue.

- **RUNUNIT**
  Specifies that the category component is a run unit.
  If you drop a run unit from a category, users with execution privilege on the category may no longer be able to execute the run unit.

- **database-name**
  Specifies the database to be accessed by the run unit.

- **subschema-name**
  Specifies the subschema to be used by the run unit.

- **program-name**
  Specifies the name of the program binding the run unit.

- **TASK**
  Specifies that the category component is a task.
  If you drop a task from a category, users with execution privilege on the category may no longer be able to execute the task.

- **task-code**
  Identifies the task.

---

**Usage**

**Execution Privilege on Individual Access Modules**

You can give access module execution privilege by issuing a GRANT EXECUTE ON ACCESS MODULE statement. However, if you subsequently add the same access module to a category, the separate grant of access module execution is ignored at runtime.

If you then drop the access module from the category, and you have not revoked the separate grant of access module execution privilege, the privilege will again be respected at runtime.

**Matching Wildcarded Category Component Names**
If you use a wildcard when you specify a category component name, the category to which the resource is assigned is the one that most closely matches the resource name.

A match is determined by these rules:

- The resource name must match the category component name character for character up to the wildcard in order to be assigned to the category.
- If the resource name matches more than one category component name up to the wildcard, the resource is assigned to the category in which the most number of characters match.

For example, assume you add these two category component definitions:

```sql
alter resource category hr_prod
  add load module hrdict.v0001*;
alter resource category hr_test
  add load module hrdict.*;
```

A load module named HRDICT.V0001.HRMAP1 will be assigned to category HR_PROD, whereas load module HRDICT.V0002.HRMAP1 will be assigned to category HR_TEST.

**Altering Categories After a Grant**

If a component is added to a category with ALTER RESOURCE after a grant of privilege on the category, the privilege is implicitly granted on the added component. Similarly, if a component is dropped from a category with ALTER RESOURCE after a grant of privilege on the category, the privilege on the dropped component is implicitly revoked.

**Examples**

**Altering an Activity:** The following statement assigns a new number to an existing activity created for a General Ledger application:

```sql
alter resource activity cgl.post
  number 5;
```

**Altering a Category:** The following statement replaces a resource in an existing category created for a General Ledger application:

```sql
alter resource category glappl
  drop queue fl*
  add queue gl*;
```

**More information**

For more information on creating and dropping resources, see the topics CREATE RESOURCE (see page 217) and DROP RESOURCE (see page 221).
CREATE RESOURCE

Creates the definition of a system, activity, or category in the system dictionary.

Authorization

To create a system, activity, or category definition, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

```
CREATE RESOURCE
   ACTIVITY application-name.activity-name NUMBER activity-number
   CATEGORY category-name ADD category-component
   SYSTEM system-identifier
```

Expansion of Category-Component

```
ACCESS MODULE dictionary-name.schema-name.access-module-name
LOAD MODULE dictionary-name.Vnnnn.load-module-name
PROGRAM file-name.program-name
QUEUE queue-name
RUNUNIT database-name.subschema-name.program-name
TASK task-code
```

Parameters

- **ACTIVITY**
  Specifies that you are defining an activity as a secured resource.
  An activity is a discrete application function. After you have defined an activity, you grant execution privilege on the activity to users.
  **Note:** For more information on activities, see Securing System Resources (see page 70).

- **application-name**
  Identifies the application that includes the activity to be secured.
  Application-name must match the name of the application (passed in #SECHECK) whose function is being secured. It can be at most eight characters in length.

- **activity-name**
  Names the application function to be secured.
  Activity-name must be unique within the application. It can be at most 18 characters in length.
• **NUMBER activity-number**
  Specifies the activity number assigned to the application function. Activity-number must be unique within the application. It must be in the range 1 to 256, and it must match the security class assigned within the application.

  **Note:** For more information on assigning activity numbers within applications, see Securing System Resources (see page 70).

• **CATEGORY category-name**
  Specifies that you are defining category category-name as a secured resource. A category can contain tasks, load modules, programs, access modules, run units, and queues. After you have defined a category, you grant execution privilege on the category to users. Category-name can be at most 32 characters in length. You can define a maximum of 32,767 categories for a DC system.

  **Note:** For more information on categories, see Securing System Resources (see page 70).

• **ADD category-component**
  Identifies a component to be added to the category. Expanded syntax for category-component directly follows syntax for CREATE RESOURCE. A component that has been assigned to a category cannot be assigned to a second category. You can wildcard the name of the category component.

  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

• **SYSTEM system-identifier**
  Specifies that system system-identifier is a secured resource. When the system is secured, users must hold definition privileges to define the system and signon privilege to sign on to the system. When the system is defined, SYSTEM ID in the SYSTEM statement of system generation must match the value of system-identifier.

• **ACCESS MODULE**
  Specifies that the category component is an access module. After you add an access module to a category, a user with execution privilege on the category can load the access module but cannot execute it.

  • **dictionary-name**
    Identifies the dictionary where the access module is stored (DDLCATLOD area).

  • **schema-name**
    Identifies the schema with which the access module is associated.

  • **access-module-name**
    Identifies an access module.

• **LOAD MODULE**
  Specifies that the category component is a load module. After you add a load module to a category, a user with execution privilege on the category can execute the load module.

  • **dictionary-name**
    Identifies the dictionary where the load module is stored (DDLDCLOD area).
• Vnnnn
  Specifies the version of the load module.
  Nnnn identifies the version number of the load module. Leading zeros must be included.

• load-module-name
  Identifies a load module.

• PROGRAM
  Specifies that the category component is a program.
  After you add a program to a category, a user with execution privilege on the category can execute the program.

• file-name
  Supplies the name of the external file of the load library where the program is stored.
  File-name is either 'CDMSLIB' or 'Vnnnn' where nnnn identifies the version number (2 - 9999) of the program. Nnnn must include leading zeroes.

• program-name
  Identifies a program.

• QUEUE
  Specifies that the category component is a queue.
  After you add a queue to a category, a user with execution privilege on the category can create or access the queue.

• queue-name
  Identifies a queue.

• RUNUNIT
  Specifies that the category component is a run unit.
  After you add a run unit to a category, a user with execution privilege on the category can use program-name to access data described in subschema-name of database-name.

• database-name
  Identifies the database to be accessed by the run unit.

• subschema-name
  Identifies the subschema to be used by the run unit.

• program-name
  Identifies the program binding the run unit.

• TASK
  Specifies that the category component is a task.
  After you add a task to a category, a user with execution privilege on the category can execute the task.

• task-code
  Identifies the task.
Adding Categories or Applications After a Grant

A category created after you grant privileges on the category with a wildcard is not included in the scope of the grant even if the resource name matches the wildcarded name. The wildcard is processed at the time the grant is made, not at runtime.

For example, if you create categories X1, X2, and X3, and then grant privileges on category X*, the three categories are within the scope of the grant. If you then create category X4, this category is not within the scope of the grant.

You can include X4 within the scope of the grant by reissuing the original grant on category X*.

The same considerations apply if you wildcard activity-name when granting execution privilege on an activity. The wildcard is processed at the time the grant is made, not at runtime.

Altering Categories After a Grant

If a component is added to a category (ALTER RESOURCE) after a grant of privilege on the category, the privilege is implicitly granted on the added component. Similarly, if a component is dropped from a category (ALTER RESOURCE) after a grant of privilege on the category, the privilege on the dropped component is implicitly revoked.

Matching Wildcarded Category Component Names

If you use a wildcard when you specify a category component name, the category to which the resource is assigned is the one that most closely matches the resource name.

A match is determined by these rules:

- The resource name must match the category component name character for character up to the wildcard in order to be assigned to the category.

- If the resource name matches more than one category component name up to the wildcard, the resource is assigned to the category in which the most number of characters match.

For example, assume you add these two category component definitions:

```c
create resource category hr_prod
    add load module hrdict.v0001*;
create resource category hr_test
    add load module hrdict.*;
```

A load module named HRDICT.V0001.HRMAP1 will be assigned to category HR_PROD, whereas load module HRDICT.V0002.HRMAP1 will be assigned to category HR_TEST.
Examples

The following statement creates a category called PROD_ACCESS that includes resources required for a user to perform production processing:

```
create resource
category prod_access
    add access module proddict.prod*
    add load module proddict.v0001.*
    add program cdmslib.*
    add rununit p*;
```

Creating an Activity

The following statement assigns an activity name and number to a General Ledger application:

```
create resource
    activity cgl.post
    number 4;
```

Creating a Category

The following statement creates a category of resources used by a General Ledger application:

```
create resource
category glappl
    add load module appldict.v0001.gl*
    add task gl*
    add queue gl*
    add program cdmslib.gl*;
```

More information

For more information on altering and dropping resources, see “ALTER RESOURCE” and “DROP RESOURCE.”

DROP RESOURCE

Deletes the definition of a secured resource.

Authorization

To delete the definition of a system, activity, or category as a secured resource, you must hold one of these privileges:

- DCADMIN
- SYSADMIN
Syntax

```
DROP RESOURCE
  ACTIVITY application-name.activity-name
  CATEGORY category-name
  SYSTEM system-identifier
```

Parameters

- **ACTIVITY**
  Specifies that the resource to be dropped is an activity.

- **application-name.activity-name**
  Identifies the activity.

- **CATEGORY**
  Specifies that the resource to be dropped is a category.

- **category-name**
  Identifies the category.

- **SYSTEM**
  Specifies that the resource to be dropped is a system.

- **system-identifier**
  Identifies the system.

Usage

**Automatic Revoking of Privileges**

When you drop a resource, you implicitly revoke all privileges granted on the resource.

Examples

**Dropping an Activity**

The following statement drops an existing activity created for a General Ledger application:

```
drop resource activity cgl.post;
```

**Dropping a Category**

The following statement drops an existing category created for a General Ledger application:

```
drop resource category glappl;
```
More information

For more information on creating and altering resources, see CREATE RESOURCE (see page 217) and ALTER RESOURCE (see page 213).

GRANT Administration Privilege

Gives one or more users or groups DCADMIN privilege.

Authorization

To grant DCADMIN privilege, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

```
GRANT DCADMIN

TO PUBLIC
```

Parameters

- **DCADMIN**
  Specifies that you are giving DCADMIN privilege to the users or groups identified in the TO parameter.
  DCADMIN controls access to DC system resources. A user with DCADMIN privilege can define system resources and can grant and revoke privileges on system resources.
  Note: For more information on the DCADMIN privilege, see Securing System Resources (see page 70).

- **TO**
  Specifies the users or groups to whom you are giving DCADMIN privilege.

- **PUBLIC**
  Specifies all users.

Important. If you grant DCADMIN to group PUBLIC, any user can administer security for the system.
- **authorization-identifier**
  Identifies a user or group.

  **Note:** Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax.

**Usage**

**Decentralizing Administration**

You can decentralize security administration for the system by granting DCADMIN and system definition privileges to other users.

A holder of SYSADMIN or DCADMIN can also specify WITH GRANT OPTION when granting system definition privileges to allow the recipient to grant the same privileges to others.

**Examples**

**Granting DCADMIN to Administrators:**

The following statement grants DCADMIN privilege to the security administrator ID and the DCA group ID:

```plaintext
grant dadmin
to secadmin, dca_group;
```

**More information**

For more information on revoking DCADMIN privilege, see [REVOKE Administration Privilege](#) (see page 207).

**GRANT Execution Privilege**

Gives one or more users or groups access to activities or categories.

**Authorization**

To grant an execution privilege, you must hold one of these privileges:

- DCADMIN
- SYSADMIN
Syntax

```
GRANT EXECUTE
  ON ACTIVITY application-name.activity-name
  CATEGORY category-name
  TO PUBLIC authorization-identifier
```

Parameters

- **EXECUTE**
  Specifies that you are giving execution privilege to the users or groups identified in the TO parameter.

- **ON**
  Specifies the resources to which execution privilege applies.

- **ACTIVITY**
  Specifies that you are giving execution privilege on one or more activities.

- **application-name.activity-name**
  Identifies an activity.
  Application-name.activity-name must have been previously defined as an activity with a CREATE RESOURCE statement.
  You can wildcard activity-name. You cannot wildcard application-name. The wildcard character can appear in activity-name at any point after the period following application-name.
  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **CATEGORY**
  Specifies that you are giving execution privilege on one or more categories.

- **category-name**
  Identifies a category.
  Category-name must have been previously defined as a category with a CREATE RESOURCE statement.
  You can wildcard category-name.
  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **TO**
  Specifies the users or groups to whom you are giving execution privilege.

- **PUBLIC**
  Specifies all users.
authorization-identifier
Identifies a user or group.

Note: Expanded syntax for authorization-identifier is presented in the section “Notes on Security Statement Syntax.”

Usage

Wildcarding Activity-Name

If you wildcard activity-name in a GRANT EXECUTE statement, you grant execution privilege on all activities whose names match the wildcarded name.

For example, this statement gives execution privilege on all DCMT activities whose names begin with 'VARY' to DCA_GROUP:

grant execute
  on activity dcmt.vary*
  to dca_group;

Adding Categories or Activities After a Grant

A category created after you grant privileges on the category with a wildcard is not included in the scope of the grant even if the resource name matches the wildcarded name. The wildcard is processed at the time the grant is made, not at runtime.

For example, if you create categories X1, X2, and X3, and then grant privileges on category X*, the three categories are within the scope of the grant. If you then create category X4, this category is not within the scope of the grant.

You can include X4 within the scope of the grant by revoking the original grant on category X* and then reissuing it.

The same considerations apply if you wildcard activity-name when granting execution privilege on an activity. The wildcard is processed at the time the grant is made, not at runtime.

Examples

Granting Execution Privilege on an Activity

The following statement grants execution privilege on a General Ledger activity to a development group:

grant execute
  on activity cgl.post
  to appldev1;

Granting Execution Privilege on a Category

The following statement grants execution privilege on the General Ledger category to all users:
grant execute
   on category glappl
   to public;

Granting Execution Privilege to PUBLIC

If execution privilege on a category has been granted to group PUBLIC, a user who has not signed on has the ability to invoke tasks and access other secured resources in the category.

GRANT Signon Privilege

Gives one or more users access to a CA IDMS system.

Authorization

To issue the GRANT SIGNON statement, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

```
GRANT SIGNON ON SYSTEM system-identifier
  PROFILE profile-name
  NULL
  TO user-identifier
```

Parameters

- **SIGNON**
  Specifies that you are giving signon privilege to the users identified in the TO parameter.

- **ON SYSTEM system-identifier**
  Identifies the system to which signon privilege applies.
  System-identifier must have been defined as a resource with the CREATE RESOURCE SYSTEM statement.
  **Note:** You cannot wildcard system-identifier.

- **PROFILE profile-name**
  Identifies the system profile to be used in signon processing.

- **NULL**
  Specifies that no system profile should be used in signon processing.
• **TO user-identifier**
  Identifies a user to whom you are giving signon privilege.

**Usage**

**Changing the User’s System Profile**

To change the system profile associated with a user identifier:

- Revoke signon privilege from the user identifier.
- Grant signon privilege to the user identifier specifying the new profile.

**Examples**

**Granting Signon with a Profile Specification**

The following statement grants signon privilege and specifies the system profile to be invoked in signon processing:

```plaintext
grant signon
  on system syst0099
  profile pub99
  to sam;
```

**Changing the User’s System Profile**

The following statements change user SAM’s system profile to CORP99:

```plaintext
revoke signon on system syst0099
  from sam;

grant signon on system syst0099
  profile corp99
  to sam;
```

**More information**

- For more information on signon processing, see [Signon Processing](https://docops.ca.com/display/IDMS19/Signon+Processing).
- For more information on revoking signon privilege, see [REVOKE Signon Privilege](see page 234).
- For more information on system profiles, see the [CA IDMS System Tasks and Operator Reference section](#).
GRANT System Definition Privileges

Gives to one or more users or groups the privilege to define a system or a system profile in the system dictionary.

Authorization

To grant system definition privileges, you must hold one of these privileges:

- DCADMIN privilege
- The corresponding grantable privilege on the system or system profile
- SYSADMIN

Syntax

```
GRANT DEFINE ─┬─ ALTER CREATE DISPLAY DROP ────
│ ┌───────┐ └─▼─┬──┬─▼─┬─▼─┬─▼─┬─▼─┘
│ └─▼─┬──┬─▼─┬─▼─┬─▼─┬─▼─┘
├─ CREATE ─┤ CREATE ─┤ CREATE ─┤ CREATE ─┤ CREATE ─┤ CREATE ─┤ CREATE ─┘
└─ DISPLAY ┘ └─ DROP ┘ └─ DROP ┘ └─ DROP ┘ └─ DROP ┘ └─ DROP ┘

ON ─┬─ SYSTEM system-identifier ──┬─ SYSTEM PROFILE profile-name ─┘

TO ─▼─ PUBLIC authorization-identifier ┘

WITH GRANT OPTION
```

Parameters

- **DEFINE**
  
  Gives the ALTER, CREATE, DISPLAY, and DROP privileges on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  
  The DEFINE privilege on a system allows the user to use the CA IDMS system generation compiler to add, modify, or delete the system. The DEFINE privilege also allows the user to display or punch the definition of the system.
  
  The DEFINE privilege on a system profile allows the user to create, alter, or drop a system profile, or to display security definitions for the system profile.
ALTER
Gives the ALTER privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The ALTER privilege on a system allows a user to use the system generation compiler to modify the definition of the system.
The ALTER privilege on a system profile allows a user to alter the system profile.

CREATE
Gives the CREATE privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The CREATE privilege on a system allows a user to use the system generation compiler to add the system.
The CREATE privilege on a system profile allows a user to create the system profile.

DISPLAY
Gives the DISPLAY privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The DISPLAY privilege allows the user to issue a DISPLAY RESOURCE statement on the named system or system profile. The grantable DISPLAY privilege allows a user to issue a DISPLAY PRIVILEGES statement on the named system or system profile.
The DISPLAY privilege on a system also allows a user to use the system generation compiler to display or punch the system definition.

DROP
Gives the DROP privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The DROP privilege on a system allows a user to use the system generation compiler to delete the definition of the system.
The DROP privilege on a system profile allows a user to drop the system profile.

ON
Specifies the resource to which the definition privileges apply.

SYSTEM system-identifier
Identifies a system.
You can wildcard system-identifier.
Note: For more information on wildcarding, see Using a Wildcard (see page 57).
If you do not wildcard system-identifier, the value you specify must match the identifier of a system secured with the CREATE RESOURCE SYSTEM statement.

SYSTEM PROFILE profile-name
Identifies a system profile.
Note: For more information on creating system profiles, see the CA IDMS System Tasks and Operator Reference section.
You can wildcard system-profile.
Note: For more information on wildcarding, see Using a Wildcard (see page 57).

TO
Specifies the users or groups to whom you are giving the definition privileges.

PUBLIC
Specifies all users.
• **authorization-identifier**  
  Identifies a user or group.  
  **Note:** Expanded syntax for authorization-identifier is presented in the section [Notes on Security Statement Syntax](#) (see page 178).

• **WITH GRANT OPTION**  
  Gives the privilege of granting the specified definition privileges to the users or groups identified in the TO parameter.  
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

### Usage

#### The DEFINE Keyword

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges on a resource to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all definition privileges that have been previously granted on the resource from the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges on the resource from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user as a way to grant all but one definition privilege.

### Examples

#### Granting Privilege to Define the System

The following statement grants the privilege to define a system to the system DCA:

```
grant define
  on system syst0099
  to dca0099;
```

### More information

• For more information on system profiles, see the [CA IDMS System Tasks and Operator Reference](#) section.

• For more information on revoking system definition privileges, see [REVOKE System Definition Privileges](#) (see page 236).
REVOKE Administration Privilege

Revises DCADMIN privilege from one or more users or groups.

Authorization

To revoke DCADMIN privilege, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

```
REVOKE DCADMIN
  FROM authorization-identifier
  PUBLIC
```

Parameters

- **DCADMIN**
  Specifies that you are revoking DCADMIN privilege from the users or groups named in the FROM parameter.

- **FROM**
  Specifies the users or groups from whom you are revoking DCADMIN privilege.

- **PUBLIC**
  Specifies all users.

- **authorization-identifier**
  Identifies a user or group.
  The privilege must have been previously given to authorization-identifier by means of the GRANT statement.
  Note: Expanded syntax for authorization-identifier is presented in section “Notes on Security Statement Syntax.”

Example

**Revoking DCADMIN from Administrators**

The following statement revokes DCADMIN privilege from the DCA group ID:

```
revoke dcadmin
  from dca_group;
```
More information

For more information on granting DCADMIN privilege, see GRANT Administration Privilege (see page 223).

REVOKE Execution Privilege

Revoke execution privilege.

Authorization

To revoke execution privilege, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

```
REVOKE EXECUTE
ON ACTIVITY application-name.activity-name
CATEGORY category-name
FROM PUBLIC authorization-identifier
```

Parameters

EXECUTE

Specifies that you are revoking execution privilege from the users or groups identified in the FROM parameter.

- **ON**
  Specifies the resource to which execution privilege applies.

- **ACTIVITY**
  Specifies that you are revoking execution privilege on one or more activities.

  `application-name.activity-name`

  Identifies an activity.

  You can wildcard activity-name. You cannot wildcard application-name. The wildcard character can appear in activity-name at any point following the period after application-name.

Note: For more information on wildcarding, see Using a Wildcard (see page 57).
• CATEGORY
  Specifies that you are revoking execution privilege on one or more categories.

• category-name
  Identifies a category.

• FROM
  Specifies the users or groups from whom you are revoking execution privilege.

• PUBLIC
  Specifies all users.

• authorization-identifier
  Identifies a user or group.
  The privilege must have been previously given to authorization-identifier by means of the GRANT statement.
  Note: Expanded syntax for authorization-identifier is presented in the section Notes on Security Statement Syntax (see page 178).

Examples

Revoking Execution Privilege on an Activity

The following statement revokes execution privilege on a General Ledger activity from a development group:

```
revoke execute
  on activity cgl.post
  from appldev1;
```

Revoking Execution Privilege on a Category

The following statement revokes execution privilege on the General Ledger category from PUBLIC:

```
revoke execute
  on category glappl
  from public;
```

More information

For more information on granting execution privilege, see GRANT Execution Privilege (see page 224).

REVOKE Signon Privilege

Revoes the privilege to access a system from one or more users.
Authorization

To revoke signon privilege, you must hold one of these privileges:

- DCADMIN
- SYSADMIN

Syntax

REVOKE SIGNON ON SYSTEM system-identifier

FROM user-identifier

Parameters

- **SIGNON**
  Specifies that you are revoking signon privilege to the system identified in the ON parameter from the users identified in the FROM parameter.

- **ON SYSTEM system-identifier**
  Identifies the system to which the signon privilege applies.

- **FROM user-identifier**
  Identifies a user from whom you are revoking signon privilege.
  The privilege must have been previously given to user-identifier by means of the GRANT statement.

Example

Revoking Signon

The following statement revokes signon privilege on a specified system from a user:

revoke signon
  on system syst0099
  from sam;

More information

For more information on granting signon privilege, see GRANT Signon Privilege (see page 227).
REVOKE System Definition Privileges

Revokes from one or more users or groups the privilege to define a system or a system profile in the system dictionary.

Authorization

To revoke system privileges, you must hold one of these privileges:

- DCADMIN privilege
- The corresponding grantable privilege on the system or system profile
- SYSADMIN

Syntax

```plaintext
REVOKE DEFINE ALTER CREATE DISPLAY DROP ON SYSTEM system-identifier SYSTEM PROFILE profile-name FROM PUBLIC authorization-identifier
```

Parameters

- **DEFINE**
  Revokes the ALTER, CREATE, DISPLAY, and DROP privileges on the resource identified in the ON parameter from the users or groups identified in the TO parameter.

- **ALTER**
  Revokes the ALTER privilege on the resource identified in the ON parameter from the users or groups identified in the TO parameter.

- **CREATE**
  Revokes the CREATE privilege on the resource identified in the ON parameter from the users or groups identified in the TO parameter.

- **DISPLAY**
  Revokes the DISPLAY privilege on the resource identified in the ON parameter from the users or groups identified in the TO parameter.

- **DROP**
  Revokes the DROP privilege on the resource identified in the ON parameter from the users or groups identified in the TO parameter.
• **ON**
  Specifies the resource to which the definition privileges apply.

• **SYSTEM system-identifier**
  Identifies a system.

• **SYSTEM PROFILE profile-name**
  Identifies a system profile.

• **FROM**
  Specifies the users or groups from whom you are revoking the specified definition privileges.

• **PUBLIC**
  Specifies all users.

• **authorization-identifier**
  Identifies a user or group.
  The privileges must have been previously given to authorization-identifier by means of the GRANT statement.
  **Note:** Expanded syntax for authorization-identifier is presented in the section "Notes on Security Statement Syntax."

**Usage**

**The DEFINE Keyword**

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges on a resource to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all definition privileges that have been previously granted on the resource from the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges on the resource from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user as a way to grant all but one definition privilege.

**Examples**

**Revoking Privilege to Define the System**

The following statement revokes the privilege to define a system from the system DCA:

```
revoke define
  on system syst0099
  from dca0099;
```
More information

- For more information on system profiles, see the CA IDMS System Tasks and Operator Reference section.
- For more information on granting system definition privileges, see GRANT System Definition Privileges (see page 229).

Syntax for Securing Database Resources

This section explains the syntax, parameters, command codes, examples, and usage of the following Resources:

- GRANT Access Module Execution Privilege (see page 238)
- GRANT Administration Privilege 1 (see page 240)
- GRANT All Table Privileges (see page 242)
- GRANT Area Access Privileges (see page 245)
- GRANT Non-SQL Definition Privilege (see page 247)
- GRANT Physical Database Definition Privileges (see page 249)
- GRANT SQL Definition Privileges (see page 253)
- GRANT Table Access Privileges (see page 257)
- REVOKE Access Module Execution Privilege (see page 260)
- REVOKE Administration Privilege 1 (see page 262)
- REVOKE All Table Privileges (see page 263)
- REVOKE Area Access Privileges (see page 265)
- REVOKE Non-SQL Definition Privilege (see page 267)
- REVOKE Physical Database Definition Privileges (see page 268)
- REVOKE SQL Definition Privileges (see page 271)
- REVOKE Table Access Privileges (see page 274)
- TRANSFER OWNERSHIP (see page 277)

GRANT Access Module Execution Privilege

Contents

- GRANT Access Module Execution Privilege Authorization (see page 239)
- GRANT Access Module Execution Privilege Syntax (see page 239)
- GRANT Access Module Execution Privilege Parameters (see page 239)
- GRANT Access Module Execution Privilege Usage (see page 240)
- GRANT Access Module Execution Privilege More Information (see page 240)

Gives one or more users or groups the privilege of executing a specified access module.
GRANT Access Module Execution Privilege Authorization

To grant access module execution privilege, one of the following must be true:

- You hold grantable execution privilege on the access module (you can grant execution privilege, but you cannot specify WITH GRANT OPTION).
- You own the schema with which the access module is associated.
- You hold DBADMIN privilege on the dictionary that contains the access module.
- You hold SYSADMIN privilege.

You must be connected to the application dictionary that contains the access module when you issue the statement.

GRANT Access Module Execution Privilege Syntax

```
GRANT EXECUTE
ON ACCESS MODULE access-module-name
TO PUBLIC authorization-identifier
WITH GRANT OPTION
```

GRANT Access Module Execution Privilege Parameters

- **ON ACCESS MODULE** `access-module-name`
  Identifies the access module to which the EXECUTE privilege applies.
  You can wildcard `access-module-name`. If you specify `schema-name`, the wildcard character is valid after the period following `schema-name`.

  **Note:** For more information, see Using a Wildcard (see page 57).

- **TO**
  Identifies the users or groups to whom you are giving EXECUTE privilege.

- **PUBLIC**
  Specifies all users.

- **authorization-identifier**
  Identifies a user or group.
• **WITH GRANT OPTION**
  Gives the privilege of granting EXECUTE privilege on the named access module to the users identified in the TO parameter.
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

**GRANT Access Module Execution Privilege Usage**

*CA IDMS Internal Security Enforcement*

When executing an access module in a database for which CA IDMS internal security is in effect, the owner of the access module must either hold the applicable privileges on the table-like objects named in the SQL statements in the module, or own the table-like objects. You own the access module if you own the schema associated with the access module.

For a user who is *not* the owner to execute an access module, these conditions must be satisfied:

- The user must hold execution privilege on the access module.
- The owner must either hold the applicable *grantable* privileges on the table-like objects named in the SQL statements in the module, or own the table-like objects.

These rules allow you to restrict a user’s means of accessing data to application programs. If you grant table access privileges, the user can also access data through the Command Facility.

*External Security*

When executing an access module in a database for which external security is in effect, the user, regardless of ownership, must hold the applicable privileges on *all* tables accessed by SQL statements in the module, whether accessed directly or indirectly through a view.

*Granting Execution Privilege*

The following GRANT statement gives execution privilege on all access modules associated with schema HR that begin with ‘EMP’ to the groups PER_GRP_1 and PER_GRP_2:

```sql
grant execute
  on access module hr.emp*
  to per_grp_1, per_grp_2;
```

**GRANT Access Module Execution Privilege More Information**

For more information *about revoking execution privilege*, see *REVOKE Access Module Execution Privilege* (see page 260).

**GRANT Administration Privilege 1**

Contents
GRANT Administration Privilege Authorization

To grant DBADMIN privilege, you must hold one of these privileges:

- DBADMIN on the database
- SYSADMIN

You must be connected to the system dictionary when you issue the statement.

GRANT Administration Privilege Syntax

```
GRANT DBADMIN ON DB database-name
TO PUBLIC authorization-identifier
```

GRANT Administration Privilege Parameters

- **DBADMIN**
  Specifies that you are giving DBADMIN privilege on the database identified in the ON parameter to the users or groups identified in the TO parameter.

- **ON DB database-name**
  Specifies the database to which DBADMIN privilege applies. *Database-name* refers to either the name of a segment or a database name entry in the database name table. DBADMIN controls access to database resources. A user with DBADMIN privilege can grant and revoke privileges for the specified database. DBADMIN privilege also allows users to maintain physical database definitions in the dictionary identified by *database-name*.

**Note:** For more information on the DBADMIN privilege, see Securing Database Resources (see page 88).

You can wildcard *database-name*.

**Note:** For more information on wildcarding, see Using a Wildcard (see page 57).
**TO**
Specifies the users or groups to whom you are giving DBADMIN privilege.

**PUBLIC**
Specifies all users.

Important: If you grant DBAMIN to group PUBLIC, any user can administer security on the database.

**authorization-identifier**
Identifies a user or group.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

Granting DBADMIN to the DBA

The following statement grants DBADMIN privilege on database GLDB to the DBA group ID:

```sql
grant dbadmin
  on db gldb
  to dba_gldb;
```

**GRANT Administration Privilege More Information**

For more information about revoking DBADMIN privilege, see REVOKE Administration Privilege (see page 207).

**GRANT All Table Privileges**

Contents

- GRANT All Table Privileges Authorization (see page 242)
- GRANT All Table Privileges Syntax (see page 243)
- GRANT All Table Privileges Parameters (see page 243)
- GRANT All Table Privileges Usage (see page 244)
- GRANT All Table Privileges More Information (see page 244)

Gives one or more users all definition and access privileges on a specified table-like object.

**GRANT All Table Privileges Authorization**

To grant all table privileges, one of the following must be true:
You hold all privileges on the table-like object as grantable privileges. (you can grant the
privileges, but you cannot specify WITH GRANT OPTION)

You own the table-like object.

You hold DBADMIN privilege the application dictionary where the table-like object is defined and
on the database that contains the table-like object data.

You hold SYSADMIN privilege.

GRANT All Table Privileges Syntax

<table>
<thead>
<tr>
<th>GRANT ALL PRIVILEGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON table table-name</td>
</tr>
<tr>
<td>TO PUBLIC authorization-identifier</td>
</tr>
<tr>
<td>WITH GRANT OPTION</td>
</tr>
</tbody>
</table>

GRANT All Table Privileges Parameters

- **ALL PRIVILEGES**
  Gives the DELETE, INSERT, SELECT, UPDATE, ALTER, CREATE, DROP, and REFERENCES privileges on
  the table-like object, as applicable, identified in the ON parameter to the users or groups
  identified in the TO parameter.

- **ON table table-name**
  Identifies the table-like object to which the table privileges apply.

  **Note:** Expanded syntax for `table-name` is presented in Notes on Security Statement Syntax.

  You can wildcard the `identifier-components` of `table-name`. If you specify `schema-name` in `table-
  name`, the wildcard character is valid after the period following `schema-name`.

  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

  If you do not specify `schema-name`, it defaults to the current schema in effect for your session.

- **TO**
  Specifies the users or groups to whom you are giving table privileges.

- **PUBLIC**
  Specifies all users.
**authorization-identifier**
Identifies a user or group.

**Note:** Expanded syntax for **authorization-identifier** is presented in *Using a Wildcard (see page 57).*

**WITH GRANT OPTION**
Gives the privilege of granting the all table privileges on **table-name** to the users or groups identified in the TO parameter. The owner of the resource, a holder of the applicable DBADMIN privilege, or a holder of SYSADMIN privilege can specify WITH GRANT OPTION. A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

---

**GRANT All Table Privileges Usage**

*Verification of Wildcarded Grants*

When you grant all table privileges, you grant a combination of table access and table definition privileges. If you wildcard **table-name**, the verification at runtime of the user’s access privilege is handled differently from verification of the user’s definition privilege:

- For definition privileges, only the closest matching wildcarded grant is used. If CREATE privilege has been granted on HR.EMP* and HR.EMPV*, then only the grant on HR.EMPV* is used to verify the privilege to create HR.EMPVU_SALARY.

- For access privileges, all matching wildcarded grants are used. If SELECT privilege has been granted on HR.EMP* and HR.EMPV*, then users or groups receiving either the HR.EMP* or the HR.EMPV* grant are authorized to select from EMPVU_SALARY.

*Granting All Privileges to All Users*

The following GRANT statement gives all users all privileges on all table-like objects in the TEST schema:

```
grant all privileges
  on test.*
  to public;
```

---

**GRANT All Table Privileges More Information**

- For more information on table access privileges, see *GRANT Table Access Privileges (see page 257).*

- For more information on table definition privileges, see *GRANT SQL Definition Privileges (see page 253).*
GRANT Area Access Privileges

Contents

- GRANT Area Access Privileges Authorization (see page 245)
- GRANT Area Access Privileges Syntax (see page 245)
- GRANT Area Access Privileges Parameters (see page 245)
- GRANT Area Access Privileges More Information (see page 247)

Gives one or more users or groups access to an area of the database.

GRANT Area Access Privileges Authorization

To grant an area access privilege, you must hold one of these privileges:

- The grantable area access privilege on the area (you can grant the privilege, but you cannot specify WITH GRANT OPTION)
- DBADMIN on DB segment-name
- SYSADMIN

You must be connected to the system dictionary when you issue the statement.

GRANT Area Access Privileges Syntax

```
GRANT DBAREAD, DBWRITE, USE
ON AREA segment-name.area-name
TO PUBLIC authorization-identifier
WITH GRANT OPTION
```

GRANT Area Access Privileges Parameters

- **DBAREAD**
  Specifies that you are giving DBAREAD privilege on the area identified in the ON parameter to the users or groups specified in the TO parameter.
  A user with DBAREAD privilege can execute database utilities that perform read-only functions in the specified area.
- **DBAWRITE**
  Specifies that you are giving DBAWRITE privilege on the area identified in the ON parameter to the users or groups specified in the TO parameter. A user with DBAWRITE privilege can execute database utilities that perform read-write functions in the specified area.

  **Note:** DBAWRITE privilege does not imply DBAREAD privilege. You must give both privileges to users or groups who need to execute all utilities.

- **USE**
  Specifies that you are giving USE privilege on the area identified in the ON parameter to the users or groups specified in the TO parameter. A user with USE privilege can create an SQL table or index in the specified area.

- **ON AREA** `segment-name.area-name`
  Identifies the area to which the specified area access privileges apply. You can wildcard `area-name`. You cannot wildcard `segment-name`. The wildcard character is valid after the period following `segment-name`.

  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **TO**
  Specifies the users or groups to whom you are giving area access privileges.

- **PUBLIC**
  Specifies all users.

- **authorization-identifier**
  Identifies a user or group.

  **Note:** Expanded syntax for `authorization-identifier` is presented in Notes on Security Statement Syntax (see page 178).

- **WITH GRANT OPTION**
  Gives the privilege of granting the specified area access privileges to the users or groups identified in the TO parameter. Only a user with DBADMIN privilege on `segment-name` or with SYSADMIN privilege can specify WITH GRANT OPTION. A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

**Granting All Area Access Privileges**

The following statement grants all area access privileges to the specified users:

```
on area gl."account-area"
to matt, alex;
```
GRANT Area Access Privileges More Information

For more information on revoking the privilege to access an area, see REVOKE Area Access Privileges (see page 265).

GRANT Non-SQL Definition Privilege

Contents

- GRANT Non-SQL Definition Privilege Authorization (see page 247)
- GRANT Non-SQL Definition Privilege Syntax (see page 247)
- GRANT Non-SQL Definition Privilege Parameters (see page 247)
- GRANT Non-SQL Definition Privilege More Information (see page 249)

Gives one or more users or groups the privilege of referencing a non-SQL-defined schema in an SQL schema definition.

GRANT Non-SQL Definition Privilege Authorization

To grant the USE privilege on a non-SQL-defined schema, you must hold one of these privileges:

- Grantable privilege on the non-SQL-defined schema (you can grant the privilege, but you cannot specify WITH GRANT OPTION).
- DBADMIN on the dictionary containing the non-SQL-defined schema definition.
- SYSADMIN.

You must be connected to the dictionary containing the non-SQL-defined schema when you issue the statement.

GRANT Non-SQL Definition Privilege Syntax

```
GRANT USE DISPLAY
   ON NONSQL SCHEMA Vnnnn nonsql-schema-name
   TO PUBLIC authorization-identifier
   WITH GRANT OPTION
```

GRANT Non-SQL Definition Privilege Parameters
• **USE**
  Gives the USE privilege on the non-SQL-defined schema identified in the ON parameter to the users or groups identified in the TO parameter.

• **DISPLAY**
  Gives the DISPLAY privilege on the non-SQL-defined schema identified in the ON parameter to the users or groups identified in the TO parameter.

• **ON NONSQL SCHEMA**
  Specifies the non-SQL-defined schema to which the USE privilege applies.

• **Vnnnn.nonsql-schema-name**
  Specifies the version number and name of the non-SQL-defined schema. The version number (nnnn) must include leading zeros.
  You can wildcard nonsql-schema-name. You cannot wildcard Vnnnn. The wildcard character is valid after the period following Vnnnn.

  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

• **TO**
  Specifies the users or groups to whom you are giving the USE privilege.

• **PUBLIC**
  Specifies all users.

• **authorization-identifier**
  Identifies a user or group.

  **Note:** Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

• **WITH GRANT OPTION**
  Gives the privilege of granting the USE privilege on the named resource to the users or groups identified in the TO parameter.
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

*Granting USE On a Non-SQL-Defined Schema*

The following statement grants the privilege of referencing a non-SQL-defined schema when creating an SQL schema:

```sql
grant use
  on nonsql schema v0001.ap
  to sal, sam;
```
GRANT Non-SQL Definition Privilege More Information

For more information on revoking the privilege to use a non-SQL-defined schema, see REVOKE Non-SQL Definition Privilege (see page 267).

GRANT Physical Database Definition Privileges

Contents

- GRANT Physical Database Definition Privileges Authorization (see page 249)
- GRANT Physical Database Definition Privileges Syntax (see page 249)
- GRANT Physical Database Definition Privileges Parameters (see page 250)
- GRANT Physical Database Definition Privileges Usage (see page 252)
- GRANT Physical Database Definition Privileges More Information (see page 253)

Gives one or more users or groups the privilege of issuing DMCL, DBTABLE, and SEGMENT physical DDL statements.

GRANT Physical Database Definition Privileges Authorization

To grant a definition privilege on a DMCL or DBTABLE, you must hold one of these privileges:

- The corresponding grantable privilege
- DBADMIN on DB SYSTEM
- SYSADMIN

To grant a physical definition privilege on a database, you must hold one of the following privileges:

- The corresponding grantable privilege
- DBADMIN on the specified DB
- SYSADMIN on the specified DB

You must be connected to the system dictionary when you issue the statement.

GRANT Physical Database Definition Privileges Syntax

```
GRANT [ALTER | CREATE | DISPLAY | DROP | USE] DEFINE
```
GRANT Physical Database Definition Privileges Parameters

- **DEFINE**
  Gives the ALTER, CREATE, DISPLAY, DROP, and USE privileges, as applicable, on the resource identified in the ON parameter to the users or groups identified in the TO parameter.

- **ALTER**
  Gives the ALTER privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  The ALTER privilege on a resource allows a user to modify the definition of the resource. The ALTER privilege on a DMCL or database name table also allows a user to generate a load module from the definition.

- **CREATE**
  Gives the CREATE privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  The CREATE privilege on a resource allows a user to define the resource.

- **DISPLAY**
  Gives the DISPLAY privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  The DISPLAY privilege allows the user to issue a DISPLAY RESOURCE statement on the named resource. The grantable DISPLAY privilege allows a user to issue a DISPLAY PRIVILEGES statement on the named resource.
  The DISPLAY privilege on a DBTABLE resource is required for a user to produce a DBTABLE listing using IDMSRPTS. The DISPLAY privilege on a DMCL resource is required for a user to produce a DMCL listing using IDMSRPTS. The DISPLAY privilege on a DB resource is required for a user to produce a segment listing using IDMSRPTS.

- **DROP**
  Gives the DROP privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  The DROP privilege on a resource allows a user to delete the definition of the resource.

- **USE**
  Gives the USE privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.

  - **DMCL** -- The USE privilege allows a user to format, print, archive, and fix the journal files defined by the DMCL and punch the DMCL load module.
- **Database name table** -- The USE privilege allows a user to punch the database name table load module and specify the database name table in the DBTABLE parameter of a DMCL definition.

- **Segment** -- The USE privilege allows a user to associate the segment with an SQL schema.

- **ON**
  Specifies the resource to which the definition privileges apply.

- **DMCL dmcl-name**
  Identifies a DMCL.
  The scope of a privilege granted on a DMCL resource includes these physical database definition statements:
  - DMCL
  - BUFFER
  - JOURNAL BUFFER
  - ARCHIVE JOURNAL
  - DISK JOURNAL
  - TAPE JOURNAL

  You can wildcard `dmcl-name`.

  > **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **DBTABLE dbtable-name**
  Identifies a database name table.
  The scope of a privilege granted on a DBTABLE resource includes these physical database definition statements:
  - DBTABLE
  - DBNAME

  You can wildcard `dbtable-name`.

  > **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **DB database-name**
  Identifies a segment or a name in the database name table.
  The scope of a privilege granted on a DB resource includes these physical database definition statements:
You can wildcard database-name.

**Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- TO
  Specifies the users or groups to whom you are giving definition privileges.

- PUBLIC
  Specifies all users.

- authorization-identifier
  Identifies a user or group.

**Note:** Expanded syntax for authorization-identifier is presented in section Notes on Security Statement Syntax (see page 178).

- WITH GRANT OPTION
  Gives the authority to grant the specified definition privileges on the named resource to the users or groups identified in the TO parameter. Only a holder of the applicable DBADMIN privilege or a holder of SYSADMIN privilege can specify WITH GRANT OPTION.
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

### GRANT Physical Database Definition Privileges Usage

#### The DEFINE Keyword

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all of the privileges in the set that have been previously granted to the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user as a way to grant all but one definition privilege.

*Security Considerations for IDMSRPTS*
If a dictionary named in an IDMSRPTS run has been secured, the user who submits the job must have EXECUTE privilege on the category containing the run unit dictionary-name.IDMSNWKG.IDMSRPTS. Additional privileges may be required depending on the reports requested:

<table>
<thead>
<tr>
<th>Report</th>
<th>Privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBTLST (DBTABLE listing)</td>
<td>DBADMIN on the dictionary or DISPLAY on the DBTABLE</td>
</tr>
<tr>
<td>DMCLST (DMCL listing)</td>
<td>DBADMIN on the dictionary or DISPLAY on the DMCL</td>
</tr>
<tr>
<td>SEGLST (segment listing)</td>
<td>DBADMIN on the dictionary or DISPLAY on the DB</td>
</tr>
<tr>
<td>All other reports</td>
<td>Governed by application dictionary security</td>
</tr>
</tbody>
</table>

Note: For more information, see Securing Application Dictionary Resources (see page 121).

Granting Privilege to Issue DMCL Statements

The following statement gives the users the privilege to issue DMCL definition statements for DMCL99:

```
grant define
  on dmcl dmcl99
  to mike, ryan;
```

GRANT Physical Database Definition Privileges More Information

For more information on revoking privilege to define physical database resources, see REVOKE Physical Database Definition Privileges (see page 268).

GRANT SQL Definition Privileges

Contents

- GRANT SQL Definition Privileges Authorization (see page 254)
- GRANT SQL Definition Privileges Syntax (see page 254)
- GRANT SQL Definition Privileges Parameters (see page 254)
- GRANT SQL Definition Privileges Usage (see page 256)
- GRANT SQL Definition Privileges More Information (see page 257)

Gives one or more users or groups the privilege of performing definition functions on a specified access module, schema, table, function, procedure, table procedure, or view.
GRANT SQL Definition Privileges Authorization

To grant definition privileges on SQL-defined database resources, one of the following must be true:

- You hold the corresponding grantable privilege on the resource (you can grant the privilege, but you cannot specify WITH GRANT OPTION).
- You own the resource.
- You hold DBADMIN on the dictionary containing the definition.
- You hold SYSADMIN privilege.

GRANT SQL Definition Privileges Syntax

```
GRANT DEFINE \n    ALTER, CREATE, DROP, REFERENCES \n    ON ACCESS MODULE access-module-name \n    SCHEMA schema-name \n    table table-name \n    TO PUBLIC, authorization-identifier \n    WITH GRANT OPTION
```

GRANT SQL Definition Privileges Parameters

- **DEFINE**
  Gives the ALTER, CREATE, DISPLAY, and DROP privileges, as applicable, on the resource identified in the ON parameter to the users or groups identified in the TO parameter.

- **ALTER**
  Gives the ALTER privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  The ALTER privilege on a resource allows a user to modify the definition of the resource.

- **CREATE**
  Gives the CREATE privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
  The CREATE privilege on a resource allows a user to define the resource.

- **DISPLAY**
  Gives the DISPLAY privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter.
The DISPLAY privilege allows the user to issue a DISPLAY RESOURCE statement on the named resource. The grantable DISPLAY privilege allows a user to issue a DISPLAY PRIVILEGES statement on the named resource. The DISPLAY privilege on access-module-name also allows a user to execute the EXPLAIN statement on the access module.

- **DROP**
  Gives the DROP privilege on the resource identified in the ON parameter to the users or groups identified in the TO parameter. The DROP privilege on a resource allows a user to delete the definition of the resource.

- **REFERENCES**
  Gives the REFERENCES privilege on the table identified in the ON parameter to the users or groups identified in the TO parameter. The REFERENCES privilege on a table allows a user to define referential constraints in which the named table is the referenced table.

  **Note:** The REFERENCES privilege applies only to tables. It does not apply to functions, procedures, table procedures, views, access modules, or schemas.

- **ON**
  Specifies the resource to which the definition privileges apply.

- **ACCESS MODULE**
  Specifies that the privileges apply to any version of access-module-name in the associated schema.

- **access-module-name**
  Identifies the access module. You can wildcard access-module-name. If you specify schema-name, the wildcard character is valid after the period following schema-name.

  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **SCHEMA schema-name**
  Identifies an SQL schema. You can wildcard schema-name in the SCHEMA parameter.

  **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

- **table table-name**
  Identifies a table-like object
You can wildcard any identifier when you grant definition privileges on `table-name`. If you specify `schema-name` in `table-name`, the wildcard character is valid after the period following `schema-name`.

**Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

If you do not specify `schema-name`, it defaults to the current schema in effect for your session.

- **TO**
  Specifies the users to whom you are giving the definition privileges.

- **PUBLIC**
  Specifies all users.

- **authorization-identifier**
  Identifies a user or group.

**Note:** Expanded syntax for `authorization-identifier` is presented in Notes on Security Statement Syntax (see page 178).

- **WITH GRANT OPTION**
  Gives the privilege of granting the specified definition privileges on the named resource to the users or groups identified in the TO parameter.
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

**GRANT SQL Definition Privileges Usage**

**Wildcarding Table-Name**

For table definition privileges, only the closest matching wildcarded grant is used. If CREATE privilege has been granted on HR.EMP* and HR.EMPV*, then only the grant on HR.EMPV* is used to verify the privilege to create HR.EMPVU_SALARY.

**The DEFINE Keyword**

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all of the privileges in the set that have been previously granted to the specified users or groups.
This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user as a way to grant all but one definition privilege.

Granting Privileges on a Schema

The following GRANT statement gives the ALTER, CREATE, DISPLAY, and DROP privileges on all schemas that begin with 'DSF' to user DSF. The statement also gives user DSF the privilege of granting the same privileges to other users.

```
grant define
    on schema dsf*
    to dsf
    with grant option;
```

GRANT SQL Definition Privileges More Information

For more information on revoking SQL definition privileges, see REVOKE SQL Definition Privileges (see page 271).

GRANT Table Access Privileges

Contents

- GRANT Table Access Privileges Authorization (see page 257)
- GRANT Table Access Privileges Syntax (see page 258)
- GRANT Table Access Privileges Parameters (see page 258)
- GRANT Table Access Privileges Usage (see page 259)
- GRANT Table Access Privileges More Information (see page 260)

Gives one or more users or groups the privilege of accessing a specified table-like object in a specified way.

GRANT Table Access Privileges Authorization

To grant table-like object access privileges, one of the following must be true:

- You hold the corresponding grantable privilege on the table-like object (you can grant the privilege, but you cannot specify WITH GRANT OPTION).
- You own the table-like object.
- You hold DBADMIN privilege on the database that contains the table-like object data.
You hold SYSADMIN privilege.

**GRANT Table Access Privileges Syntax**

```
GRANT ACCESS
   DELETE
   INSERT
   SELECT
   UPDATE

ON table table-name

TO PUBLIC
   authorization-identifier
   WITH GRANT OPTION
```

**GRANT Table Access Privileges Parameters**

- **ACCESS**
  Gives the DELETE, INSERT, SELECT, and UPDATE privileges on the table-like object identified in the ON parameter to the users or groups identified in the TO parameter.

- **DELETE**
  Gives the DELETE privilege on the table-like object identified in the ON parameter to the users or groups identified in the TO parameter.
  The DELETE privilege on a table-like object allows a user to delete rows from the table or view.

- **INSERT**
  Gives the INSERT privilege on the table-like object identified in the ON parameter to the users or groups identified in the TO parameter.
  The INSERT privilege on a table-like object allows a user to insert rows into the table or view.

- **SELECT**
  Gives the SELECT privilege on the table-like object identified in the ON parameter to the users or groups identified in the TO parameter.
  The SELECT privilege on a table-like object allows a user to the following:
  - Retrieve data from the table-like object.
  - Name the table-like object in a subquery.
  - Define a view derived from the table-like object.

- **UPDATE**
  Gives the UPDATE privilege on the table-like object identified in the ON parameter to the users or groups identified in the TO parameter.
  The UPDATE privilege on a table-like object allows a user to modify data in the table-like object.

- **ON table table-name**
  Specifies the table-like object to which the table access privileges apply.
You can wildcard any identifier when you grant access on table-name. If you specify schema-name in table-name, the wildcard character is valid after the period following schema-name.

If you do not specify schema-name, it defaults to the current schema in effect for your session.

- **TO**
  Specifies the users or groups to whom you are giving table access privileges.

- **PUBLIC**
  Specifies all users.

- **authorization-identifier**
  Identifies a user or group.

- **WITH GRANT OPTION**
  Gives the privilege of granting the specified access privileges on the named table or view to the users or groups identified in the TO parameter.
  A privilege granted with the WITH GRANT OPTION is called a grantable privilege.

**GRANT Table Access Privileges Usage**

**Verifying Privileges Granted with Wildcards**

For table access privileges, all matching wildcarded grants are used. If SELECT privilege has been granted on HR.EMP* and HR.EMPV*, then users or groups receiving either the HR.EMP* or the HR.EMPV* grant are authorized to select from EMPVU_SALARY.

This differs from the use of wildcards for all other types of resources or privileges, where only the closest matching wildcarded grant of privilege is used to verify the user's authorization.

**The ACCESS Keyword**

When you use the ACCESS keyword with a GRANT statement, you grant a set of access privileges on a table-like object to one or more users or groups.

When you use the ACCESS keyword with a REVOKE statement, you revoke any access privileges that have been previously granted on the table-like object from the specified users or groups.
This means that if you GRANT SELECT privilege on a table, you can revoke the privilege with either a REVOKE SELECT statement or a REVOKE ACCESS statement. Using REVOKE ACCESS is an efficient technique when you intend to revoke all access privileges on a table from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT ACCESS on a table to a user and then REVOKE DELETE on the table from the same user as a way to grant all but one table access privilege.

Granting Selected Privileges on a Table

The following GRANT statement gives the SELECT and UPDATE privileges on the EMPLOYEE table associated with the current schema to users KRP, SAE, and PGD:

grant select, update
  on employee
  to krp, sae, pgd;

GRANT Table Access Privileges More Information

For more information on revoking table access privileges, see REVOKE Table Access Privileges (see page 274).

REVOKE Access Module Execution Privilege

Contents

- REVOKE Access Module Execution Privilege Authorization (see page 260)
- REVOKE Access Module Execution Privilege Syntax (see page 261)
- REVOKE Access Module Execution Privilege Parameters (see page 261)

Revolves from one or more users or groups the privilege of executing a specified access module.

REVOKE Access Module Execution Privilege Authorization

To revoke access module execution privilege, one of the following must be true:

- You hold grantable execution privilege on the access module (you can grant the privilege, but you cannot specify WITH GRANT OPTION).
- You own the schema associated with the access module.
- You hold DBADMIN privilege on the dictionary that contains the access module.
- You hold SYSADMIN privilege.

You must be connected to the application dictionary that contains the access module when you issue the statement.
REVOKE Access Module Execution Privilege Syntax

REVOKE EXECUTE

ON ACCESS MODULE schema-name access-module-name

FROM PUBLIC authorization-identifier

REVOKE Access Module Execution Privilege Parameters

- **EXECUTE**
  Specifies that you are revoking execution privilege on the access module identified in the ON parameter from the users or group identified in the FROM parameter.

- **ON ACCESS MODULE**
  Specifies the access module to which execution privilege applies.

- **access-module-name**
  Identifies the access module. You can wildcard access-module-name. If you specify schema-name, the wildcard character is valid after the period following schema-name.

  Note: For more information on wildcarding, see Using a Wildcard (see page 57).

- **FROM**
  Specifies the users or groups from whom you are revoking execution privilege.

- **PUBLIC**
  Specifies all users. The privilege must have been previously given to PUBLIC by means of the GRANT statement.

- **authorization-identifier**
  Identifies a user or group. The privilege must have been previously given to authorization-identifier by means of the GRANT statement.

  Note: Expanded syntax for authorization-identifier is presented in section Notes on Security Statement Syntax (see page 178).

Revoking Execution Privilege

The following statement revokes execution privilege on all access modules associated with schema HR that begin with 'HR.EMP' from group PER_GRP_2:

```sql
revoke execute
  on access module hr.emp*
  from per_grp_2;
```
For more information on granting execution privilege, see GRANT Access Module Execution Privilege (see page 238).

REVOKE Administration Privilege 1

Revolves from one or more users or groups the DBADMIN privilege.

Authorization

To revoke DBADMIN privilege, you must hold one of these privileges:

- DBADMIN on the database
- SYSADMIN

You must be connected to the system dictionary when you issue the statement.

Syntax

```sql
REVOKE DBADMIN ON DB database-name
FROM PUBLIC authorization-identifier
```

Parameters

- **DBADMIN**
  Specifies that you are revoking DBADMIN privilege on the database identified in the ON parameter from the users or groups identified in the FROM parameter.

- **ON DB database-name**
  Specifies database to which DBADMIN privilege applies. `Database-name` refers to either a segment or a name in the database name table. For more information on the DBADMIN privilege, see Securing Database Resources (see page 88).

  You can wildcard `database-name`. For more information on wildcards, see Using a Wildcard (see page 57).

- **FROM**
  Specifies the users or groups from whom you are revoking DBADMIN privilege.

- **PUBLIC**
  Specifies all users. The privilege must have been previously given to PUBLIC by means of the GRANT statement.
authorization-identifier
Identifies a user or group.
The privilege must have been previously given to authorization-identifier by means of the GRANT statement.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

Revoking DBADMIN From the DBA

The following statement revokes DBADMIN privilege on database GLDB from the DBA group ID:

```
revoke dbadmin
on db gldb
from dba_gldb;
```

REVOKE All Table Privileges

Contents

- Authorization (see page 263)
- Syntax (see page 263)
- Parameters (see page 264)
- More Information (see page 264)

Revolves from one or more users or groups all definition and access privileges on a specified table-like object.

Authorization

To revoke all table privileges, one of the following must be true:

- You hold all privileges as grantable privileges on the table-like object.
- You own the table-like object.
- You hold DBADMIN privileges for the dictionary that contains the table-like object definition and the database that contains the table-like object data.
- You hold SYSADMIN privilege.

Syntax

```
REVOKE ALL PRIVILEGES
ON table table-name
```
Parameters

- **ALL PRIVILEGES**
  Revokes the DELETE, INSERT, SELECT, UPDATE, ALTER, CREATE, DROP, and REFERENCES privileges, as applicable, on the table-like object identified in the ON parameter from the users or groups identified in the FROM parameter.

- **ON table table-name**
  Identifies the table-like object to which the privileges apply.

  **Note:** Expanded syntax for `table-name` is presented in Notes on Security Statement Syntax. You can wildcard the identifier of `table-name`. If you specify `schema-name` in `table-name`, the wildcard character is valid after the period following `schema-name`. **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

If you do not specify `schema-name`, it defaults to the current schema in effect for your session.

- **FROM**
  Specifies the users or groups from whom you are revoking table privileges.

- **PUBLIC**
  Specifies all users.
  The privileges must have been previously given to PUBLIC by means of the GRANT statement.

- **authorization-identifier**
  Identifies a user or group.
  The privileges must have been previously given to `authorization-identifier` by means of the GRANT statement.

  **Note:** Expanded syntax for `authorization-identifier` is presented in Notes on Security Statement Syntax (see page 178).

*Revoking All Privileges From All Users*

The following statement revokes all privileges on all table-like objects in the TEST schema from the group PUBLIC:

```
revoke all privileges on test.*
  from public;
```
For more information on table access privileges, see **GRANT Table Access Privileges** (see page 257) and **REVOKE Table Access Privileges** (see page 274).

For more information on table definition privileges, see **GRANT SQL Definition Privileges** (see page 253) and **REVOKE SQL Definition Privileges** (see page 271).

### REVOKE Area Access Privileges

#### Contents

- Authorization (see page 265)
- Syntax (see page 265)
- Parameters (see page 265)

Revolves from one or more users or groups access to an area.

#### Authorization

To revoke an area access privilege, you must hold one of these privileges:

- The grantable area access privilege on the area
- DBADMIN on DB `segment-name`
- SYSADMIN

You must be connected to the system dictionary when you issue the statement.

#### Syntax

```plaintext
REVOKE DBAREAD, DBWRITE, USE
    ON AREA segment-name.area-name
    FROM PUBLIC authorization-identifier
```

#### Parameters

- **DBAREAD**
  
  Specifies that you are revoking DBAREAD privilege on the area named in the ON parameter from the users or groups named in the FROM parameter.
DBAWRITE
Specifies that you are revoking DBAWRITE privilege on the area named in the ON parameter from the users or groups named in the FROM parameter.

USE
Specifies that you are revoking USE privilege on the area named in the ON parameter from the users or groups named in the FROM parameter.

ON AREA segment-name.area-name
Identifies the area to which the specified area access privileges apply. You can wildcard area-name when you revoke area access privileges. You cannot wildcard segment-name. The wildcard character is valid after the period following segment-name.

Note: For more information on wildcarding, see Using a Wildcard (see page 57).

FROM
Specifies the users or groups from whom you are revoking area access privileges.

PUBLIC
Specifies all users. The privileges must have been previously given to PUBLIC by means of the GRANT statement.

authorization-identifier
Identifies a user or group. The privileges must have been previously given to authorization-identifier by means of the GRANT statement.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

Revoking Area Access Privileges

The following statement revokes an area access privilege from the specified user:

```
revoke dbawrite
    on area gl."account-area"
    from alex;
```

Note: For more information on granting the privilege to access an area, see GRANT Area Access Privileges (see page 245).
REVOKE Non-SQL Definition Privilege

Contents

- Authorization (see page 267)
- Syntax (see page 267)
- Parameters (see page 267)
- More Information (see page 268)

Revolves from one or more users or groups the privilege of referencing a non-SQL-defined schema in an SQL schema definition.

Authorization

To revoke USE privilege on a non-SQL-defined schema, you must hold one of these privileges:

- Grantable USE privilege on the non-SQL-defined schema
- DBADMIN on the dictionary containing the non-SQL schema definition
- SYSADMIN

You must be connected to the dictionary containing the non-SQL-defined schema when you issue the statement.

Syntax

```
REVOKE USE DISPLAY
ON NONSQL SCHEMA Vnnnn.nonsql-schema-name
FROM PUBLIC authorization-identifier
```

Parameters

- **USE**
  Specifies that you are revoking the USE privilege on the non-SQL-defined schema identified in the ON parameter to the users or groups identified in the FROM parameter.

- **DISPLAY**
  Specifies that you are revoking the DISPLAY privilege on the non-SQL-defined schema identified in the ON parameter to the users or groups identified in the FROM parameter.

- **ON NONSQL SCHEMA**
  Specifies the non-SQL-defined schema to which the USE privilege applies.
**Vnnnn.nonsql-schema-name**
Specifies the version number and name of the non-SQL-defined schema. The version number (nnnn) must include leading zeros.
You can wildcard nonsql-schema-name. You cannot wildcard Vnnnn. The wildcard character is valid after the period following Vnnnn.

⚠️ **Note:** For more information on wildcarding, see Using a Wildcard (see page 57).

**FROM**
Specifies the users or groups from whom you are revoking the USE privilege.

**PUBLIC**
Specifies all users.
The privilege must have been previously given to PUBLIC by means of the GRANT statement.

**authorization-identifier**
Identifies a user or group.
The privilege must have been previously given to authorization-identifier by means of the GRANT statement.

⚠️ **Note:** Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

*Revoking Use of a Non-SQL-Defined Schema*

The following statement revokes the privilege of referencing a non-SQL-defined schema when creating an SQL schema:

```sql
revoke use
  on nonsql schema v0001.ap
from sam;
```

**More Information**

For more information about granting the privilege to use a non-SQL-defined schema, see GRANT Non-SQL Definition Privilege (see page 247).

**REVOKE Physical Database Definition Privileges**

**Contents**

- Authorization (see page 269)
- Syntax (see page 269)
- Parameters (see page 269)
Revoke from one or more users or groups the privilege of executing the DMCL, DBTABLE, and SEGMENT physical DDL statements.

**Authorization**

To revoke a definition privilege on a DMCL or DBTABLE, you must hold one of these privileges:

- The corresponding grantable privilege
- DBADMIN on DB SYSTEM
- SYSADMIN

To revoke a physical definition privilege on a database, you must hold one of these privileges:

- The corresponding grantable privilege
- DBADMIN on the database
- SYSADMIN

You must be connected to the system dictionary when you issue the statement.

**Syntax**

```
REVOKE DEFINE ─┬─ ALTER ─┐
│ ┌─────── , ─┐ │
└─▼─┬─ CREATE ─┘  └─▼─┬─ DISPLAY ─┘
        ┌─▼─┬─ DROP ─┘        ┌─▼─┬─ USE ─┘
        └─▼─┬─ USE ─┘          └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        │                  ┌─▼─┬─ USE ─┘
        │                  └─▼─┬─ USE ─┘
        └─────────────── , ───────────────┘
```

**Parameters**

- **DEFINE**
  Revoke the ALTER, CREATE, DISPLAY, DROP, and USE privileges on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **ALTER**
  Revoke the ALTER privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.
CREATE
Revokes the CREATE privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

DISPLAY
Revokes the DISPLAY privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

DROP
Revokes the DROP privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

USE
Revokes the USE privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

ON
Specifies the resource to which the definition privileges apply.

DMCL *dmcl-name*
Identifies a DMCL.
You can wildcard *dmcl-name*.

**Note:** For more information on wildcarding, see *Using a Wildcard (see page 57)*.

DBTABLE *dbtable-name*
Identifies a database name table.
You can wildcard *dbtable-name*.

**Note:** For more information on wildcarding, see *Using a Wildcard (see page 57)*.

DB *database-name*
Identifies a segment or a name in the database name table.
You can wildcard *database-name*.

**Note:** For more information on wildcarding, see *Using a Wildcard (see page 57)*.

FROM
Specifies the users or groups from whom you are revoking definition privileges.

PUBLIC
Specifies all users.
The privileges must have been previously given to PUBLIC by means of the GRANT statement.
authorization-identifier
Identifies a user or group.
The privileges must have been previously given to authorization-identifier by means of the GRANT statement.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

Usage

The DEFINE Keyword

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges on a resource to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all definition privileges that have been previously granted on the resource from the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges on the resource from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user as a way to grant all but one definition privilege.

Revoking Privilege to Issue DMCL Statements

The following statement revokes from the user the privilege to issue DMCL definition statements for DMCL99:

```
revoke define
  on dmcl dmcl99
  from ryan;
```

More Information

For more information about granting privilege to define physical database resources, see GRANT Physical Database Definition Privileges (see page 249).

REVOKE SQL Definition Privileges

Contents

- Authorization (see page 272)
Revolves from one or more users or groups the privilege of performing selected actions on a specified schema, access module, or table-like object.

Authorization

To revoke a definition privilege for an SQL-defined database resource, one of the following must be true:

- You hold the grantable definition privilege on the resource
- You own the resource
- You hold DBADMIN privilege on the dictionary containing the definitions
- You hold SYSADMIN privilege

Syntax

```
REVOKE
  DEFINE  
  │   ALTER  CREATE  DISPLAY  DROP  REFERENCES  │
  │  ON  ACCESS MODULE  SCHEMA  schema-name  access-module-name  │
  │              table  table-name
  │
  FROM  PUBLIC  authorization-identifier
```

Parameters

- **DEFINE**
  Revokes the ALTER, CREATE, DISPLAY, and DROP privileges on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **ALTER**
  Revokes the ALTER privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

- **CREATE**
  Revokes the CREATE privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.
DISPLAY
Revoke the DISPLAY privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

DROP
Revoke the DROP privilege on the resource identified in the ON parameter from the users or groups identified in the FROM parameter.

REFERENCES
Revoke the REFERENCES privilege on the table identified in the ON parameter from the users or groups identified in the FROM parameter.

ON
Specifies the resource to which the definition privileges apply.

ACCESS MODULE access-module-name
Identifies an access module.
Privileges on any version of access-module-name in the associated schema are revoked.
You can wildcard access-module-name. If you specify schema-name, the wildcard character is valid after the period following schema-name.

Note: For more information on wildcarding, see Using a Wildcard (see page 57).

SCHEMA schema-name
Identifies an SQL schema.
You can wildcard schema-name.

Note: For more information on wildcarding, see Using a Wildcard (see page 57).

table table-name
Identifies a table-like object.
You can wildcard any identifier in table-name when you grant definition privileges on table-name.
If you specify schema-name in table-name, the wildcard character is valid after the period following schema-name.

Note: For more information on wildcarding, see Using a Wildcard (see page 57).

If you do not specify schema-name, it defaults to the current schema in effect for your session.

FROM
Specifies the users or groups from whom you are revoking definition privileges.

PUBLIC
Specifies all users.
The privileges must have been previously given to PUBLIC by means of the GRANT statement.
- **authorization-identifier**
  Identifies a user or group. The privileges must have been previously given to `authorization-identifier` by means of the GRANT statement.

  **Note:** Expanded syntax for `authorization-identifier` is presented in Notes on Security Statement Syntax (see page 178).

### Usage

#### The DEFINE Keyword

When you use the DEFINE keyword with a GRANT statement, you grant a set of definition privileges on a resource to one or more users or groups.

When you use the DEFINE keyword with a REVOKE statement, you revoke all definition privileges that have been previously granted on the resource from the specified users or groups.

This means that if you GRANT CREATE privilege on a resource, you can revoke the privilege with either a REVOKE CREATE statement or a REVOKE DEFINE statement. Using REVOKE DEFINE is an efficient technique when you intend to revoke all definition privileges on the resource from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT DEFINE on a resource to a user and then REVOKE DROP on the resource from the same user as a way to grant all but one definition privilege.

#### Revoking Privileges on a Schema

The following statement revokes the ALTER, CREATE, DISPLAY, and DROP privileges on all schemas that begin with 'DSF' from user DSF:

```sql
revoke define
  on schema dsf*
from dsf;
```

### More Information

For more information about granting SQL definition privileges, see GRANT SQL Definition Privileges (see page 253).

### REVOKE Table Access Privileges

#### Contents

- Authorization (see page 275)
- Syntax (see page 275)
Revokes from one or more users or groups the privilege of accessing a specified table-like object in a specified way.

**Authorization**

To revoke a table-like object access privilege, one of the following must be true:

- You hold the corresponding grantable privilege on the table-like object.
- You own the table-like object.
- You hold DBADMIN privilege on the database that contains the table-like object.
- You hold SYSADMIN privilege.

**Syntax**

```
REVOKE ACCESS [DELETE | INSERT | SELECT | UPDATE] ON table table-name
FROM PUBLIC . authorization-identifier
```

**Parameters**

- **ACCESS**
  Revokes the DELETE, INSERT, SELECT, and UPDATE privileges on the table-like object identified in the ON parameter from the users or groups identified in the FROM parameter.

- **DELETE**
  Revokes the DELETE privilege on the table-like object identified in the ON parameter from the users or groups identified in the FROM parameter.

- **INSERT**
  Revokes the INSERT privilege on the table-like object identified in the ON parameter from the users or groups identified in the FROM parameter.

- **SELECT**
  Revokes the SELECT privilege on the table-like object identified in the ON parameter from the users or groups identified in the FROM parameter.
UPDATE
Revokes the UPDATE privilege on the table-like object identified in the ON parameter from the users or groups identified in the FROM parameter.

ON table table-name
Identifies the table-like object to which the access privileges apply.

Note: Expanded syntax for table-name is presented in Notes on Security Statement Syntax.

You can wildcard any identifier in table-name when you grant access on table-name. If you specify schema-name in table-name, the wildcard character is valid after the period following schema-name.

Note: For more information on wildarding, see Using a Wildcard (see page 57).

If you do not specify schema-name, it defaults to the current schema in effect for your session.

FROM
Specifies the users or groups from whom you are revoking the specified table access privileges.

PUBLIC
Specifies all users.
The privileges must have been previously given to PUBLIC by means of the GRANT statement.

authorization-identifier
Identifies a user or group.
The privileges must have been previously given to authorization-identifier by means of the GRANT statement.

Note: Expanded syntax for authorization-identifier is presented in Notes on Security Statement Syntax (see page 178).

Usage

The ACCESS Keyword

When you use the ACCESS keyword with a GRANT statement, you grant a set of access privileges on a table-like object to one or more users or groups.

When you use the ACCESS keyword with a REVOKE statement, you revoke any access privileges that have been previously granted on the table or view from the specified users or groups.
This means that if you GRANT SELECT privilege on a table, you can revoke the privilege with either a REVOKE SELECT statement or a REVOKE ACCESS statement. Using REVOKE ACCESS is an efficient technique when you intend to revoke all access privileges on a table from a user or group, whether the privileges were granted singly or as a set.

Similarly, you can GRANT ACCESS on a table to a user and then REVOKE DELETE on the table from the same user as a way to grant all but one table access privilege.

**Revoking Selected Privileges on a Table**

The following statement revokes the SELECT and UPDATE privileges on the EMPLOYEE table associated with the current schema from users KRP, SAE, and PGD:

```
revoke select, update
  on employee
  from krp, sae, pgd;
```

**More Information**

For more information about granting table access privileges, see GRANT Table Access Privileges (see page 257).

**TRANSFER OWNERSHIP**

Passes ownership of an SQL schema from one user or group to another user or group.

- Authorization (see page 277)
- Syntax (see page 277)
- Parameters (see page 278)
- Usage (see page 278)

**Authorization**

To transfer ownership, one of the following must be true:

- You own the schema
- You hold DBADMIN privilege on the dictionary in which the schema is defined
- You hold SYSADMIN privilege

**Syntax**

```
转移所有权到 SCHEMA schema-name
```

```
转移所有权到 authorization-identifier
```
Parameters

- **OF SCHEMA schema-name**
  Specifies the schema whose ownership is being transferred to the user or group identified in the TO parameter.
  *Schema-name* must identify a schema defined in the dictionary.

- **TO authorization-identifier**
  Identifies a user or group.

---

**Note:** Expanded syntax for *authorization-identifier* is presented in Notes on Security Statement Syntax (see page 178).

---

Usage

*Schema Ownership*

At any given time, a schema can be owned by one user or group. The initial owner is the user who created the schema. When ownership of a schema is transferred to a group, each user in the group has all the privileges associated with ownership.

*Ownership of Other Resources*

Technically, schemas are the only database resources that users own. However, by association, the user or group that owns a schema is also said to own the tables, functions, procedures, table procedures, views, and access modules in the schema.

*Ownership Privileges*

The owner of a schema has all applicable privileges on resources in the schema, as well as the privilege of granting those privileges to other users or groups. If you transfer ownership of a schema to another user or group, you no longer own the resources in the schema.

*Transferring Ownership to a Single User*

The following TRANSFER OWNERSHIP statement transfers ownership of the PKE_SCH schema to user PKE:

```plaintext
transfer ownership of schema pke_sch to pke;
```

*Transferring Ownership to a Group*

The following TRANSFER OWNERSHIP statement transfers ownership of the SALES schema to the SALES_GRP group:

```plaintext
transfer ownership of schema sales to sales_grp;
```
Syntax for Security Display Statements

This section explains the syntax, parameters, examples, and usage of the following Statements:

- Notes on DISPLAY/PUNCH Statement Syntax (see page 279)
- DISPLAY SYSADMIN PRIVILEGES (see page 281)
- DISPLAY PRIVILEGES on a Global Resource (see page 283)
- DISPLAY GROUP (see page 284)
- DISPLAY USER (see page 287)
- DISPLAY USER PROFILE (see page 290)
- DISPLAY DCADMIN PRIVILEGES (see page 291)
- DISPLAY PRIVILEGES on a System Resource (see page 293)
- DISPLAY RESOURCE (System) (see page 295)
- DISPLAY PRIVILEGES on a Database Resource (see page 297)
- DISPLAY RESOURCE (Database) (see page 300)

Notes on DISPLAY/PUNCH Statement Syntax

Contents

- Usage (see page 280)

How to Submit Statements

You submit security DISPLAY statements through the CA IDMS Command Facility in online mode or batch mode.

What DISPLAY Statements Do

DISPLAY statements enable you to display resource definitions in a CA IDMS centralized security database and privileges granted on resources.

What PUNCH Statements Do

PUNCH statements submitted online display the output online as DISPLAY statements do.

PUNCH statements submitted in batch mode write the output to the SYSPCH file.

Common Parameters

The following parameters are valid in all DISPLAY statements:

- WITH display-option
  Specifies that output is to include information represented by display-option.

- ALSO WITH display-option
  Specifies that output is also to include information represented by display-option.
• **WITHOUT display-option**
  Specifies that output is to exclude information represented by `display-option`.

• **AS SYNTAX**
  Specifies that lines of output representing executable syntax do not begin with comment characters.

• **AS COMMENTS**
  Specifies that lines of output representing executable syntax begin with comment characters.

**Usage**

**Wildcards**

If a wildcarded name was used in the resource definition or the grant of privilege on the resource, the DISPLAY statement must specify the wildcarded name.

**Correspondence to DDDL Syntax**

The preceding common parameters for DISPLAY statements have the same function that they have in DDDL syntax. In general, the conventions of security DISPLAY statement syntax will be familiar to the DDDL user.

⚠️ **Note:** For more information on DDDL, see the *CA IDMS IDD DDDL Reference section*.

**Specifying AS SYNTAX**

This example shows the output of a security DISPLAY statement that specifies AS SYNTAX:

```sql
DISPLAY USER DICK WITH ALL AS SYNTAX;
CREATE USER DICK
  *** USER IS ACTIVE
  *** DESCRIPTION 'CHIEF ANALYST'
  *** NAME 'RICHARD A. ANALYST'
  *** PASSWORD ASSIGNED
  *** PROFILE PAYROLL
  *** CREATED 1991-07-18-12.51.17.187373 BY KKS
  *** LAST UPDATED 1991-07-18-12.51.23.000517 BY KKS
  *** WITHIN GROUP DEVTEAM
  *** HOLDS DEFINE PRIVILEGES ON USER TOM
  *** HOLDS DEFINE PRIVILEGES ON SYSTEM SYSTEM71
  *** HOLDS SIGNON PRIVILEGES ON SYSTEM SYSTEM71
  *** HOLDS DBADMIN PRIVILEGES ON DB FRED
  *** HOLDS DBWRITE PRIVILEGES ON AREA THESEGMENT.CORPTSP
  *** HOLDS USE PRIVILEGES ON DMCL THEDMCL
  *** HOLDS USE PRIVILEGES ON DTABLE THEDBTABLE
  *** HOLDS USE PRIVILEGES ON DB THEDB
  *** HOLDS EXECUTE PRIVILEGES ON CATEGORY CATE3
  *** HOLDS EXECUTE PRIVILEGES ON ACTIVITY DCM.T002
  *** HOLDS REFERENCES PRIVILEGES ON ACCESS MODULE THESCHEMA.THEAM
  *** HOLDS EXECUTE PRIVILEGES ON ACCESS MODULE THESCHEMA.THEAM
  *** HOLDS ALL PRIVILEGES ON TABLE THESCHEMA.THE*
  *** HOLDS DISPLAY PRIVILEGES ON ACCESS MODULE THESCHEMA.*
  *** HOLDS DISPLAY PRIVILEGES ON ACCESS MODULE THESCHEMA.FRED*
  *** HOLDS REFERENCES PRIVILEGES ON TABLE THESCHEMA.TETHABLE
```
Specifying AS COMMENTS

This example shows the output of a security DISPLAY statement that specifies AS COMMENTS:

DISPLAY USER DICK WITH ALL AS COMMENTS;
  /*+   CREATE USER DICK
     *+     USER IS ACTIVE
     *+     DESCRIPTION 'CHIEF ANALYST'
     *+     NAME 'RICHARD A. ANALYST'
     *+     PASSWORD ASSIGNED
     *+     PROFILE PAYROLL
     *+     CREATED 1991-07-18-12.51.17.187373 BY KKS
     *+     LAST UPDATED 1991-07-18-12.51.23.000517 BY KKS
     *+     WITHIN GROUP DEVTEAM
     *+     HOLDS DEFINE PRIVILEGES ON USER TOM
     *+     HOLDS SIGNON PRIVILEGES ON SYSTEM SYSTEM71
     *+     HOLDS DBADM PRIVILEGES ON DB FRED
     *+     HOLDS DBWRITE PRIVILEGES ON AREA THESEGMENT.CORPTSP
     *+     HOLDS USE PRIVILEGES ON DMCL THEDMCL
     *+     HOLDS USE PRIVILEGES ON DTABLE THEDTABLE
     *+     HOLDS USE PRIVILEGES ON DB THEDB
     *+     HOLDS EXECUTE PRIVILEGES ON CATEGORY CATE3
     *+     HOLDS EXECUTE PRIVILEGES ON ACTIVITY DCMT.N002
     *+     HOLDS REFERENCES PRIVILEGES ON ACCESS MODULE THESCHEMA.THEAM
     *+     HOLDS EXECUTE PRIVILEGES ON ACCESS MODULE THESCHEMA.THEAM
     *+     HOLDS ALL PRIVILEGES ON TABLE THESCHEMA.THE*
     *+     HOLDS DISPLAY PRIVILEGES ON ACCESS MODULE THESCHEMA.*
     *+     HOLDS DISPLAY PRIVILEGES ON ACCESS MODULE THESCHEMA.FRED*
     *+     HOLDS REFERENCES PRIVILEGES ON TABLE THESCHEMA.TETABLE
     *+     HOLDS SELECT, INSERT, UPDATE PRIVILEGES ON TABLE THESCHEMA.TETABLE
     *+     HOLDS DISPLAY PRIVILEGES ON TABLE THESCHEMA.*

DISPLAY SYSADMIN PRIVILEGES

Contents

- Authorization (see page 281)
- Syntax (see page 281)
- Parameters (see page 282)
- Usage (see page 282)

Displays users who have been granted SYSADMIN privilege for the domain.

Authorization

To display SYSADMIN privilege, you must hold SYSADMIN privilege.

Syntax

```
    DISPLAY SYSADMIN privileges
```

10-Jan-2018  281/367
Parameters

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of the privilege and the user or group to whom it was granted.

- **HISTORY**
  Specifies display of the following:
  - The date and time the privilege was granted to the user or group, and the ID of the user granted the privilege.
  - The date and time the privilege was last updated, and the ID of the user who updated the privilege.

- **VERB**
  Specifies the verb used in the display output.
  If VERB is not specified, the privilege is displayed in the form of GRANT statements.
  - **GRANT**
    Specifies display of SYSADMIN privilege in the form of GRANT statements.
  - **REVOKE**
    Specifies substitution of "REVOKE" for "GRANT" in the display output.

Usage

*Specifying WITH VERB REVOKE*

By specifying WITH VERB REVOKE and AS SYNTAX, you display output that you can then submit to revoke privilege from those who hold the privilege. This is useful when you need to revoke a privilege from most or all users who hold it.
However, be sure to leave at least one user with SYSADMIN privilege.

DISPLAY PRIVILEGES on a Global Resource

Contents

- Authorization (see page 283)
- Syntax (see page 283)
- Parameters (see page 283)
- Usage (see page 284)

Displays privileges granted on a global resource and the users and groups who hold the privilege.

Authorization

To display privileges on a global resource, you must hold the grantable DISPLAY privilege on the resource or SYSADMIN privilege.

Syntax

```
►►─┬─ DISPlay PRIVileges on resource ─┬─ GROup group-identifier ─┬─ USER user-identifier ─┬─ USER PROfile profile-name
   └─ PUNch ─┬─ WITH display-option ─┼─ WITH user-identifier ─┼─ WITH group-identifier ─┘
   └─ WITH user-identifier ┘
   └─ WITH group-identifier ┘
   └─ WITHOUT display-option ┘
   └─ WITHOUT user-identifier ┘
   └─ WITHOUT group-identifier ┘
   └─ display-option ┘
   └─ VERB GRAnit ┘
   └─ VERB REVoke ┘
   └─ AS SYNTAX ┘
   └─ AS COMments ┘
   └─ ALL display-option ┘
   └─ NONE display-option ┘
   └─ HISTORY display-option ┘
```

Expansion of Display-Option

```
►►─┬─ ALL display-option ─┬─ NONe display-option ─┬─ HIStory display-option
   └─ display-option ┘
   └─ display-option ┘
   └─ display-option ┘
```

Parameters

- **GROup group-identifier**
  Identifies a group.

- **USER user-identifier**
  Identifies a user.
● **USER PROfile profile-name**
  Identifies a user profile.

● **ALL**
  Specifies all display options.

● **NONE**
  Specifies display of only the privileges and the authorization identifier to which they have been granted.

● **HISTORY**
  Specifies display of the following:

  ▪ The date and time privilege on the resource was granted to the user or group, and the ID of the user granted the privilege.

  ▪ The date and time privilege on the resource was last updated, and the ID of the user who updated the privilege.

● **VERB**
  Specifies the verb to be used in the display output.
  If VERB is not specified, privileges are displayed in the form of GRANT statements.

  ▪ **GRANT**
    Specifies that privileges are displayed in the form of GRANT statements.

  ▪ **REVOKE**
    Specifies substitution of "REVOKE" for "GRANT" in the display output.

**Usage**

*Specifying WITH VERB REVOKE*

By specifying WITH VERB REVOKE and AS SYNTAX, you display output that you can then submit to revoke privilege from those who hold the privilege. This is useful when you need to revoke a privilege from most or all users who hold it.

**DISPLAY GROUP**

**Contents**

- Authorization (see page 285)
- Syntax (see page 285)
- Parameters (see page 285)
- Usage (see page 287)

Displays information on a group.
Authorization

To display group information, you must hold SYSADMIN privilege.

Syntax

```
  DISPlay    GROup group-identifier
          PUNCH

  WITH       display-option
            ALSo WITh
            WITHOut

  AS        SYNTAX COMMENTS
```

Expansion of Display-Option

- ALL
- NONe
- DETails
- HISTory
- USErs
- all PRIVileges
- ACCESS Module privileges
- ACTivity privileges
- AREA privileges
- CATegory privileges
- DB privileges
- DBADMIN privileges
- DBTable privileges
- DCADMIn privileges
- DMCL privileges
- GROup privileges
- NONSQL SCHEMA privileges
- SCHEMA privileges
- SYSADMIn privileges
- SYStem privileges
- SYStem PROfile privileges
- TABLE privileges
- USEr PRIVileges
- USEr PROfile privileges

Parameters

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of the group identifier and its status (active or logically deleted).

- **DETAils**
  Specifies display of the status of the group and a description of the group, if one is included in the definition.
- **HISTORY**
  Specifies display of the following:
  
  - The date and time the group definition was created, and the ID of the user who created the definition.
  
  - The date and time the group definition was last updated, and the ID of the user who updated the definition.

- **USERS**
  Specifies display of all users who are members of the group.

- **all PRIVILEGES**
  Specifies display of all privileges held by the group.

- **ACCESS MODE privileges**
  Specifies display of privileges on access modules, if any are held by the group.

- **ACTIVITY privileges**
  Specifies display of privileges on activities, if any are held by the group.

- **AREA privileges**
  Specifies display of privileges on areas, if any are held by the group.

- **CATEGORY privileges**
  Specifies display of privileges on categories, if any are held by the group.

- **DB privileges**
  Specifies display of privileges on databases, if any are held by the group.

- **DBADMIN privileges**
  Specifies display of DBADMIN privilege, if it is held by the group.

- **DBTable privileges**
  Specifies display of privileges on database name tables, if any are held by the group.

- **DCADMIN privileges**
  Specifies display of DCADMIN privilege, if it is held by the group.

- **DMCI privileges**
  Specifies display of privileges on DMCLs, if any are held by the group.

- **GROUP privileges**
  Specifies display of privileges on groups, if any are held by the group.

- **NONSQL SCHEMA privileges**
  Specifies display of privileges on non-SQL-defined schemas, if any are held by the group.

- **SCHEMA privileges**
  Specifies display of privileges on SQL-defined schemas, if any are held by the group.
- **SYSADMIN privileges**
  Specifies display of SYSADMIN privilege, if it is held by the group.

- **SYstem privileges**
  Specifies display of privileges on systems, if any are held by the group.

- **SYStem PROfile privileges**
  Specifies display of privileges on system profiles, if any are held by the group.

- **TABle privileges**
  Specifies display of privileges on tables, if any are held by the group.

- **USER PRивileges**
  Specifies display of privileges on users, if any are held by the group.
  Note that at least the first three characters of the keyword PRIVILEGES must be specified to display users on which the group holds privileges. This removes any potential ambiguity with specification of the preceding USERS display option.

- **USER PROfile privileges**
  Specifies display of privileges on user profiles, if any are held by the group.

**Usage**

*Privileges Assigned in More Than One Dictionary*

It may be necessary to connect to more than one dictionary to see all privileges assigned to a group. For example, if you issue the DISPLAY statement while connected to the system dictionary, the output will not include the SQL table privileges assigned in an application dictionary.

**DISPLAY USER**

**Contents**

- Authorization (see page 287)
- Syntax (see page 288)
- Parameters (see page 288)
- Usage (see page 290)

Displays information on a user.

**Authorization**

To display privileges granted to a user you must hold SYSADMIN privilege.
Syntax

```
►►─┬─ DISPLAY ─┬─ USER user-identifier
     └─ PUNCH ──┘

►─┬───────────────────────────────────────────┬──────────────────────────────►◄
│ │        ┌───────────────────┐ │        └─▼─┬─ WITH ──────┬─▼─
└─▼─┬─ ALSO WITH ──┘           └─▼─            display-option
├─ ALSo WITH ─┤
└─ WITHOUT ──┘

►─┬─────────────────────┬────────────────────────────────────────────────────►◄
└─ AS ─┬─ SYNTAX ────┬─┘
      └─ COMMENTS ─┘
```

Expansion of Display-Option

```
►►─┬─ ALL ────────────┬───────────────────────────────────────────►◄
├─ NONe ──────────────┤
├─ DETAILS ────────────┤
├─ HISTORY ────────────┤
├─ GROUPS ──────────────┤
├─ all PRIVileges ──────┤
├─ ACCESS MODULE privileges ────┤
├─ ACTIVITY privileges ────┤
├─ AREA privileges ────────┤
├─ CATEGORY privileges ──────┤
├─ DB privileges ────────┤
├─ DBADMIn privileges ────┤
├─ DBTable privileges ────┤
├─ DCADMIn privileges ────┤
├─ DMC1 privileges ────┤
├─ GROUP PRivilages ────┤
├─ NONSQL SCHEMA privileges ────┤
├─ SCHEMA privileges ────┤
├─ SYSADMIn privileges ────┤
├─ SYSTEM privileges ────┤
├─ SYSTem PROfile privileges ────┤
├─ TABLE privileges ────┤
├─ USER privileges ────┤
├─ USER PROFILE privileges ────┤
└─ USER PROFILE privileges ────┘
```

Parameters

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of only the user identifier and its status (active or logically deleted).

- **DETAILS**
  Specifies display of the following information on the user:
  - The status of the user identifier.
  - The user's name.
  - Whether a password has been assigned to the user.
  - The user profile, if any, associated with the user.
  - The description, if any, of the user.
• **HISTORY**
  Specifies display of the following:
  
  - The date and time the user definition was created, and the ID of the user who created the definition.
  
  - The date and time the user definition was last updated and the ID of the user who updated the definition.

• **GROUPS**
  Specifies display of all groups of which the user is a member.

• **all PRIVILEGES**
  Specifies display of all privileges held by the user.

• **ACCESS MODULE privileges**
  Specifies display of privileges on access modules, if any are held by the user.

• **ACTIVITY privileges**
  Specifies display of privileges on activities, if any are held by the user.

• **AREA privileges**
  Specifies display of privileges on areas, if any are held by the user.

• **CATEGORY privileges**
  Specifies display of privileges on categories, if any are held by the user.

• **DB privileges**
  Specifies display of privileges on databases, if any are held by the user.

• **DBADMIN privileges**
  Specifies display of DBADMIN privilege, if it is held by the user.

• **DBTable privileges**
  Specifies display of privileges on database name tables, if any are held by the user.

• **DCADMIN privileges**
  Specifies display of DCADMIN privilege, if it is held by the user.

• **DMCL privileges**
  Specifies display of privileges on DMCLs, if any are held by the user.

• **GROUP PRIVILEGES**
  Specifies display of privileges on groups, if any are held by the user.
  Note that at least the first three characters of the keyword PRIVILEGES must be specified to display groups on which the user holds privileges. This removes any potential ambiguity with specification of the preceding GROUPS display option.

• **NONSQL SCHEMA privileges**
  Specifies display of privileges on non-SQL-defined schemas, if any are held by the user.

• **SCHEMA privileges**
  Specifies display of privileges on SQL-defined schemas, if any are held by the user.
- **SYSADMIN privileges**
  Specifies display of SYSADMIN privilege, if it is held by the user.

- **SYSTEM privileges**
  Specifies display of privileges on systems (including SIGNON), if any are held by the user.

- **SYSTEM PROFILE privileges**
  Specifies display of privileges on system profiles, if any are held by the user.

- **TABLE privileges**
  Specifies display of privileges on tables, if any are held by the user.

- **USER privileges**
  Specifies display of privileges on users, if any are held by the user.

- **USER PROFILE privileges**
  Specifies display of privileges on user profiles, if any are held by the user.

**Usage**

*Privileges Assigned in More Than One Dictionary*

It may be necessary to connect to more than one dictionary to see all privileges assigned to a user. For example, if you issue the DISPLAY statement while connected to the system dictionary, the output will not include the SQL table privileges assigned in an application dictionary.

**DISPLAY USER PROFILE**

**Contents**

- Authorization (see page 290)
- Syntax (see page 290)
- Parameters (see page 291)

Displays information on a user profile.

**Authorization**

To display user profile information, you must hold SYSADMIN privilege.

**Syntax**

```
DISPLAY USER PROFILE profile-name
```
Parameters

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of the profile name.

- **DETAILS**
  Specifies display of all attributes defined in the profile.

- **HISTORY**
  Specifies display of the following:
  - The date and time the user profile definition was created, and the ID of the user who created the definition.
  - The date and time the user profile definition was last updated, and the ID of the user who updated the definition.

**DISPLAY DCADMIN PRIVILEGES**

Contents

- Authorization (see page 291)
- Syntax (see page 292)
- Parameters (see page 292)
- Usage (see page 292)

Displays users and groups who hold DCADMIN privilege for the system.

**Authorization**

To display DCADMIN privilege, you must hold DCADMIN privilege or SYSADMIN privilege.
Syntax

```
DISPLAY PUNCH DCADMx privileges

WITH display-option

VERB GRANT REVOKE

AS SYNTAX COMMENTS

Expansion of Display-Option

ALL NONE HISTORY

Parameters

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of the users or groups to whom the privilege was granted.

- **HISTORY**
  Specifies display of the following:
  - The date and time the privilege was granted to the user or group, and the ID of the user granted the privilege.
  - The date and time the privilege was last updated, and the ID of the user who updated the privilege.

- **VERB**
  Specifies the verb used in the display output.
  If VERB is not specified, the privilege is displayed in the form of GRANT statements.
  - **GRANT**
    Specifies that the privilege is displayed in the form of GRANT statements.
  - **REVOKE**
    Specifies substitution of "REVOKE" for "GRANT" in the display output.

Usage

*Specifying WITH VERB REVOKE*
By specifying WITH VERB REVOKE and AS SYNTAX, you display output that you can then submit to revoke privilege from those who hold the privilege. This is useful when you need to revoke a privilege from most or all users who hold it.

**DISPLAY PRIVILEGES on a System Resource**

### Authorization

To display privileges on a system resource, you must hold one of these privileges:

- DCADMIN privilege
- SYSADMIN privilege

You can display privileges on a system or system profile if you hold DISPLAY privilege WITH GRANT OPTION on the resource.

### Syntax

```
Display PRileges on resource

Activity application-name.activity-name
Category category-name
System system-identifier
System Profile profile-name

With
    Also With
    Without

VERB Grant
    REVerse

AS Syntax
    COMments

Expansion of Display-Option

    ALL
    NONe
    HIsitory
```
Parameters

- **ACTivity**
  Specifies that the type of resource is activity.

- **application-name.activity-name**
  Identifies the application in which the activity is defined and the name of the activity.

- **CATegory**
  Specifies that the type of resource is category.

- **category-name**
  Identifies the category.

- **SYStem**
  Specifies that the type of resource is system.

- **system-identifier**
  Identifies the system.

- **SYStem PROfile**
  Specifies that the type of resource is system profile.

- **profile-name**
  Identifies the system profile.

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of only the privileges and the authorization identifier to which they have been granted.

- **HISTORY**
  Specifies display of the following:
  - The date and time privilege on the resource was granted to the user or group, and the ID of the user who granted the privilege.
  - The date and time privilege on the resource was last updated, and the ID of the user who updated the privilege.

- **VERB**
  Specifies the verb to be used in the display output.
  If VERB is not specified, the privilege is displayed in the form of GRANT statements.

  - **GRAnt**
    Specifies that privileges are to be displayed in the form of GRANT statements.

  - **REVoKe**
    Specifies substitution of "REVOKE" for "GRANT" in the display output.
Usage

Specifying WITH VERB REVOKE

By specifying WITH VERB REVOKE and AS SYNTAX, you display output that you can then submit to revoke privilege from those who hold the privilege. This is useful when you need to revoke a privilege from most or all users who hold it.

DISPLAY RESOURCE (System)

Contents

- Authorization (see page 295)
- Syntax (see page 295)
- Parameters (see page 296)

Displays information on a system resource.

Authorization

To display a system resource, you must hold one of these privileges:

- DISPLAY privilege WITH GRANT OPTION on the resource
- DCADMIN privilege
- SYSADMIN privilege

Syntax

```
DISPLAY RESOURCE
  │ PUNCH
  │ ACTIVITY application-name.activity-name
  │ CATEGORY category-name
  │ SYSTEM system-identifier
  │ WITH display-option
  │ ALSO WITHOUT
  │ AS SYNTAX COMMENTS
```

Expansion of Display-Option
Parameters

- **ACTivity**
  Specifies that the type of resource is activity.

- **application-name.activity-name**
  Identifies the application in which the activity is defined and the activity name.

- **CATegory**
  Specifies that the type of resource is category.

- **category-name**
  Identifies the category.

- **SYStem**
  Specifies that the type of resource is system.

- **system-identifier**
  Identifies the system.

- **SYStem PRofile**
  Specifies that the type of resource is system profile.

- **profile-name**
  Identifies the system profile.

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of only the resource type such as:
  - For category, the resource name
  - For activity, the activity name and number
  - For system, the system identifier

- **DETAils**
  For a category, specifies display of the resources in the category.
  For a system profile, specifies display of the attributes defined in the profile.

⚠️ **Note:** The DETAILS option is meaningful only for the CATEGORY resource type.
DISPLAY PRIVILEGES on a Database Resource

Contents

- Authorization (see page 297)
- Syntax (see page 297)
- Parameters (see page 298)
- Usage (see page 299)

Displays privileges granted on a database resource and the users and groups who hold the privileges.

Authorization

To display privileges on a database resource, you must hold DBADMIN privilege on the current dictionary, SYSADMIN privilege, or the privilege listed in this table:

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Privilege to display privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Module</td>
<td>DISPLAY (WITH GRANT OPTION) on the access module</td>
</tr>
<tr>
<td>Area</td>
<td>DISPLAY (WITH GRANT OPTION) on the segment containing the area</td>
</tr>
<tr>
<td>Database</td>
<td>DISPLAY (WITH GRANT OPTION) on the database identified as a DB resource</td>
</tr>
<tr>
<td>Database name table</td>
<td>DISPLAY (WITH GRANT OPTION) on the database name表</td>
</tr>
<tr>
<td>DMCL</td>
<td>DISPLAY (WITH GRANT OPTION) on the DMCL</td>
</tr>
<tr>
<td>Non-SQL-defined schema</td>
<td>DISPLAY (WITH GRANT OPTION) on the non-SQL-defined schema</td>
</tr>
<tr>
<td>SQL-defined schema</td>
<td>DISPLAY (WITH GRANT OPTION) on the schema</td>
</tr>
<tr>
<td>Table</td>
<td>DISPLAY (WITH GRANT OPTION) on the table</td>
</tr>
</tbody>
</table>

Syntax

```plaintext
$DISPLAY PRIVileges on resource
```

```
ACCESS MODULE schema-name.access-module-name
AREA segment-name.area-name
DB database-name
DBTable dtable-name
DMCL dmcl-name
NONSQL SCHEMA Vnnnn.nonsql-schema-name
SCHEMA schema-name
```
Parameters

- **ACCess MODule**
  Specifies that the type of resource is access module.

- **schema-name.access-module-name**
  Identifies the access module and its schema-name qualifier.

- **AREa**
  Specifies that the type of resource is area.

- **segment-name.area-name**
  Identifies the segment containing the area and the area name.

- **DB**
  Specifies that the type of resource is database.

- **database-name**
  Identifies a segment or a name in the database name table.

- **DBTable**
  Specifies that the type of resource is database name table.

- **dbtable-name**
  Identifies the database name table.

- **DMCL**
  Specifies that the type of resource is DMCL.

- **dmcl-name**
  Identifies the DMCL.

- **NONSQL SCHema**
  Specifies that the type of resource is non-SQL-defined schema.
- **Vnnen.nonsql-schema-name**
  Identifies the non-SQL-defined schema and its version (nnnn).

- **SCHema**
  Specifies that the type of resource is an SQL-defined schema.

- **schema-name**
  Identifies the schema.

- **TABle**
  Specifies that the type of resource is table.

- **schema-name.table-name**
  Identifies the table and its schema-name qualifier.

- **ALL**
  Specifies all display options.

- **NONE**
  Specifies display of only the privileges and the authorization identifier to which they have been granted.

- **HISTORY**
  Specifies display of the following:
  - The date and time privilege on the resource was granted to the user or group, and the ID of the user who granted the privilege.
  - The date and time privilege on the resource was last updated, and the ID of the user who updated the privilege.

- **VERB**
  Specifies the verb to be used in the display output.
  If VERB is not specified, the privilege is displayed in the form of GRANT statements.

  - **GRANT**
    Specifies that privileges are displayed in the form of GRANT statements.

  - **REVOKE**
    Specifies substitution of "REVOKE" for "GRANT" in the display output.

Usage

*Specifying WITH VERB REVOKE*

By specifying WITH VERB REVOKE and AS SYNTAX, you display output that you can then submit to revoke privilege from those who hold the privilege. This is useful when you need to revoke a privilege from most or all users who hold it.
DISPLAY RESOURCE (Database)

Contents

- Authorization (see page 300)
- Syntax (see page 300)
- Parameters (see page 301)

Displays information on a database resource.

Authorization

To display a database resource, you must hold DBADMIN privilege on the current dictionary, SYSADMIN privilege, or the privilege listed in this table:

<table>
<thead>
<tr>
<th>Resource type</th>
<th>Privilege to display privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Module</td>
<td>DISPLAY on the access module</td>
</tr>
<tr>
<td>Area</td>
<td>DBADMIN on the segment containing the area</td>
</tr>
<tr>
<td>Database</td>
<td>DBADMIN on the database identified as a DB resource</td>
</tr>
<tr>
<td>Database name table</td>
<td>DISPLAY on the database name table</td>
</tr>
<tr>
<td>DMCL</td>
<td>DISPLAY on the DMCL</td>
</tr>
<tr>
<td>Non-SQL-defined schema</td>
<td>DISPLAY on the non-SQL-defined schema</td>
</tr>
<tr>
<td>SQL-defined schema</td>
<td>DISPLAY on the schema</td>
</tr>
<tr>
<td>Table</td>
<td>DISPLAY on the table</td>
</tr>
</tbody>
</table>

Syntax

```
DISPLAY RESOURCE
PUNCH

ACCESS MODULE schema-name.access-module-name
DB database-name
DBTABLE dbtable-name
DMCL dmcl-name
NONSQL SCHEMA Vnnnn.nonsql-schema-name
SCHEMA schema-name
TABLE schema-name.table-name

WITH ALL
ALSO WITHOUT
WITHOUT NONE

AS SYNTAX COMMENTS
```
Parameters

- **ACCess MODule**
  Specifies that the type of resource is access module.

- **schema-name.access-module-name**
  Identifies the access module and its schema-name qualifier.

- **AREa**
  Specifies that the type of resource is area.

- **segment-name.area-name**
  Identifies the segment containing the area and the area name.

- **DB**
  Specifies that the type of resource is database.

- **database-name**
  Identifies a segment or a name in the database name table.

- **DBTable**
  Specifies that the type of resource is database name table.

- **dbtable-name**
  Identifies the database name table.

- **DMCL**
  Specifies that the type of resource is DMCL.

- **dmcl-name**
  Identifies the DMCL.

- **NONSQL SCHema**
  Specifies that the type of resource is non-SQL-defined schema.

- **Vnnnn.nonsql-schema-name**
  Identifies the non-SQL-defined schema and its version (nnnn).

- **SCHema**
  Specifies that the type of resource is schema.

- **schema-name**
  Identifies the schema.

- **TABle**
  Specifies that the type of resource is table.

- **schema-name.table-name**
  Identifies the table and its schema-name qualifier.

- **ALL**
  Specifies all display options.
### NONe
Specifies display of the resource type and name only.

### H1story
Specifies display of the following:
- The date and time the resource was created, and the ID of the user who created it.
- The date and time the resource was last updated, and the ID of the user who updated it.
- For a table or schema, the authorization identifier of the owner.

### DISPLAY/PUNCH ALL Syntax for Security Definitions

In addition to using DISPLAY and PUNCH syntax for specific resource definitions in a CA IDMS security database, you can issue either a DISPLAY ALL or PUNCH ALL statement for an entity type to display or punch all occurrences defined within that entity type. For example, you can issue a DISPLAY ALL RESOURCE AREAS to see the security definitions for all areas secured in a security database. You can also select occurrences of an entity type to display or punch by specifying the following:

- Conditional expressions
- First occurrence of the entity type
- Last occurrence of the entity type
- A specific number of occurrences

### Choosing Which Entity Occurrences to Display (see page 302)
### Issue Statements from CA IDMS Command Facility (see page 303)
### Syntax (see page 303)
### Parameters (see page 304)
### Usage (see page 306)

### Choosing Which Entity Occurrences to Display

The DISPLAY and PUNCH ALL syntax supports a variety of selection criteria to select occurrences to DISPLAY or PUNCH. You can use a conditional expression with Boolean criteria to select occurrences, including a mask comparison. The mask comparison supports the use of different keywords for each entity type. A table of keywords by entity type is presented in Usage (see page 306).
Issue Statements from CA IDMS Command Facility

You can issue DISPLAY/PUNCH statements from either the Online (OCF) or Batch (BCF) Command Facility.

Syntax

```
DISPLAY/PUNCH entity-type entity-count
WHERE conditional-expression
VERB DISPLAY/PUNCH CREATE ALTER DROP
AS COMMENTS SYNTAX
```

Expansion of Conditional-Expression

```
mask-comparison value-comparison
( mask-comparison value-comparison )
AND mask-comparison value-comparison
( mask-comparison value-comparison )
NOT mask-comparison value-comparison
```

Expansion of Mask-Comparison

```
entity-option-keyword
CONTAINS 'mask-value'
MATCHES 'mask-value'
```

Expansion of Value-Comparison

```
'character-string-literal'
numeric-literal
entity-option-keyword
IS NOT
EQ NOT
GT NOT
LT
```

Parameters

- **ALL**
  Lists all occurrences of the requested entity type that the current user is authorized to display. **Online users**: With a large number of entity occurrences, ALL may have a slow response time.

- **FIRst**
  Lists the first occurrence of the named entity type.

- **LASt**
  Lists the last occurrence of the named entity type.

- **entity-count**
  Specifies the number of occurrences of the named entity type to list. 1 is the default.

- **entity-type**
  Identifies the entity type that is the object of the DISPLAY/PUNCH ALL request. Valid values appear in the table in *Usage (see page 306)*.

- **VERB DISplay/PUNch/CREate/ALTer/DROp**
  Specifies the verb that is to accompany DISPLAY/PUNCH output. DISPLAY is the default.

- **AS SYNTAX**
  Specifies that the text output by the DISPLAY/PUNCH verb is to appear as syntax. In an online session, text displayed as syntax can be edited and resubmitted to the command facility. If the PUNCH command is issued in batch mode, the batch command facility directs the output to the SYSPCH file, where it can be edited and subsequently resubmitted.

- **AS COMMENTS**
  Specifies that the text output by the DISPLAY/PUNCH verb be formatted as comments; comments are preceded by "+" and are ignored by the command facility.

- **WHERE conditional-expression**
  Specifies criteria to be used in selecting occurrences of the requested entity type. The outcome of a test for the condition determines which occurrences of the named entity type are displayed.

- **mask-comparison**
  Compares an entity type operand with a mask value.

  - **entity-option-keyword**
    Identifies the left operand as a syntax option associated with the named entity type. The table in *Usage (see page 306)* lists valid options for each entity type.

  - **CONTAINS**
    Searches the left operand for an occurrence of the right operand. The length of the right operand must be less than or equal to the length of the left operand. If the right operand is not contained entirely in the left operand, the outcome of the condition is false.
MATCHES

Compares the left operand with the right operand one character at a time, beginning with the leftmost character in each operand. When a character in the left operand does not match a character in the right operand, the outcome of the condition is false.

'mask-value'

Identifies the right operand as a character string; the specified value must be enclosed in quotation marks. Mask-value can contain the following special characters:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>@</td>
<td>Matches any alphabetic character in entity-option-keyword.</td>
</tr>
<tr>
<td>#</td>
<td>Matches any numeric character in entity-option-keyword.</td>
</tr>
<tr>
<td>*</td>
<td>Matches any character in entity-option-keyword.</td>
</tr>
</tbody>
</table>

value-comparison

Compares values contained in the left and right operands based on the specified comparison operator.

'character-string-literal'

Identifies a character string enclosed in quotes.

numeric-literal

Identifies a numeric value.

data

Identifies a syntax option associated with the named entity type; valid options for each entity type are listed in the table presented in Usage (see page 306).

IS

Specifies that the left operand must equal the right operand for the condition to be true.

NE

Specifies that the left operand must not equal the right operand for the condition to be true.

EQ/= Specifies that the left operand must equal the right operand for the condition to be true.

GT/> Specifies that the left operand must be greater than the right operand for the condition to be true.

LT/< Specifies that the left operand must be less than the right operand for the condition to be true.

GE

Specifies that the left operand must be greater than or equal to the right operand for the condition to be true.

LE

Specifies that the left operand must be less than or equal to the right operand for the condition to be true.
- **NOT**
  Specifies that the opposite of the condition fulfills the test requirements. If NOT is specified, the condition must be enclosed in parentheses.

- **AND**
  Indicates the expression is true only if the outcome of both test conditions is true.

- **OR**
  Indicates the expression is true if the outcome of either one or both test conditions is true.

## Usage

**Output Contains Only Enough Information to Display/Punch Entity**

Output produced by DISPLAY or PUNCH ALL consists only of the information necessary to execute a DISPLAY/PUNCH request for each entity occurrence. For example, Resource DMCL occurrences are displayed with their name, and AREA occurrences with their fully qualified name (that is, segmentname.arenaname). In an online session, the user can execute the displayed statements by pressing [Enter]. This two-step process allows the user to scan the names of entity occurrences related to the database in which the statement is issued.

**Valid Entity Types and Option Keywords for Conditional Expressions**

The following table lists valid entity types and keywords that you can specify as `entity-type` and `entity-option-keyword` in the DISPLAY ALL and PUNCH ALL syntax.

<table>
<thead>
<tr>
<th>Entity type</th>
<th>Entity-option keyword</th>
<th>Selects based on</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Security components</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>Unqualified Name (1)</td>
<td></td>
</tr>
<tr>
<td>FULL NAME</td>
<td>Qualified Name (1)</td>
<td></td>
</tr>
<tr>
<td>RESOURCENAME</td>
<td>Unqualified Name</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Resources only) (1)</td>
<td></td>
</tr>
<tr>
<td>CREATED BY</td>
<td>User who created occurrence</td>
<td></td>
</tr>
<tr>
<td>PREPARED BY</td>
<td>User who created occurrence</td>
<td></td>
</tr>
<tr>
<td>LAST UPDATED BY</td>
<td>User who last updated occurrence</td>
<td></td>
</tr>
<tr>
<td>REVISED BY</td>
<td>User who last updated occurrence</td>
<td></td>
</tr>
<tr>
<td>DATE last UPDATED</td>
<td>Date (MM/DD/YY) occurrence last updated</td>
<td></td>
</tr>
<tr>
<td>MONTH last UPDATED</td>
<td>Month occurrence last updated</td>
<td></td>
</tr>
<tr>
<td>DAY last UPDATED</td>
<td>Day occurrence last updated</td>
<td></td>
</tr>
<tr>
<td>YEAR last UPDATED</td>
<td>Year occurrence last updated</td>
<td></td>
</tr>
<tr>
<td>DATE CREATED</td>
<td>Date (MM/DD/YY) occurrences created</td>
<td></td>
</tr>
<tr>
<td>MONTH CREATED</td>
<td>Month occurrence created</td>
<td></td>
</tr>
<tr>
<td>DAY CREATED</td>
<td>Day occurrence created</td>
<td></td>
</tr>
<tr>
<td>YEAR CREATED</td>
<td>Year occurrence created</td>
<td></td>
</tr>
<tr>
<td>Entity type</td>
<td>Entity-option keyword</td>
<td>Selects based on</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Global Security components</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GROups</td>
<td>GROUP name</td>
<td>Name (ID) of Group</td>
</tr>
<tr>
<td></td>
<td>STATus</td>
<td>Status of GROUP (ACTIVE, INACTIVE, LOGICALLY DELETED)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GROUP name</td>
<td>Name (ID) of Group</td>
</tr>
<tr>
<td></td>
<td>STATus</td>
<td>Status of GROUP (ACTIVE, INACTIVE, LOGICALLY DELETED)</td>
</tr>
<tr>
<td></td>
<td>FULL NAME</td>
<td>Full Name of User</td>
</tr>
<tr>
<td></td>
<td>PROFILE</td>
<td>Profile assigned to User</td>
</tr>
<tr>
<td>USErs</td>
<td>USEr name</td>
<td>Name (ID) of User</td>
</tr>
<tr>
<td></td>
<td>STATus</td>
<td>Status of USER (ACTIVE, INACTIVE, LOGICALLY DELETED)</td>
</tr>
<tr>
<td></td>
<td>FULL NAME</td>
<td>Full Name of User</td>
</tr>
<tr>
<td></td>
<td>PROFILE</td>
<td>Profile assigned to User</td>
</tr>
<tr>
<td>USEr PROfiles</td>
<td>USEr PROfile name</td>
<td>Profile Name</td>
</tr>
<tr>
<td></td>
<td>PROfile name</td>
<td>Profile Name</td>
</tr>
<tr>
<td><strong>Physical Database Security components</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESource AREas</td>
<td>resource AREa NAME</td>
<td>Unqualified AREA name (1)</td>
</tr>
<tr>
<td></td>
<td>SEGment name</td>
<td>Area’s segment name</td>
</tr>
<tr>
<td>RESource DBs</td>
<td>resource DB NAME</td>
<td>Name of Database</td>
</tr>
<tr>
<td>RESource DBTables</td>
<td>resource DBTable NAME</td>
<td>Name of DBTable</td>
</tr>
<tr>
<td>RESource DMClS</td>
<td>resource DMCL NAME</td>
<td>Name of DMCl</td>
</tr>
<tr>
<td>RESource NONsql SCHEmas</td>
<td>resource NONSQL SCHEMA NAME</td>
<td>Name of NON SQL Schema</td>
</tr>
<tr>
<td><strong>SQL Security Components</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESource ACCess MODules or RESource AMS</td>
<td>resource ACCess MODule NAME</td>
<td>Unqualified Name of Access Module (1)</td>
</tr>
<tr>
<td></td>
<td>AM NAME</td>
<td>Unqualified Name of Access Module (1)</td>
</tr>
<tr>
<td></td>
<td>SCHEMA name</td>
<td>Schema Name of Access Module</td>
</tr>
<tr>
<td>RESource SCHEmas</td>
<td>resource SCHEMA NAME</td>
<td>Name of SQL Schema</td>
</tr>
<tr>
<td>RESource TABles</td>
<td>resource TABle NAME</td>
<td>Unqualified Name of Table (1)</td>
</tr>
<tr>
<td></td>
<td>SCHEMA NAME</td>
<td>Schema Name of Table</td>
</tr>
<tr>
<td><strong>System Security Components</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESource ACTivities</td>
<td>resource ACTivity name NUMBER</td>
<td>Name of Activity</td>
</tr>
<tr>
<td></td>
<td>NUMBER</td>
<td>Activity Number</td>
</tr>
<tr>
<td>RESource CATegories</td>
<td>resource CATegory NAME NUMBER</td>
<td>Name of Category</td>
</tr>
<tr>
<td></td>
<td>NUMBER</td>
<td>Category Number</td>
</tr>
<tr>
<td>RESource SYStems</td>
<td>resource SYStem NAME</td>
<td>Name of System</td>
</tr>
<tr>
<td>SYStem PROfiles</td>
<td>system PROFILE NAME</td>
<td>Profile Name</td>
</tr>
</tbody>
</table>

The following **Resource Category Components** can be selected using the specified entity-option keyword (in addition to those specified in the preceding Resource Categories).
### Entity type

<table>
<thead>
<tr>
<th>Entity type</th>
<th>Entity-option keyword</th>
<th>Selects based on</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESource CATegory ACCESS MODules or RESource CATegory AMS</td>
<td>ACCESS MODule name</td>
<td>Unqualified Access Module Name (1) Dictionary Name</td>
</tr>
<tr>
<td></td>
<td>DICTName</td>
<td>Dictionary Name</td>
</tr>
<tr>
<td></td>
<td>Dictionary name</td>
<td>SQL Schema Name</td>
</tr>
<tr>
<td></td>
<td>SCHema name</td>
<td></td>
</tr>
<tr>
<td>RESource category LOad MODules</td>
<td>LOad MODule name</td>
<td>Unqualified Load Module Name (1)</td>
</tr>
<tr>
<td></td>
<td>DICTName</td>
<td>Dictionary Name</td>
</tr>
<tr>
<td></td>
<td>Dictionary name</td>
<td>Version Number</td>
</tr>
<tr>
<td></td>
<td>Version</td>
<td>(in Vnnn format)</td>
</tr>
<tr>
<td>RESource category PROgrams</td>
<td>PROgram name</td>
<td>Unqualified Program name</td>
</tr>
<tr>
<td></td>
<td>File name</td>
<td>File Name (CDMSLIB)</td>
</tr>
<tr>
<td></td>
<td>Version</td>
<td>Version Number</td>
</tr>
<tr>
<td></td>
<td>(in Vnnn format)</td>
<td></td>
</tr>
<tr>
<td>RESource category QUEues</td>
<td>QUEue name</td>
<td>Name of Queue</td>
</tr>
<tr>
<td>RESource category RUNunits</td>
<td>RUNunit name</td>
<td>Unqualified Rununit name (1)</td>
</tr>
<tr>
<td></td>
<td>DATABASE NAMe</td>
<td>Database Name</td>
</tr>
<tr>
<td></td>
<td>DBName</td>
<td>Database Name</td>
</tr>
<tr>
<td></td>
<td>SUBschema name</td>
<td>Subschema Name</td>
</tr>
<tr>
<td></td>
<td>PROgram name</td>
<td>Program Name</td>
</tr>
<tr>
<td>RESource category TASKs</td>
<td>TASK name</td>
<td>Name of task</td>
</tr>
</tbody>
</table>

(1) Unqualified name selections are based on the primary name of the entity occurrence only. To select based on the fully qualified occurrence name, token FULL NAME must be specified. Security components with qualified names are specified in the following table.

#### Fully Qualified Names of Security Components

The fully qualified names of security components are listed in the following table.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Fully qualified name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS MODULE</td>
<td><code>schema-name.access-module-name</code></td>
</tr>
<tr>
<td>AREA</td>
<td><code>segment-name.area-name</code></td>
</tr>
<tr>
<td>TABLE</td>
<td><code>schema-name.table-name</code></td>
</tr>
<tr>
<td>CATEGORY ACCESS MODULE</td>
<td><code>dictname.schema-name.access-module-name</code></td>
</tr>
<tr>
<td>CATEGORY LOAD MODULE</td>
<td><code>dictname.Vnnnn.load-module-name</code></td>
</tr>
<tr>
<td>CATEGORY RUNUNIT</td>
<td><code>dbname.subschema-name.program-name</code></td>
</tr>
<tr>
<td>CATEGORY PROGRAM</td>
<td><code>CDMSLIB.program-name</code> or <code>Vnnnn.program-name</code></td>
</tr>
</tbody>
</table>

For all other security components, unqualified and qualified names are the same.
Date and Year 2000 support

You can use date selection criteria and year 2000 support in DISPLAY/PUNCH ALL statements to display security entities.

You implement date selection criteria in these WHERE clause options:

- DATE CREATED
- DATE LAST UPDATED

You can specify the date as a value-comparison string in the form 'MM/DD/YY' in the right side of the conditional expression. CA IDMS extracts it in CCYYMMDD form to accurately determine the relationship of dates. For example, this DISPLAY ALL statement:

```
DISPLAY ALL USERS WHERE DATE CREATED > '01/01/96';
```

establishes a search criteria to identify the USERS whose DATE CREATED values are greater than the specified string. The DISPLAY ALL process determines that the date '01/01/96' is greater than the date '12/31/95'.

Alternatively, you may specify the value-comparison string on either side of the conditional expression in the form 'CCYYMMDD' to achieve the same results.

You can also substitute day, month, or year for each of these WHERE clause options. For example, this DISPLAY ALL statement specifies a search condition that is based on month and year:

```
DISPLAY ALL RESOURCE AREAS
  WHERE MONTH CREATED = '01'
  AND YEAR CREATED > '95';
```

Default Order of Precedence Applied to Logical Operators

Conditional expressions can contain a single condition, or two or more conditions combined with the logical operators AND or OR. The logical operator NOT specifies the opposite of the condition. The command facility evaluates operators in a conditional expression one at a time, from left to right, in order of precedence. The default order of precedence is as follows:

- MATCHES or CONTAINS keywords
- EQ, NE, GT, LT, GE, LE operators
- NOT
- AND
- OR

If parentheses are used to override the default order of precedence, the command facility evaluates the expression within the innermost parentheses first.

The following examples show sample DISPLAY statements for security definitions.

```
DISPLAY ALL GROUPS WHERE STATUS IS 'ACTIVE'
```
Security Macro JCL

For more information, see the following topics:

- z/OS - JCL (see page 310)
- z/VSE - JCL (see page 312)
- CMS Commands 1 (see page 315)

z/OS - JCL

Contents

- CTABGEN (see page 310)
- GTABGEN (see page 311)
- SECRTT (see page 311)
- UTABGEN (see page 312)

CTABGEN

The #CTABGEN macro can be assembled using the following JCL.

```jcl
// EXEC HLASMCL
//ASM.SYSLIB DD DSN=yourHLQ.CAGJMAC,DISP=SHR
// DD DSN=zOS.maclib,DISP=SHR
//ASM.SYSIN DD *
#CTABGEN macro
END
/*
//LKED.SYSLMOD DD DSN=yourHLQ.custom.loadlib,DISP=SHR
//LKED.SYSIN DD *
ENTRY CTABEP1
MODE AMODE(31),RMODE(ANY)
NAME IDMSCTAB(R)
/*
```
- **yourHLQ.CAGJMAC**
  Data set name of the CA IDMS macro library.

- **zos.maclib**
  Data set name of the operating system macro library.

- **yourHLQ.custom.loadlib**
  Data set name of the CA IDMS custom load library.

**GTABGEN**

The #GTABGEN macro can be assembled using the following JCL.

```jcl
// EXEC HLASMCL
//ASM.SYSLIB DD DSN=yourHLQ.CAGJMAC,DISP=SHR
//ASM.SYSIN DD *
// #GTABGEN macro
END
/*
//LKED.SYSLMOD DD DSN=yourHLQ.custom.loadlib,DISP=SHR
//LKED.SYSIN DD *
ENTRY GTABEP1
MODE AMODE(31),RMODE(ANY)
NAME IDMSGTAB(R)
/*
/*
```

- **yourHLQ.CAGJMAC**
  Data set name of the CA IDMS macro library.

- **zos.maclib**
  Data set name of the operating system macro library.

- **yourHLQ.custom.loadlib**
  Data set name of the CA IDMS custom load library.

**SECRTT**

```jcl
// EXEC HLASMCL
//ASM.SYSLIB DD DSN=yourHLQ.CAGJMAC,DISP=SHR
//ASM.SYSIN DD *
// #SECRTT TYPE=INITIAL,SVCNUM=svcnumber
// #SECRTT macros
// #SECRTT TYPE=FINAL
END
/*
//LKED.SYSLMOD DD DSN=yourHLQ.custom.loadlib,DISP=SHR
//LKED.SYSIN DD *
ENTRY SRTTEP1
MODE AMODE(31),RMODE(ANY)
NAME RHDCSRTT(R)
/*
/*
```
UTABGEN

The #UTABGEN macro can be assembled using the following JCL.

```jcl
// EXEC HLASMCL
//ASM.SYSLIB DD DSN=yourHLQ.CAGJMAC,DISP=SHR
// DD DSN=zOS.maclib,DISP=SHR
//ASM.SYSIN DD *
  #UTABGEN macro
  END
/*
//LKED.SYSLMOD DD DSN=yourHLQ.custom.loadlib,DISP=SHR
//LKED.SYSIN DD *
ENTRY UTABEP1
  MODE AMODE(31),RMODE(ANY)
  NAME IDMSUTAB(R)
/*
//*/
```

- **yourHLQ.CAGJMAC**
  Data set name of the CA IDMS macro library
- **zOS.maclib**
  Data set name of the operating system macro library
- **yourHLQ.custom.loadlib**
  Data set name of the CA IDMS custom load library

z/VSE - JCL

Contents

- **CTABGEN** (see page 313)
- **GTABGEN** (see page 313)
- **SECRRT** (see page 314)
- **UTABGEN** (see page 314)

Using MSHP

Any modifications to CA IDMS load libraries in z/VSE should be applied by MSHP.
Note: For instructions on how to assemble and link edit a module using MSHP, see the Installing section -- z/VSE.

CTABGEN

// DLBL userlib// EXTENT ,nnnnnn// LIBDEF SOURCE,SEARCH=(userlib.idmslib)
// OPTION CATAL
// PHASE IDMSCTAB,*
// EXEC ASMA90
#CTABGEN macro
END
/*
ENTRY CTABEP1
// DLBL userlib
// EXTENT ,nnnnnn
// LIBDEF PHASE,TO=(userlib.idmslib)
// EXEC LNKEDT

- userlib
  Filename of the user library.

- nnnnnn
  Volume serial number of the library.

- userlib.idmslib
  File identifier of the CA IDMS sublibrary.

GTABGEN

// DLBL userlib// EXTENT ,nnnnnn// LIBDEF SOURCE,SEARCH=(userlib.idmslib)
// OPTION CATAL
// PHASE IDMSGTAB,*
// EXEC ASMA90
#GTABGEN macro
END
/*
ENTRY GTABEP1
// DLBL userlib
// EXTENT ,nnnnnn
// LIBDEF PHASE,TO=(userlib.idmslib)
// EXEC LNKEDT

- userlib
  Filename of the user library.

- nnnnnn
  Volume serial number of the library.

- userlib.idmslib
  File identifier of the CA IDMS sublibrary.
### SECRRTT

```plaintext
// DLBL  userlib//  EXTENT ,nnnnnn//  LIBDEF  SOURCE,SEARCH=(userlib.idmslib)
// OPTION  CATAL
// PHASE  RHDCSRTT,*
// EXEC  ASMA90
#SECRRTT  TYPE=INITIAL,SVCNUM=svcnumber
#SECRRTT  macro
...
#SECRRTT  TYPE=FINAL
END
/*
ENTRY  SRTTEP1
// DLBL  userlib
// EXTENT ,nnnnnn//  LIBDEF  PHASE,TO=(userlib.idmslib)
// EXEC  LNKEDT
```

- **userlib**
  - Filename of the user library.

- **nnnnnn**
  - Volume serial number of the library.

- **userlib.idmslib**
  - File identifier of the CA IDMS sublibrary.

### UTABGEN

```plaintext
// DLBL  userlib//  EXTENT ,nnnnnn//  LIBDEF  SOURCE,SEARCH=(userlib.idmslib)
// OPTION  CATAL
// PHASE  IDMSUTAB,*
// EXEC  ASMA90
#UTABGEN  macro
END
/*
ENTRY  UTABEP1
// DLBL  userlib//  EXTENT ,nnnnnn//  LIBDEF  PHASE,CATALOG=userlib.idmslib//  EXEC
```

- **userlib**
  - Filename of the user library.

- **nnnnnn**
  - Volume serial number of the library.

- **userlib.idmslib**
  - File identifier of the CA IDMS sublibrary.

- **Sample IDMSUTAB**
  - A sample IDMSUTAB.A module id provided with zero security. You may modify and assemble this source, or code your own.
CMS Commands 1

Contents

- CTABGEN (see page 315)
- GTABGEN (see page 315)
- SECRTT (see page 316)
- UTABGEN (see page 316)

CTABGEN

GLOBAL MACLIB idmslib OSMACRO
FILEDEF SYSLIN DISK IDMSUTAB TEXT A
ASMAHL idmsctab (NODECK OBJECT PRINT)
CAE5LNKB IDMSCTAB SYSLIN userload RENT

- idmslib
  Filename of the CA IDMS MACLIB library.

- idmsctab
  Filename of the file containing the #CTABGEN macro source. Filetype ASSEMBLE.

- userload
  Filename of the output LOADLIB where IDMSCTAB will be placed.

IDMSCTAB SYSLIN is provided with the install.

GTABGEN

GLOBAL MACLIB idmslib OSMACRO
FILEDEF SYSLIN DISK DBGSTAB TEXT A
ASMAHL idmsgtab (NODECK OBJECT PRINT)
CAE5LNKB IDMSGTAB SYSLIN userload RENT

- idmslib
  Filename of the CA IDMS MACLIB library

- idmsgtab
  Filename of the file containing the #GTABGEN macro source. Filetype ASSEMBLE.

- userload
  Filename of the output LOADLIB where IDMSGTAB will be placed.

IDMSCTAB SYSLIN is provided with the install.
SECRTT

GLOBAL MACLIB idmslib OSMACRO
FILEDEF SYSLIN DISK RHDCSRTT TEXT A
ASMAHL rhdcslt (NODECK OBJECT PRINT

CAE5LNKB RHDCSRTT SYSLIN userload RENT

- idmslib
  Filename of the CA IDMS MACLIB library

- rhdcslt
  Filename of the file containing the #SECRTT macro source. Filetype ASSEMBLE

- userload
  Filename of the output LOADLIB where the relinked RHDCSRTT will be placed

RHDCSRTT SYSLIN is provided with the install.

UTABGEN

GLOBAL MACLIB idmslib OSMACRO
FILEDEF SYSLIN DISK IDMSUTAB TEXT A
ASMAHL idmsutab (NODECK OBJECT PRINT

CAE5LNKB IDMSUTAB SYSLIN userload RENT

- idmslib
  Filename of the CA IDMS MACLIB library

- idmsutab
  Filename of the file containing the #UTABGEN macro source. Filetype ASSEMBLE

- userload
  Filename of the output LOADLIB library where IDMSUTAB will be placed

- Sample IDMSUTAB

A sample IDMSUTAB ASSEMBLE module is provided with zero security. You may modify and assemble this source, or code your own.

IDMSUTAB SYSLIN is provided with the install.

Security Database Information and DSECTs

You can access security information in the following ways:

- Issuing DISPLAY statements through the CA IDMS Command Facility
- Issuing SQL DML statements through the CA IDMS Command Facility
Security DISPLAY statements

What You Can Do

You can submit to the CA IDMS Command Facility DISPLAY or PUNCH statements to display this security information:

- About global resources:
  - Holders of SYSADMIN privilege
  - User information, including definitions, groups, and privileges
  - Group information, including definitions, users, and privileges
  - User profile information, including attributes

- About system resources:
  - Holders of DCADMIN privilege
  - Resource definitions
  - Privileges granted on resources

- About database resources:
  - Resource definitions
  - Privileges granted on resources

Documentation of DISPLAY Statements

⚠️ For documentation of DISPLAY statement syntax, see Syntax for Security Display Statements (see page 279).

DML Access

Navigational DML

For navigational DML access to a security database, you specify the appropriate security subschema and issue DML statements to navigate the security records that are described later in this section.
You can issue navigational DML statements through CA OLQ, CA Culprit, or a user-written program to access the user catalog through subschema IDMSSECU and other security definitions in the system dictionary through subschema IDMSSECS. You cannot access SQL-defined database resources through navigational DML.

SQL DML

For SQL access to information on global and system resources, you define SQL schemas for the non-SQL-defined security schemas (IDMSSECU and IDMSSECS) and issue SQL statements that specify records in the security databases as tables.

Security information on SQL-defined database resources resides in the SQL-defined tables in the SYSTEM schema that are described later in this section.

For more information, see the following topics:
- About Security Information (see page 318)
- Global and System Resource Security Records (see page 322)
- Database Resource Security Tables (see page 334)
- SATTDS (see page 340)
- SDUSDS (see page 341)
- SECACAB (see page 341)
- SECEQU (see page 342)
- SECRB (see page 346)
- SECRLST (see page 348)
- SECRITTD (see page 348)
- SPRFDS (see page 351)
- SRESDS (see page 351)
- SRGADS (see page 352)
- SRGPDS (see page 353)
- SROPDS (see page 353)
- SRSADS (see page 354)
- SUGPDS (see page 355)
- SUSDDS (see page 355)
- SUSRDS (see page 356)

About Security Information

Dictionary Connection for Storing Information

When you define resources or manipulate privileges in CA IDMS internal security, your session should be connected to the proper dictionary:

- For global resources -- The system dictionary or an application dictionary
- For system resources -- The system dictionary
- For non-SQL-defined database resources:
DMCL, database, database name table, area -- The system dictionary

Non-SQL-defined schema -- The application dictionary

For SQL-defined database resources -- The application dictionary or system dictionary

Note: If your session is connected to an application dictionary and you issue security statements specifying resources that must be defined in the system dictionary, the statements will be processed and the security information will be stored in the application dictionary. However, at runtime, the information will not be used.

Where Security Information is Maintained

CA IDMS maintains security definitions in these areas:

- The user catalog (SYSUSER.DDLSEC area)
- The DDLML, DDLCAT, and DDLCATX areas of the system dictionary and application dictionaries

Information about privileges on a resource is maintained in the same area as the resource definition.

User Catalog

Security information on global resources is maintained in these user catalog records:

- ATTRIBUTE
- DELUSER
- PROFILE
- RESOURCE
- RESOURCEAUTH
- USER
- USERGROUP

User catalog records reside in the SYSUSER.DDLSEC area and are accessible through subschema IDMSSECU, which is defined in dictionaries against which IDMSDIRL has been run.

System Dictionary

Security information on system resources and non-SQL-defined database resources is maintained in these system dictionary records:

- ATTRIBUTE
System dictionary security records reside in the SYSTEM.DDLDML area and are accessible through subschema IDMSSECS, which is defined in dictionaries against which IDMSDIRL has been run.

Security information on SQL-defined database resources is maintained in these tables of the catalog component of the dictionary:

- SYSTEM.RESOURCE
- SYSTEM.RESOURCEAUTH
- SYSTEM.RESOURCEGROUP
- SYSTEM.RESGROUPAUTH

Security tables for SQL-defined database resources reside in the DDLCAT area, and indexes on security tables reside in the DDLCATX area.

**Application Dictionary**

Security information on SQL-defined database resources is maintained in these tables of the catalog component of the dictionary:

- SYSTEM.RESOURCE
- SYSTEM.RESOURCEAUTH
- SYSTEM.RESOURCEGROUP
- SYSTEM.RESGROUPAUTH

Security tables for SQL-defined database resources reside in the DDLCAT area, and indexes on security tables reside in the DDLCATX area.

Security information on non-SQL-defined schemas is maintained in these system dictionary records:

- RESOURCE
- RESOURCEAUTH

These records are stored in the DDLDML area and are accessible through subschema IDMSSECS, which is defined in dictionaries against which IDMSDIRL has been run.
Summary

This table summarizes security information on CA IDMS resources, including where the information is stored and the privileges that apply to each resource:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Keyword</th>
<th>Location</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Global resources</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSADMIN</td>
<td>SYSA</td>
<td>SYSUSER.DDLSEC</td>
<td>SYSADMIN</td>
</tr>
<tr>
<td>User</td>
<td>USER</td>
<td>SYSUSER.DDLSEC</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2))</td>
</tr>
<tr>
<td>Group</td>
<td>GROU</td>
<td>SYSUSER.DDLSEC</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2))</td>
</tr>
<tr>
<td>User profile</td>
<td>UPRF</td>
<td>SYSUSER.DDLSEC</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2))</td>
</tr>
<tr>
<td><strong>System resources</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCADMIN</td>
<td>DCA</td>
<td>System dictionary</td>
<td>DCADMIN</td>
</tr>
<tr>
<td>System</td>
<td>SYST</td>
<td>System dictionary</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), SIGNON</td>
</tr>
<tr>
<td>System profile</td>
<td>SPRF</td>
<td>System dictionary</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2))</td>
</tr>
<tr>
<td>Application (1)</td>
<td>SAPP</td>
<td>System dictionary</td>
<td>EXECUTE</td>
</tr>
<tr>
<td>Activity</td>
<td>ACTI</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'SAPP')</td>
</tr>
<tr>
<td>Category(1)</td>
<td>CATE</td>
<td>System dictionary</td>
<td>EXECUTE</td>
</tr>
<tr>
<td>Access module (runtime)</td>
<td>SACC</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'CATE')</td>
</tr>
<tr>
<td>Dictionary load module</td>
<td>SLOD</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'CATE')</td>
</tr>
<tr>
<td>Program (load module)</td>
<td>SPGM</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'CATE')</td>
</tr>
<tr>
<td>Queue</td>
<td>QUEU</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'CATE')</td>
</tr>
<tr>
<td>Run unit</td>
<td>NRU</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'CATE')</td>
</tr>
<tr>
<td>Task</td>
<td>TASK</td>
<td>System dictionary</td>
<td>(EXECUTE on the associated 'CATE')</td>
</tr>
<tr>
<td><strong>Non-SQL-defined database resources</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>AREA</td>
<td>System dictionary</td>
<td>DBAREAD, DBAWRITE, USE</td>
</tr>
<tr>
<td>Database</td>
<td>DB</td>
<td>System dictionary</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DBTB</td>
</tr>
<tr>
<td>Resource</td>
<td>Keyword Location</td>
<td>Privileges</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Database name table</td>
<td>System dictionary</td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE)</td>
<td></td>
</tr>
<tr>
<td>DMCL</td>
<td>DMCL</td>
<td>System dictionary</td>
<td></td>
</tr>
<tr>
<td>Non-SQL-defined</td>
<td>NSCH</td>
<td>USE</td>
<td></td>
</tr>
<tr>
<td>schema</td>
<td></td>
<td>Application dictionary</td>
<td></td>
</tr>
<tr>
<td>SQL-defined</td>
<td></td>
<td>DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), OWNER</td>
<td></td>
</tr>
<tr>
<td>database resources</td>
<td></td>
<td>Application dictionary</td>
<td></td>
</tr>
<tr>
<td>Schema(3)</td>
<td>QSCH</td>
<td>EXECUTE, DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), REFERENCES(2)</td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td></td>
<td>EXECUTE, DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), REFERENCES(2)</td>
<td></td>
</tr>
<tr>
<td>module</td>
<td></td>
<td>EXECUTE, DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), REFERENCES(2)</td>
<td></td>
</tr>
<tr>
<td>(definition)</td>
<td></td>
<td>EXECUTE, DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), REFERENCES(2)</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>TABL</td>
<td>ALL [DEFINE (ALTER, CREATE, DISPLAY, DROP, USE(2)), REFERENCES, ACCESS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DELETE, INSERT, SELECT, UPDATE), OWNER(2)]</td>
<td></td>
</tr>
</tbody>
</table>

1. Resource group.

2. Privilege not meaningful for resource.

3. Resource and resource group.

Global and System Resource Security Records

Contents
- ATTRIBUTE (see page 323)
- DELUSER (see page 323)
- PROFILE (see page 324)
- RESGROUPAUTH (see page 325)
- RESOURCE (see page 326)
- RESOURCEAUTH (see page 327)
- RESOURCESGROUP (see page 329)
- USER (see page 330)
This section contains the descriptions of records containing security information on global resources, system resources, and non-SQL-defined database resources.

**ATTRIBUTE**

**Purpose**

An occurrence of this record represents an attribute defined in a profile.

**Access**

- IDMSSECU subschema (user profiles)
- IDMSSECS subschema (system profiles)
- Location mode: VIA PROFILE-ATTR
- Member of PROFILE-ATTR set

**Record elements**

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROFILENAME</td>
<td>PIC X(18)</td>
<td>Name of the profile.</td>
</tr>
<tr>
<td>KEYWORD</td>
<td>PIC X(8)</td>
<td>The keyword that identifies the attribute.</td>
</tr>
<tr>
<td>LENGTH</td>
<td>PIC S9(4) USAGE COMP</td>
<td>Number of characters in the attribute value.</td>
</tr>
<tr>
<td>VALUE</td>
<td>PIC X(32)</td>
<td>A user-assigned value for the attribute.</td>
</tr>
<tr>
<td>OVERRIDE</td>
<td>PIC X(1)</td>
<td>Defines whether the value may be overridden at runtime: 'Y' -- Value may be overridden at runtime 'N' -- Value is static and may not be overridden</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who created the attribute.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who last updated the attribute.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X(13)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

**DELUSER**

**Purpose**

An occurrence of this record represents a user or group that has been logically deleted from the user catalog. The SDEL task uses this information to determine which resource and resource group authorizations should be physically deleted.
Access

- IDMSSECU subschema
- Location mode: VIA IX-DELUSER
- Member of IX-DELUSER set

Record Elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>PIC X(18)</td>
<td>The authorization ID of a user or group that has been logically deleted.</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who created the record occurrence.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who last updated the record occurrence.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X(10)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

PROFILE

Purpose

An occurrence of this record represents a profile definition.

Access

- Subschema IDMSSECU (user profiles)
- Subschema IDMSSECS (system profiles)
- Location mode: CALC on PROFILENAME
- Owner of PROFILE-ATTR set

Record Elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROFILENAME</td>
<td>PIC X(18)</td>
<td>The name of the profile.</td>
</tr>
<tr>
<td>TYPE</td>
<td>PIC X(1)</td>
<td>The profile type: 'U' -- User profile 'S' -- System profile</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time created.</td>
</tr>
</tbody>
</table>
### RESGROUPAUTH

**Purpose**

An occurrence of this record represents an execute privilege granted to an authorization ID on a category or on application activities.

**Access**

- Subschema IDMSSECS
- Location mode: CALC on AUTHID
- Member of RESGROUP-AUTH set

**Record Elements**

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>PIC X(18)</td>
<td>The authorization ID of the user or group that holds EXECUTE privilege.</td>
</tr>
<tr>
<td>RESOURCETYPE</td>
<td>PIC X(4)</td>
<td>The code for the resource group to which the privilege applies: 'SAPP' -- Application</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'CATE' -- Category</td>
</tr>
<tr>
<td>RESOURCENAME</td>
<td>PIC X(60)</td>
<td>The name of the category or the name of the application with which the activities are associated.</td>
</tr>
<tr>
<td>RUNTIME</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>Runtime privilege that has been granted on the category or activity: 1 -- EXECUTE</td>
</tr>
<tr>
<td></td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>(Not applicable -- always 0)</td>
</tr>
<tr>
<td></td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>(Not applicable -- always 0)</td>
</tr>
<tr>
<td></td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>(Not applicable -- always 0)</td>
</tr>
<tr>
<td></td>
<td>PIC X(32)</td>
<td>A bit map that designates the activities within the application in RESOURCENAME to which the EXECUTE privilege applies.</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time created.</td>
</tr>
<tr>
<td>Element name</td>
<td>Picture and usage</td>
<td>Description of contents</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>UTIME</td>
<td>PIC X(64) USAGE</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td></td>
<td>BIT</td>
<td></td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who created the resource group authorization.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who last updated the resource group authorization.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X(16)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

**RESOURCE**

**Purpose**

An occurrence of this record represents a resource.

**Access**

- Subschema IDMSSECU (global resources)
- Subschema IDMSSECS (system and non-SQL-defined resources)
- Location mode: CALC on RESOURCETYPE and RESOURCENAME
- Member of IX-RESOURCE set
- Member of RESGROUP-RES set (IDMSSECS subschema only)
- Owner of RESOURCE-AUTH set

**Record Elements**

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESOUR</td>
<td>PIC X(4)</td>
<td>Resource type keyword.</td>
</tr>
<tr>
<td>CETYPE</td>
<td></td>
<td>Using IDMSSECS subschema:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'ACTI' 'QUEU'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'AREA' 'SACC'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'DB' 'SLOD'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'DBTB' 'SPGM'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'DCA' 'SPRF'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'DMCL' 'SYST'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'NRU' 'TASK'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'NSCH'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using IDMSSECU subschema:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'GROU' 'UPRF'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'SYSA' 'USER'</td>
</tr>
<tr>
<td>RESOUR</td>
<td>PIC X(60)</td>
<td>The resource name.</td>
</tr>
<tr>
<td>CENAME</td>
<td></td>
<td>For RESOURCETYPE 'ACTI', the value is the fully qualified activity name (for example, DCMT.SHUTDOWN).</td>
</tr>
<tr>
<td><strong>Element name</strong></td>
<td><strong>Picture and usage</strong></td>
<td><strong>Description of contents</strong></td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>GROUPTYPEIND</td>
<td>PIC X(8) USAGE BIT</td>
<td>Null value indicator for the GROUPTYPE field: B'11111111' -- Resource does not belong to a group</td>
</tr>
<tr>
<td>GROUPYPE</td>
<td>PIC X(4)</td>
<td>The group code of the resource group (if there is one) to which the resource belongs: 'CATE' -- Category 'SAPP' -- Application ' ' -- Resource does not belong to a group</td>
</tr>
<tr>
<td>GROUPNAMEIND</td>
<td>PIC X(8) USAGE BIT</td>
<td>Null value indicator for the GROUPNAME field: B'11111111' -- No value in GROUPNAME</td>
</tr>
<tr>
<td>GROUPNAME</td>
<td>PIC X(60)</td>
<td>The name of the resource group (if there is one) to which the resource belongs.</td>
</tr>
<tr>
<td>NUMBER</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>For a resource assigned to a category (NRU, QUEU, SACC, SLOD, SPGM, or TASK), the unique number that identifies the category with which the resource is associated. For an activity, the unique activity number within the application. For a resource that is not an activity or does not belong to a category, the value is 0.</td>
</tr>
<tr>
<td>AMRUNABLE</td>
<td>PIC X(1)</td>
<td>(Reserved -- initialized to 'N'.)</td>
</tr>
<tr>
<td>STATUS</td>
<td>PIC X(1)</td>
<td>(Reserved -- initialized to 'A'.)</td>
</tr>
<tr>
<td>AMGRAINTABLE</td>
<td>PIC X(1)</td>
<td>(Reserved -- initialized to 'N'.)</td>
</tr>
<tr>
<td>AMLASTCHANGE</td>
<td>PIC X(8)</td>
<td>(Reserved -- initialized to binary zeros.)</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTFIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who created the resource.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who last updated the resource.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X(21)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

**RESOURCEAUTH**

**Purpose**

An occurrence of this record represents the privileges that have been granted to an authorization ID on a resource.

**Access**

- Subschema IDMSSECU (privileges on global resources)
Subschema IDMSSECS (privileges on system resources and non-SQL-defined database resources)

CALC on AUTHID

Member of RESOURCE-AUTH set

Record Elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>PIC X(18)</td>
<td>The authorization ID of the user or group to whom the privileges have been granted.</td>
</tr>
<tr>
<td>RESOUR CETYPE</td>
<td>PIC X(4)</td>
<td>The resource type keyword. Using IDMSSECS subschema: 'AREA' 'DMCL' 'DB' 'NSCH' 'DBTB' 'SPRF' 'DCA' 'SYST' Using IDMSSECU subschema: 'GROU' 'UPRF' 'SYSA' 'USER'</td>
</tr>
<tr>
<td>RESOUR CENAME</td>
<td>PIC X(60)</td>
<td>The resource name.</td>
</tr>
<tr>
<td>RUNTIM EAUTH</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>(Not applicable -- initialized to binary zeros. EXECUTE privileges on categories and activities are stored in the RESGROUPAUTH record.)</td>
</tr>
<tr>
<td>RUNTIM EAUTHW</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>(Not applicable -- initialized to binary zeros.)</td>
</tr>
<tr>
<td>DEFNAUTH TH</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>Definition privileges that have been granted on the resource: 1 -- CREATE 2 -- ALTER 4 -- DROP 8 -- DISPLAY 16 -- USE ( 2n ) -- (The sum of two or more of the above, representing multiple privileges) 31 -- DEFINE (all definition privileges) Note: DEFINE privilege always includes USE; however, USE is meaningful only for AREA, DB, DBTB, DMCL, and NSCH.</td>
</tr>
<tr>
<td>DEFNAUTH THW</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>Definition privileges a user represented by AUTHID may grant to other authorization IDs.</td>
</tr>
<tr>
<td>OTHERAUTH</td>
<td>PIC S9(4) USAGE COMP SYNC</td>
<td>Special privileges: 1 -- SYSADMIN (Resource type SYSA only) 4 -- DBADMIN (Resource type DB only) 8 -- DCADMIN (Resource type DCA only)</td>
</tr>
<tr>
<td>Element name</td>
<td>Picture and usage</td>
<td>Description of contents</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>RESOURCETY</td>
<td>PIC X(4)</td>
<td>The code for the resource group: 'CATE' -- Category 'SAPP' -- Application</td>
</tr>
<tr>
<td>RESOURCENA</td>
<td>PIC X(60)</td>
<td>The name of the application or category.</td>
</tr>
<tr>
<td>OWNER</td>
<td>PIC X(18)</td>
<td>(Not applicable -- initialized to spaces.)</td>
</tr>
<tr>
<td>CATEGORYNO</td>
<td>PIC S9(4) USAGE</td>
<td>For categories, the unique number assigned to the category. For applications, the value is 0.</td>
</tr>
<tr>
<td></td>
<td>COMP SYNC</td>
<td></td>
</tr>
</tbody>
</table>

**RESOURCETYPE**

**Purpose**

An occurrence of this record represents an application or category that has been defined with a CREATE RESOURCE statement. Applications are defined when the first CREATE RESOURCE ACTIVITY `application-name.activity-name` statement is issued.

**Access**

- Subschema IDMSSECS
- Location mode: CALC on RESOURCETYPE and RESOURCENAME
- Member of IX-RESGROUP set; owner of RESGROUP-AUTH and RESGROUP-RES sets

**Record Elements**
USER

Purpose

An occurrence of this record represents a user or group defined within a CA IDMS security domain. Users and groups are defined by the same record so that privileges on resources -- whether to a user or a group -- can be granted and enforced in the same manner. Note that if the occurrence represents a group, there is no associated password or user profile.

Access

- IDMSSECU subschema
- Location mode: CALC on AUTHID
- Owner of the USER-GROUP and GROUP-USER sets

Record elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>PIC X(18)</td>
<td>The authorization ID of a user or group.</td>
</tr>
<tr>
<td>TYPE</td>
<td>PIC X(1)</td>
<td>Identifies the authorization ID as representing a user or group: 'G' -- Group</td>
</tr>
<tr>
<td>STATUS</td>
<td>PIC X(1)</td>
<td>Current status of the authorization ID: 'A' -- Active 'D' -- Logically deleted</td>
</tr>
<tr>
<td>PASSWO RD</td>
<td>PIC X(8)</td>
<td>The encrypted form of the plain text password, or: Binary zeros, when TYPE field is 'U' and the password unassigned or null Spaces, when TYPE field is 'G' or when TYPE field is 'U' and AUTHID is 'SYSTEM'</td>
</tr>
<tr>
<td>NAME</td>
<td>PIC X(32)</td>
<td>The full name associated with the user, if specified; initialized to spaces when not specified for the user or when the authorization ID represents a group.</td>
</tr>
<tr>
<td>Element name</td>
<td>Picture and usage</td>
<td>Description of contents</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>DESCRITION</td>
<td>PIC X (40)</td>
<td>Textual information that the security administrator may associate with the authorization ID but that has no special meaning to the security system; initialized to spaces when not specified for the user or group.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X (2)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
<tr>
<td>PROFILE NAME</td>
<td>PIC X (18)</td>
<td>The name of the user profile assigned to this authorization ID; initialized to spaces for all groups and when not specified for a user.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X (4)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X (8)</td>
<td>(Reserved -- initialized to binary zeros.)</td>
</tr>
<tr>
<td>GTIME</td>
<td>PIC X (64) USAGE BIT</td>
<td>(Reserved -- initialized to binary zeros.)</td>
</tr>
<tr>
<td>PTIME</td>
<td>PIC X (64) USAGE BIT</td>
<td>(Reserved -- initialized to binary zeros.)</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X (64) USAGE BIT</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>PIC X (64) USAGE BIT</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X (18)</td>
<td>ID of the user who created the user definition.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X (18)</td>
<td>ID of the user who last updated the user definition.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X (16)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

**Special Record Occurrences**

These are special occurrences of the USER record:

- Group 'PUBLIC' -- created on the first grant of privilege to PUBLIC (group 'PUBLIC' cannot be dropped):
  - AUTHID: 'PUBLIC'
  - TYPE: 'G'
- STATUS: (space)
- PASSWORD: (spaces)
- NAME: (spaces)
- DESCRIPTION: 'System Category bit map owner'
- PROFILENAME: (spaces)
- CUSER: 'SYSTEM'
- UUSER: 'SYSTEM'
- Other fields as noted in the USER record description above

- User 'SYSTEM' -- created by the first CREATE RESOURCE CATEGORY statement (user 'SYSTEM' cannot be dropped):
  - AUTHID: 'SYSTEM'
  - TYPE: 'U'
  - STATUS: (space)
  - PASSWORD: (spaces)
  - NAME: (spaces)
  - DESCRIPTION: 'PUBLIC Group'
  - PROFILENAME: (spaces)
  - CUSER: 'SYSTEM'
  - UUSER: 'SYSTEM'
  - Other fields as noted in the USER record description above

**USERDATA**

**Purpose**

An occurrence of this record can represent:

- The association of a user or group with all categories on which the user or group holds execution privilege
- In one case, an occurrence, with the value 'SYSTEM' in the AUTHID field and 'C' in the TYPE field, that represents all categories defined to the system
- The association of a user and a system profile, as specified in a GRANT SIGNON statement
Access

- IDMSSECS subschema
- Location mode: CALC on AUTHID
- Sets: None

Record Elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>PIC X(18)</td>
<td>The authorization ID of the user.</td>
</tr>
<tr>
<td>LENGTH</td>
<td>PIC S9(4) USAGE COMP</td>
<td>The number of bytes (a multiple of 32) currently allocated to the category bit map; initialized to 0 when TYPE is 'P'.</td>
</tr>
<tr>
<td>TYPE</td>
<td>PIC X(1)</td>
<td>Type of user data: 'C' -- Category 'P' -- Profile</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>PIC X(8)</td>
<td>The identifier of a CA IDMS system; initialized to spaces when TYPE is 'C'.</td>
</tr>
<tr>
<td>PROFILENAME</td>
<td>PIC X(18)</td>
<td>The name of the system profile; initialized to spaces when TYPE is 'C'.</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>PIC X(64) USAGE BIT</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who created the record occurrence.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who last updated the record occurrence.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X(5)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
<tr>
<td>CATEGORIES</td>
<td>PIC X OCCURS 0 TO 4096 DEPENDING ON LENGTH</td>
<td>The bit map used to identify categories. Each bit in the bit map corresponds to a category number from 1 to 32,767.</td>
</tr>
</tbody>
</table>

Special Record Occurrence

The following occurrence of the USERDATA record is created by the first CREATE RESOURCE CATEGORY statement and is used to manage category number assignment:

- AUTHID: 'SYSTEM'
- LENGTH: (variable -- initially 32)
- TYPE: 'C'
- SYSTEM: (spaces)
- PROFILENAME: (spaces)
- CUSER: 'SYSTEM'
USERGROUP

Purpose

An occurrence of this record represents the association of a user with a group.

Access

- IDMSSECU subschema
- Location mode: VIA USER-GROUP
- Member of the USER-GROUP and GROUP-USER sets

Record Elements

<table>
<thead>
<tr>
<th>Element name</th>
<th>Picture and usage</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROUPID</td>
<td>PIC X(18)</td>
<td>The authorization ID of the group of which the user is a member.</td>
</tr>
<tr>
<td>USERID</td>
<td>PIC X(18)</td>
<td>The authorization ID of the user who is a member of the group.</td>
</tr>
<tr>
<td>DEFAULTGROU</td>
<td>PIC X(1)</td>
<td>Default group indicator:</td>
</tr>
<tr>
<td>P</td>
<td></td>
<td>'D' -- Default group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>' ' -- Not the default group</td>
</tr>
<tr>
<td>CTIME</td>
<td>PIC X(64) USAGE</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>PIC X(64) USAGE</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who created the user and group association.</td>
</tr>
<tr>
<td>UUSER</td>
<td>PIC X(18)</td>
<td>ID of the user who last updated the user and group association.</td>
</tr>
<tr>
<td>FILLER</td>
<td>PIC X(11)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

Database Resource Security Tables

Contents

- SYSTEM.RESGROUPAUTH (see page 335)
- SYSTEM.RESOURCE (see page 336)
- SYSTEM.RESOURCEAUTH (see page 337)
- SYSTEM.RESOURCEGROUP (see page 339)
This section contains the descriptions of tables maintained in the catalog component of the dictionary for use by CA IDMS centralized security.

**SYSTEM.RESGROUPAUTH**

**Purpose**

A row of this table represents the privileges that have been granted to an authorization ID on an SQL-defined database resource group. A row of this table is stored only when a CREATE SCHEMA statement is issued.

**Columns**

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>CHAR(18)</td>
<td>The authorization ID of the user or group that holds one or more privileges on the resource group.</td>
</tr>
<tr>
<td>RESOURCETYPE</td>
<td>CHAR(4)</td>
<td>The code for the resource group to which the privileges apply: 'QSCH' -- SQL Schema</td>
</tr>
<tr>
<td>RESOURSNAME</td>
<td>CHAR(60)</td>
<td>The schema name.</td>
</tr>
<tr>
<td>RUNTIMEAUTH</td>
<td>BINARY(2)</td>
<td>Runtime privileges that have been granted on the schema: 143 -- ALL (ACCESS and OWNER privileges)</td>
</tr>
<tr>
<td>RUNTIMEAUTHW</td>
<td>BINARY(2)</td>
<td>The runtime privileges a user represented by AUTHID may grant to other authorization IDs: 143 -- ALL</td>
</tr>
<tr>
<td>DEFINAUTH</td>
<td>BINARY(2)</td>
<td>Definition privileges that have been granted on the schema and the resources associated with the schema: 191 -- ALL (DEFINE, REFERENCES, and OWNER)</td>
</tr>
</tbody>
</table>

When a CREATE SCHEMA statement is executed, a row is inserted into SYSTEM.RESGROUPAUTH. AUTHID is the ID of the user executing the statement, and RUNTIMEAUTH is 143 (ALL). If ownership of the schema is transferred, AUTHID is modified but RUNTIMEAUTH remains 143.

ALL includes schema ownership, which can be transferred, and these access privileges, which can be granted as appropriate on tables and access modules:
- SELECT
- EXECUTE
- INSERT
- UPDATE
- DELETE

The definition privileges a user represented by AUTHID may grant to other authorization IDs:
- 191 -- ALL (definition privileges, REFERENCES, and OWNER)

ALL includes schema ownership, which can be transferred, and these privileges, which can be granted as appropriate on tables and access modules:
- CREATE
### SYSTEM.RESOURCE

**Purpose**

A row of this table represents a resource (a table or access module) associated with a schema, or the schema itself.

**Columns**

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESOURCE TYPE</td>
<td>CHAR (4)</td>
<td>Resource type keyword: 'DACC' -- Access module 'QSCH' -- SQL Schema 'TABL' -- Table</td>
</tr>
<tr>
<td>RESOURCE NAME</td>
<td>CHAR (60)</td>
<td>The resource name; for a table or access module, the name is fully qualified with <code>schema-name</code>.</td>
</tr>
</tbody>
</table>

**Columns**

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DROP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISPLAY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REFERENCES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OTHERA</td>
<td>BINA (Not applicable -- initialized to binary zeros.)</td>
<td></td>
</tr>
<tr>
<td>UTH</td>
<td>RY (2)</td>
<td></td>
</tr>
<tr>
<td>FUNCTIONS</td>
<td>BINA (Not applicable -- initialized to binary zeros.)</td>
<td></td>
</tr>
<tr>
<td>ONS</td>
<td>RY (32)</td>
<td></td>
</tr>
<tr>
<td>CTIME</td>
<td>TIME Time created.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STA MP</td>
<td></td>
</tr>
<tr>
<td>UTIME</td>
<td>TIME Time of the last update.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STA MP</td>
<td></td>
</tr>
<tr>
<td>CUSER</td>
<td>CHAR (18)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID of the user who created the resource group authorization.</td>
<td></td>
</tr>
<tr>
<td>UUSER</td>
<td>CHAR (18)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID of the user who last updated the resource group authorization.</td>
<td></td>
</tr>
<tr>
<td>FILLER</td>
<td>BINA (Reserved -- initialized to spaces.)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RY (16)</td>
<td></td>
</tr>
</tbody>
</table>
### Column name | Data type | Description of contents
--- | --- | ---
GROUPTYP | CHAR(4) | The resource group to which the resource belongs: 'QSCH' -- SQL schema Null when RESOURCETYPE is 'QSCH'
GROUPNAME | CHAR(60) | The name of the resource group (SQL Schema) to which the resource belongs; null when RESOURCETYPE is 'QSCH'
NUMBER | SMALLINT | (Not applicable -- initialized to zero.)
AMRUNNABLE | CHAR(1) | (Reserved -- initialized to 'N'.)
STATUS | CHAR(1) | (Reserved -- initialized to 'A'.)
AMGRANTABLE | CHAR(1) | (Reserved -- initialized to 'N'.)
AMLASTCHANGETIMATEMP | TIMESTAMP | (Reserved -- initialized to binary zeros.)
CTIME | TIMESTAMP | Time created.
UTIME | TIMESTAMP | Time of the last update.
CUSER | CHAR(18) | ID of the user who created the resource.
UUSER | CHAR(18) | ID of the user who last updated the resource.
FILLER | BINARY(21) | (Reserved -- initialized to spaces.)

---

**SYSTEM.RESOURCEAUTH**

**Purpose**

A row of this table represents the privileges that have been granted to an authorization ID on an SQL-defined database resource.

**Columns**

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHID</td>
<td>CHAR(18)</td>
<td>The authorization ID of the user or group to whom the privileges have been granted.</td>
</tr>
<tr>
<td>RESOURCETYPE</td>
<td>CHAR(4)</td>
<td>The resource type keyword: 'DACC' -- Access module 'QSCH' -- Schema 'TABL' -- Table</td>
</tr>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description of contents</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>RESOURCENAME</td>
<td>CHAR (60)</td>
<td>The resource name.</td>
</tr>
<tr>
<td>RUNTIMEAUTH</td>
<td>BINARY (2)</td>
<td>Runtime privileges that have been granted on the resource. For RESOURCETYPE 'DACC': 1 -- EXECUTE(1) For RESOURCETYPE 'QSCH'(2) and 'TABL': 1 -- SELECT 2 -- INSERT 4 -- UPDATE 8 -- DELETE nn -- (The sum of two or more of the above, representing multiple privileges) 15 -- ACCESS (all of the above) 128 -- OWNER(3) 143 -- ALL (ACCESS and OWNER)</td>
</tr>
<tr>
<td>RUNTIMEAUTHW</td>
<td>BINARY (2)</td>
<td>The runtime privileges a user represented by AUTHID may grant to other authorization IDs.(4)</td>
</tr>
<tr>
<td>DEFNAUTH</td>
<td>BINARY (2)</td>
<td>Definition privileges that have been granted on the resource: 1 -- CREATE 2 -- ALTER 4 -- DROP 8 -- DISPLAY 16 -- USE nn -- (The sum of two or more of the above, representing multiple privileges) 31 -- DEFINE (CREATE, ALTER, DROP, DISPLAY, USE) 32 -- REFERENCES 63 -- DEFINE and REFERENCES 128 -- OWNER(5) 191 -- ALL (DEFINE, REFERENCES, and OWNER)(6)</td>
</tr>
<tr>
<td>DEFNAUTHW</td>
<td>BINARY (2)</td>
<td>The definition privileges a user represented by AUTHID may grant to other authorization IDs.(5) (6)</td>
</tr>
<tr>
<td>OTHERAUTH</td>
<td>BINARY (2)</td>
<td>(Not applicable -- initialized to binary zeros.)</td>
</tr>
<tr>
<td>CTIME</td>
<td>TIMESTAMP</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>TIMESTAMP</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>CHAR (18)</td>
<td>ID of the user who created the resource authorization.</td>
</tr>
<tr>
<td>UUSER</td>
<td>CHAR (18)</td>
<td>ID of the user who last updated the resource authorization.</td>
</tr>
<tr>
<td>FILLER</td>
<td>BINARY (16)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>
(1) An occurrence of a row with RESOURCETYPE 'DACC' and RUNTIMEAUTH 1 (EXECUTE) results when privilege is granted on the access module directly as an SQL-defined database resource, not as part of a category, which is a system resource.

(2) An occurrence of a row with RESOURCETYPE 'QSCH' and a nonzero value in RUNTIMEAUTH results only when a schema is created. RUNTIMEAUTH is 143 (ALL) and remains 143 even if schema ownership is transferred because the runtime privileges cannot be revoked. All other occurrences of a row with RESOURCETYPE 'QSCH' refer to definition privileges only and, therefore, contain zero in RUNTIMEAUTH and RUNTIMEAUTHW.

(3) The OWNER privilege for a table can be granted only with a GRANT ALL PRIVILEGES statement and revoked only with a REVOKE ALL PRIVILEGES statement. However, it is possible for the value of RUNTIMEAUTH (and RUNTIMEAUTHW) to be 128 (OWNER) if all privileges are granted on a table and then access privileges are revoked.

(4) Runtime privileges for RESOURCETYPE 'QSCH' are grantable only in the sense that they are automatically transferred when ownership of the schema is transferred.

(5) OWNER privilege is never assigned to DEFNAUTH (or DEFNAUTHW) for resource types 'DACC' and 'TABL'.

(6) When an SQL schema is created, the value of RESOURCETYPE is 'QSCH' and the value of DEFNAUTH is 191 (ALL: DEFINE, REFERENCES, and OWNER). REFERENCES and definition privileges can be revoked; ownership can be transferred but not revoked.

**SYSTEM.RESOURCEGROUP**

**Purpose**

A row of this table represents an SQL schema resource group. A row of this table is stored when one of these events occurs:
The first GRANT statement is issued for an access module or table associated with an undefined schema.

A CREATE SCHEMA statement is issued before any GRANT statements for associated access modules or tables.

## Columns

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESOURCETYPE</td>
<td>CHAR(4)</td>
<td>The code for the resource group: 'QSCH' -- SQL schema</td>
</tr>
<tr>
<td>RESOURCENAME</td>
<td>CHAR(60)</td>
<td>The schema name.</td>
</tr>
<tr>
<td>OWNER</td>
<td>CHAR(18)</td>
<td>The authorization ID of the user or group who owns the schema.</td>
</tr>
<tr>
<td>CATEGORYNO</td>
<td>SMALLINT</td>
<td>(Not applicable -- initialized to zero.)</td>
</tr>
<tr>
<td>FUNCTIONASSIGNMENT</td>
<td>BINARY(1)</td>
<td>(Not applicable -- initialized to space.)</td>
</tr>
<tr>
<td>CTIME</td>
<td>TIMESTAMP</td>
<td>Time created.</td>
</tr>
<tr>
<td>UTIME</td>
<td>TIMESTAMP</td>
<td>Time of the last update.</td>
</tr>
<tr>
<td>CUSER</td>
<td>CHAR(18)</td>
<td>ID of the user who created the resource group.</td>
</tr>
<tr>
<td>UUSER</td>
<td>CHAR(18)</td>
<td>ID of the user who last updated the resource group.</td>
</tr>
<tr>
<td>FILLER</td>
<td>BINARY(19)</td>
<td>(Reserved -- initialized to spaces.)</td>
</tr>
</tbody>
</table>

### SATTDS

COPY #SATTS

**********************************************************************
***                  SATT DSECT                             PROFILE ATTRIBUTE***
***                  PROFILE NAME                          COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.***
***                  *
***                  *
***                  *
**********************************************************************

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>SATT</th>
<th>DSECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>SATTPROF DS CL18</td>
<td>PROFILE NAME</td>
<td></td>
</tr>
<tr>
<td>000012</td>
<td>SATTKEYW DS CL8</td>
<td>ATTRIBUTE KEYWORD</td>
<td></td>
</tr>
<tr>
<td>00001A</td>
<td>SATTLTH DS H</td>
<td>ATTRIBUTE LENGTH</td>
<td></td>
</tr>
<tr>
<td>00001C</td>
<td>SATTVALU DS CL32</td>
<td>ATTRIBUTE VALUE</td>
<td></td>
</tr>
<tr>
<td>00003C</td>
<td>SATTOVER DS CL1</td>
<td>ATTRIBUTE OVERRIDE FLAG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>'Y' IF RUNTIME OVERRIDE ALLOWED</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>'N' IF OTHERWISE</td>
<td></td>
</tr>
<tr>
<td>00003D</td>
<td>SATTCTIM DS CL8</td>
<td>TIME/DATA ATTRIBUTE WAS CREATED</td>
<td></td>
</tr>
<tr>
<td>000045</td>
<td>SATTUTIM DS CL8</td>
<td>TIME/DATA ATTRIBUTE WAS LAST UPDATED</td>
<td></td>
</tr>
<tr>
<td>00004D</td>
<td>SATTCAID DS CL18</td>
<td>CREATOR OF THE ATTRIBUTE</td>
<td></td>
</tr>
<tr>
<td>00005F</td>
<td>SATTUAID DS CL18</td>
<td>LAST UPDATOR OF THE ATTRIBUTE</td>
<td></td>
</tr>
</tbody>
</table>
**SDUSDS**

COPY #SDUSDS

*********************************************************************

*** SDELUDS DSECT
*** USER
*** COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.
***

Offset | Value
-------|------
06/24/91 | SDUSAUID DS CL18 AUTHORIZATION ID
000000 | SDUSCTIM DS CL8 TIME/DATA AUTH. ID WAS CREATED
000012 | SDUSUTIM DS CL8 TIME/DATA AUTH. ID WAS LAST UPDATED
000022 | SDUSAVID DS CL18 CREATOR OF THE AUTH. ID.
000034 | SDUSAUID DS CL18 LAST UPDATOR OF THE AUTH. ID
000046 | DS XL10 * UNUSED
00050 | SDUSDSLN EQU *-SDUS * LENGTH OF DSECT

*********************************************************************

**SECACAB**

COPY #SECACAB

*********************************************************************

*** SECACAB - SECURITY ACCESS AUTHORITY BYTES
*** THIS DSECT COVERS THE 6 BYTES OF ACCESS AUTHORITY
*** FLAGS FOUND IN THE RESOURCE-TYPE RECORD/BLOCK, THE
*** RESOURCE RECORD, THE AUTHORITY RECORD, AND VARIOUS FIELDS
*** IN THE SECURITY REQUEST BLOCK (#SECRB)
*** COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.

Offset | Value
-------|------
22 | **RUN-TIME AUTHORITIES**

000000 | **SAARSEL** #FLAG SACRSEL SELECT
000001 | **SAARSEL** DS OXL1
000002 | **SAARSEL** EQU SACRSEL
000003 | **SAAREXE** #FLAG SACREXE EXECUTE
000004 | **SAAREXE** DS OXL1
000005 | **SAAREXE** EQU SACREXE
000006 | **SAARINS** #FLAG SARCINS INSERT
000007 | **SAARINS** DS OXL1
000008 | **SAARINS** EQU SARCINS
000009 | **SAARUPD** #FLAG SACRUPD UPDATE
000010 | **SAARUPD** DS OXL1
000011 | **SAARUPD** EQU SACRUPD
000012 | **SAARDEL** #FLAG SACRDEL DELETE
000013 | **SAARDEL** DS OXL1

06/08/89 13:35:

000000 | SECACAB DSECT
SECEQU
**PRIMARY FUNCTION CODES**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>SFCINIT EQU 2</td>
<td>System initialization</td>
</tr>
<tr>
<td>00003</td>
<td>SFCCHECK EQU 3</td>
<td>Authorization check</td>
</tr>
<tr>
<td>00004</td>
<td>SFCSIGNON EQU 4</td>
<td>User SIGNON -</td>
</tr>
<tr>
<td>00005</td>
<td>SFCSIGNOFF EQU 5</td>
<td>User SIGNOFF</td>
</tr>
<tr>
<td>00006</td>
<td>SFCBLDNM EQU 6</td>
<td>Build fully qualified res name</td>
</tr>
<tr>
<td>00007</td>
<td>SFCXTRCT EQU 7</td>
<td>Get job card user-id</td>
</tr>
<tr>
<td>00008</td>
<td>SFCSTATS EQU 8</td>
<td>Get ESM status</td>
</tr>
<tr>
<td>00009</td>
<td>SFCCRYPT EQU 9</td>
<td>Password encryption</td>
</tr>
<tr>
<td>0000A</td>
<td>SFCPWVER EQU 10</td>
<td>Password reverification</td>
</tr>
<tr>
<td>0000B</td>
<td>SFCAPMAP EQU 11</td>
<td>Get/build application bitmap</td>
</tr>
<tr>
<td>0000C</td>
<td>SFCBULK EQU 12</td>
<td>Bulk check - access mod security</td>
</tr>
<tr>
<td>0000D</td>
<td>SFCAMCHK EQU 13</td>
<td>Access module runnability check</td>
</tr>
</tbody>
</table>

**SECONDARY FUNCTION CODES (FOR UTILITY ROUTINES)**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00002</td>
<td>SUTLFLDL EQU 2</td>
<td>Backscan variable length field</td>
</tr>
<tr>
<td>00004</td>
<td>SUTLBLDN EQU 4</td>
<td>Build fully qualified resname</td>
</tr>
</tbody>
</table>

**ACCESS AUTHORITY BITS**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>SACRSEL EQU 'X'01'</td>
<td>RUN-TIME - SELECT</td>
</tr>
<tr>
<td>00001</td>
<td>SACREXE EQU 'X'01'</td>
<td>- EXECUTE</td>
</tr>
<tr>
<td>00002</td>
<td>SACRINS EQU 'X'02'</td>
<td>- INSERT</td>
</tr>
<tr>
<td>00004</td>
<td>SACRUPD EQU 'X'04'</td>
<td>- UPDATE</td>
</tr>
<tr>
<td>00008</td>
<td>SACRDEL EQU 'X'08'</td>
<td>- DELETE</td>
</tr>
<tr>
<td>0000F</td>
<td>SACRACC EQU 'X'0F'</td>
<td>- ACCESS</td>
</tr>
<tr>
<td>00080</td>
<td>SACROWN EQU 'X'80'</td>
<td>- OWNER</td>
</tr>
<tr>
<td>0008F</td>
<td>SACRALL EQU 'X'8F'</td>
<td></td>
</tr>
<tr>
<td>00001</td>
<td>SACDCRE EQU 'X'01'</td>
<td>DEFINITION - CREATE</td>
</tr>
<tr>
<td>00002</td>
<td>SACDALT EQU 'X'02'</td>
<td>- ALTER</td>
</tr>
<tr>
<td>00004</td>
<td>SACDDRO EQU 'X'04'</td>
<td>- DROP</td>
</tr>
<tr>
<td>00008</td>
<td>SACDDIS EQU 'X'08'</td>
<td>- DISPLAY</td>
</tr>
<tr>
<td>00010</td>
<td>SACUSE EQU 'X'10'</td>
<td>- USE</td>
</tr>
<tr>
<td>0001F</td>
<td>SACDEF EQU 'X'1F'</td>
<td>- DEFINE</td>
</tr>
<tr>
<td>00020</td>
<td>SACDEF EQU 'X'20'</td>
<td>- REFERENCES</td>
</tr>
<tr>
<td>00080</td>
<td>SACDOWM EQU 'X'80'</td>
<td>- OWNER</td>
</tr>
<tr>
<td>000BF</td>
<td>SACDALL EQU 'X'BF'</td>
<td></td>
</tr>
<tr>
<td>00001</td>
<td>SACASYS EQU 'X'01'</td>
<td>ADMINISTRATIVE - SYSDM</td>
</tr>
<tr>
<td>00002</td>
<td>SACADDA EQU 'X'02'</td>
<td>- DDDM</td>
</tr>
<tr>
<td>00004</td>
<td>SACADBA EQU 'X'04'</td>
<td>- DDDB</td>
</tr>
<tr>
<td>00008</td>
<td>SACACDA EQU 'X'08'</td>
<td>- DCADM</td>
</tr>
<tr>
<td>00010</td>
<td>SACASIG EQU 'X'10'</td>
<td>- SIGNON</td>
</tr>
<tr>
<td>00020</td>
<td>SACADBR EQU 'X'20'</td>
<td>- DBAREAD</td>
</tr>
<tr>
<td>00040</td>
<td>SACADB EQU 'X'40'</td>
<td>- DBAWRITE</td>
</tr>
<tr>
<td>0007F</td>
<td>SACALL EQU 'X'7F'</td>
<td></td>
</tr>
</tbody>
</table>

**RESOURCE TYPE NUMBERS**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>SRTNUDEF EQU 32768</td>
<td>Number for user defined resource</td>
</tr>
<tr>
<td>00001</td>
<td>SRTNSPEC EQU 01</td>
<td>SPECIAL</td>
</tr>
<tr>
<td>00002</td>
<td>SRTNDCA EQU 02</td>
<td>DC Admin</td>
</tr>
<tr>
<td>00003</td>
<td>SRTNSYSA EQU 03</td>
<td>System admin</td>
</tr>
<tr>
<td>0000A</td>
<td>SRTNLSP EQU 10</td>
<td>Highest 'SPECIAL' resource</td>
</tr>
<tr>
<td>00080</td>
<td>SRTNDDDA EQU 11</td>
<td>DD Admin</td>
</tr>
<tr>
<td>0008C</td>
<td>SRTNSNDS EQU 12</td>
<td>Subschem</td>
</tr>
<tr>
<td>0000D</td>
<td>SRTNDAPP EQU 13</td>
<td>ADS application</td>
</tr>
</tbody>
</table>
### CA IDMS - 19.0

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000E</td>
<td>SRTNMSG</td>
<td>EQU 14 Message</td>
</tr>
<tr>
<td>0000F</td>
<td>SRTNDIAL</td>
<td>EQU 15 ADS Dialog</td>
</tr>
<tr>
<td>00010</td>
<td>SRTNDATT</td>
<td>EQU 16 Attribute</td>
</tr>
<tr>
<td>00011</td>
<td>SRTNCLA</td>
<td>EQU 17 Class</td>
</tr>
<tr>
<td>00012</td>
<td>SRTNDELETE</td>
<td>EQU 18 Element</td>
</tr>
<tr>
<td>00013</td>
<td>SRTNDLOAD</td>
<td>EQU 19 Load module</td>
</tr>
<tr>
<td>00014</td>
<td>SRTNMAP</td>
<td>EQU 20 Map</td>
</tr>
<tr>
<td>00015</td>
<td>SRTNMOD</td>
<td>EQU 21 Module</td>
</tr>
<tr>
<td>00016</td>
<td>SRTNDPGM</td>
<td>EQU 22 Program</td>
</tr>
<tr>
<td>00017</td>
<td>SRTNREC</td>
<td>EQU 23 Record</td>
</tr>
<tr>
<td>00018</td>
<td>SRTNDE</td>
<td>EQU 24 User defined entity</td>
</tr>
<tr>
<td>00019</td>
<td>SRTNDFIL</td>
<td>EQU 25 File</td>
</tr>
<tr>
<td>0001A</td>
<td>SRTNDFIL</td>
<td>EQU 26 Panel</td>
</tr>
<tr>
<td>0001B</td>
<td>SRTNDSYS</td>
<td>EQU 27 IDD system</td>
</tr>
<tr>
<td>0001C</td>
<td>SRTNUSR</td>
<td>EQU 28 IDD user</td>
</tr>
<tr>
<td>0001D</td>
<td>SRTNDES</td>
<td>EQU 29 Destination</td>
</tr>
<tr>
<td>0001E</td>
<td>SRTNDLIN</td>
<td>EQU 30 Line</td>
</tr>
<tr>
<td>0001F</td>
<td>SRTNDLTE</td>
<td>EQU 31 Logical terminal</td>
</tr>
<tr>
<td>00020</td>
<td>SRTNDPTE</td>
<td>EQU 32 Physical terminal</td>
</tr>
<tr>
<td>00021</td>
<td>SRTNDQUE</td>
<td>EQU 33 Queue</td>
</tr>
<tr>
<td>00022</td>
<td>SRTNTASK</td>
<td>EQU 34 Task</td>
</tr>
<tr>
<td>00023</td>
<td>SRTNDACT</td>
<td>EQU 35 COBOL syntax, Culprit over</td>
</tr>
<tr>
<td>00024</td>
<td>SRTNLC</td>
<td>EQU 36 LAST DICTIONARY RESOURCE TYPE NO.</td>
</tr>
<tr>
<td>00025</td>
<td>SRTNACC</td>
<td>EQU 37 Access module</td>
</tr>
<tr>
<td>00026</td>
<td>SRTNDMCL</td>
<td>EQU 38 DMCL</td>
</tr>
<tr>
<td>00027</td>
<td>SRTNAREA</td>
<td>EQU 39 Area</td>
</tr>
<tr>
<td>00028</td>
<td>SRTNDB</td>
<td>EQU 40 Database</td>
</tr>
<tr>
<td>00029</td>
<td>SRTNDB</td>
<td>EQU 41 Last database type</td>
</tr>
<tr>
<td>00030</td>
<td>SRTNACC</td>
<td>EQU 42 Access module</td>
</tr>
<tr>
<td>00031</td>
<td>SRTNDB</td>
<td>EQU 43 Database</td>
</tr>
<tr>
<td>00032</td>
<td>SRTNDB</td>
<td>EQU 44 Last database type</td>
</tr>
<tr>
<td>00033</td>
<td>SRTNDB</td>
<td>EQU 45 Last database type</td>
</tr>
<tr>
<td>00034</td>
<td>SRTNDB</td>
<td>EQU 46 Last database type</td>
</tr>
<tr>
<td>00035</td>
<td>SRTNDB</td>
<td>EQU 47 Last database type</td>
</tr>
<tr>
<td>00036</td>
<td>SRTNDB</td>
<td>EQU 48 Last database type</td>
</tr>
<tr>
<td>00037</td>
<td>SRTNDB</td>
<td>EQU 49 Last database type</td>
</tr>
<tr>
<td>00038</td>
<td>SRTNDB</td>
<td>EQU 50 Last database type</td>
</tr>
<tr>
<td>00039</td>
<td>SRTNDB</td>
<td>EQU 51 Last database type</td>
</tr>
<tr>
<td>00040</td>
<td>SRTNDB</td>
<td>EQU 52 Last database type</td>
</tr>
<tr>
<td>00041</td>
<td>SRTNDB</td>
<td>EQU 53 Last database type</td>
</tr>
<tr>
<td>00042</td>
<td>SRTNDB</td>
<td>EQU 54 Last database type</td>
</tr>
<tr>
<td>00043</td>
<td>SRTNDB</td>
<td>EQU 55 Last database type</td>
</tr>
<tr>
<td>00044</td>
<td>SRTNDB</td>
<td>EQU 56 Last database type</td>
</tr>
<tr>
<td>00045</td>
<td>SRTNDB</td>
<td>EQU 57 Last database type</td>
</tr>
<tr>
<td>00046</td>
<td>SRTNDB</td>
<td>EQU 58 Last database type</td>
</tr>
<tr>
<td>00047</td>
<td>SRTNDB</td>
<td>EQU 59 Last database type</td>
</tr>
<tr>
<td>00048</td>
<td>SRTNDB</td>
<td>EQU 60 Last database type</td>
</tr>
<tr>
<td>00049</td>
<td>SRTNDB</td>
<td>EQU 61 Last database type</td>
</tr>
<tr>
<td>00050</td>
<td>SRTNDB</td>
<td>EQU 62 Last database type</td>
</tr>
<tr>
<td>00051</td>
<td>SRTNDB</td>
<td>EQU 63 Last database type</td>
</tr>
<tr>
<td>00052</td>
<td>SRTNDB</td>
<td>EQU 64 Last database type</td>
</tr>
<tr>
<td>00053</td>
<td>SRTNDB</td>
<td>EQU 65 Last database type</td>
</tr>
<tr>
<td>00054</td>
<td>SRTNDB</td>
<td>EQU 66 Last database type</td>
</tr>
<tr>
<td>00055</td>
<td>SRTNDB</td>
<td>EQU 67 Last database type</td>
</tr>
<tr>
<td>00056</td>
<td>SRTNDB</td>
<td>EQU 68 Last database type</td>
</tr>
<tr>
<td>00057</td>
<td>SRTNDB</td>
<td>EQU 69 Last database type</td>
</tr>
<tr>
<td>00058</td>
<td>SRTNDB</td>
<td>EQU 70 Last database type</td>
</tr>
<tr>
<td>00059</td>
<td>SRTNDB</td>
<td>EQU 71 Last database type</td>
</tr>
<tr>
<td>00060</td>
<td>SRTNDB</td>
<td>EQU 72 Last database type</td>
</tr>
<tr>
<td>00061</td>
<td>SRTNDB</td>
<td>EQU 73 Last database type</td>
</tr>
<tr>
<td>00062</td>
<td>SRTNDB</td>
<td>EQU 74 Last database type</td>
</tr>
<tr>
<td>00063</td>
<td>SRTNDB</td>
<td>EQU 75 Last database type</td>
</tr>
<tr>
<td>00064</td>
<td>SRTNDB</td>
<td>EQU 76 Last database type</td>
</tr>
<tr>
<td>00065</td>
<td>SRTNDB</td>
<td>EQU 77 Last database type</td>
</tr>
<tr>
<td>00066</td>
<td>SRTNDB</td>
<td>EQU 78 Last database type</td>
</tr>
<tr>
<td>00067</td>
<td>SRTNDB</td>
<td>EQU 79 Last database type</td>
</tr>
<tr>
<td>00068</td>
<td>SRTNDB</td>
<td>EQU 80 Last database type</td>
</tr>
<tr>
<td>00069</td>
<td>SRTNDB</td>
<td>EQU 81 Last database type</td>
</tr>
<tr>
<td>00070</td>
<td>SRTNDB</td>
<td>EQU 82 Last database type</td>
</tr>
<tr>
<td>00071</td>
<td>SRTNDB</td>
<td>EQU 83 Last database type</td>
</tr>
<tr>
<td>00072</td>
<td>SRTNDB</td>
<td>EQU 84 Last database type</td>
</tr>
<tr>
<td>00073</td>
<td>SRTNDB</td>
<td>EQU 85 Last database type</td>
</tr>
<tr>
<td>00074</td>
<td>SRTNDB</td>
<td>EQU 86 Last database type</td>
</tr>
<tr>
<td>00075</td>
<td>SRTNDB</td>
<td>EQU 87 Last database type</td>
</tr>
<tr>
<td>00076</td>
<td>SRTNDB</td>
<td>EQU 88 Last database type</td>
</tr>
<tr>
<td>00077</td>
<td>SRTNDB</td>
<td>EQU 89 Last database type</td>
</tr>
<tr>
<td>00078</td>
<td>SRTNDB</td>
<td>EQU 90 Last database type</td>
</tr>
</tbody>
</table>

These are SYSGEN definition restypes

Numbers 31-50 = Database resource types

Numbers 51-70 = User resource types

Numbers 71-90 = System resource types

*---------------------------------------------------------------------*
* ** AUTHORIZATION ID TYPE                                           *
*---------------------------------------------------------------------*
<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000E4</td>
<td>SATUSER</td>
<td>EQU C'U' USER</td>
</tr>
<tr>
<td>000C7</td>
<td>SATGRP</td>
<td>EQU C'G' GROUP</td>
</tr>
</tbody>
</table>

*---------------------------------------------------------------------*
* ** AUTHORIZATION ID FLAGS                                          *
*---------------------------------------------------------------------*
<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000C9</td>
<td>SAFINAC</td>
<td>EQU C'I' INACTIVE</td>
</tr>
<tr>
<td>000C1</td>
<td>SAFACTV</td>
<td>EQU C'A' ACTIVE</td>
</tr>
<tr>
<td>000C4</td>
<td>SAFDEL</td>
<td>EQU C'D' LOGICALLY DELETED</td>
</tr>
</tbody>
</table>

*---------------------------------------------------------------------*
* ** DEFAULT GROUP FLAG                                              *
*---------------------------------------------------------------------*
CA IDMS - 19.0

0040  SDGNDEFG  EQU  C' '  NOT A DEFAULT GROUP INDICATOR
00C4  SDGDEFG  EQU  C'D'  DEFAULT GROUP INDICATOR

* FUNCTION NUMBER ASSIGNMENT FLAG

00E4  SFNUSER  EQU  C'U'  FUNCTION NOS. ASSIGNED BY USER
00E2  SFNSYS   EQU  C'S'  FUNCTION NOS. ASSIGNED BY SYSTEM

* SECURITY METHOD

0009  SMEIDMS  EQU  C'I'  IDMS SECURITY
00C5  SMEEXT   EQU  C'E'  EXTERNAL SECURITY
000D  SMOFF    EQU  C'O'  NO SECURITY

* SECURITY MODE

00D6  SMOFF    EQU  C'O'  SECURITY MODE IS OFF
00E6  SMOWARN  EQU  C'W'  SECURITY MODE IS WARN
00C5  SMOENF   EQU  C'E'  SECURITY MODE IS ENFORCE

* MULTIPLE SIGNON

00F0  SMSNO    EQU  C'0'  MULTIPLE SIGNON NO
00F1  SMSYES   EQU  C'1'  MULTIPLE SIGNON YES

* WITH GRANT FLAG

000F  SWGOPT   EQU  X'01'  WITH GRANT OPTION

* GROUP/AUTH-ID LENGTH EQUATE

0012  SGAUIDL  EQU  18

* RESOURCE TYPE FLAG

0040  SLRLIB   EQU  C' '  LIBRARY RESOURCE
00D5  SLRLNLIB EQU  C'N'  NON-LIBRARY RESOURCE

* RUNNABLE ACCESS MODULE FLAG

008E  S RAMYES  EQU  C'Y'  ACCESS MODULE RUNNABLE
00D5  S RAMNO   EQU  C'N'  ACCESS MODULE NOT RUNNABLE

* GRANTABLE ACCESS MODULE FLAG

008E  SGAMYES  EQU  C'Y'  ACCESS MODULE GRANTABLE
00D5  SGAMNO   EQU  C'N'  ACCESS MODULE NOT GRANTABLE

* RESOURCEGROUP TYPE FLAG

00C3  S RGCAT   EQU  C'C'  CATEGORY
0040  SRGNCAT  EQU  C' '  NON-CATEGORY

* USERDATA TYPE FLAG

00C3  SUDTCAT  EQU  C'C'  CATEGORY
00D7  SUDTPRF  EQU  C'P'  PROFILE

* PROFILE TYPE FLAG

008E  SPTUSER  EQU  C'U'  USER PROFILE
00E2  SPTSYS   EQU  C'S'  SYSTEM PROFILE

* ATTRIBUTE OVERRIDE FLAG

008E  SAOYES   EQU  C'Y'  ATTRIBUTE MAY BE OVERRIDEN
00D5  SAONO    EQU  C'N'  ATTRIBUTE MAY NOT BE OVERRIDEN
<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>SRBFUNC DS XL1</td>
<td>Function code</td>
</tr>
<tr>
<td>000001</td>
<td>SRBSCREL DS XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINT #FLAG X'80'</td>
<td>Processing is INTERNAL</td>
</tr>
<tr>
<td>000002</td>
<td>SRBSCREL DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SRBPINTM DS 0XL1</td>
<td>SRB release level</td>
</tr>
<tr>
<td></td>
<td>SRBPINTM EQU X'80'</td>
<td></td>
</tr>
</tbody>
</table>
CA IDMS - 19.0

00000C  SRBXINST DS A  Pointer to feedback area
000060  SRBLNG2 EQU *-SECRB  Length of the fixed portion of SRB
000180  SRBLNG2F EQU ((SRBLNG2)+3)/4
000060  SRB0PP DS 0F  Start of SRB extensions
*
* SRB extension for SIGNON/SIGNOFF
*
*
000060  SRBSGSN DS A  Pointer to SIGNON element
000064  SRBSGLTF DS A  Address of signed on LTE (XFR SGNON)
00006C  SRBSGLTT DS A  Address of LTE signing on
00006D  SRBSGUSL DS XL1  Length of user ID
00006D  SRBSGUSR DS CL18  User ID
00007F  SRBSGGLR DS XL1  Length of group
000080  SRBSGGRP DS CL18  Group
000092  SRBSGPSL DS XL1  Length of new password
000093  SRBSGPSW DS CL8  password
00009C  SRBSGNPL DS CL8  New password
0000A4  SRBSGACC DS CL32  Accounting information
*
SRBSGPC #FLAG X'80'  Skip password checking
0000C4  SRBSGPCI DS 0XL1
000080  SRBSGPCM EQU X'80'
SRBSGPT #FLAG X'40'  Password already encrypted
0000C4  SRBSGPTI DS 0XL1
000040  SRBSGPTM EQU X'40'
SRBSGSM #FLAG X'20'  Suppress SIGNOFF message
0000C4  SRBSGSMI DS 0XL1
000020  SRBSGSMM EQU X'20'
SRBSGPC #FLAG X'10'  Copy SIGNON from specified LTE
0000C4  SRBSGCPI DS 0XL1
000010  SRBSGCPM EQU X'10'
SRBSGJU #FLAG X'08'  Signon with user-id from jobcard
0000C4  SRBSGJUI DS 0XL1
000008  SRBSGJM EQU X'08'
SRBSGFG1 DS XL1  FLAG 1
*
* The following fields are returned by SIGNON
*
*
0000C5  SRBSGSPR DS CL18  System profile module name
0000D7  SRBSGUPR DS CL18  User profile module name
0000E9  SRBSGLEN EQU *-SECRB  Length of SRB for SIGNON/SIGNOFF
0003B  SRBSGLNF EQU (SRBSGLEN+3)/4
0000E9  ORG SRB0PP  RESET
*
* SRB extension for password encryption
*
*
000060  SRBRCUSL DS XL1  User ID length
000061  SRBRCUSR DS CL18  User ID
000073  SRBRCPSL DS XL1  Plaintext password length
000074  SRBRCPSW DS CL8  Plaintext password
00007C  SRBRCRPS DS CL8  Encrypted password
000084  SRBRCLEN EQU *-SECRB  Length of SRB for pswd encryption
000210  SRBRCLENF EQU (SRBRCLEN+3)/4
000084  ORG SRB0PP  RESET
*Offset  Value  * SRB extension for security check, bulk
check and AM check
*
*
000060  SRBCSSON DS A  Pointer to SIGNON element
SRBCSCL #FLAG X'80'  LOG=NO requested
000064  SRBCSCLN1 DS 0XL1
000080  SRBCSCLN M EQU X'80'
000064  SRBCSCLG DS X  Option flags
000065  DS XL3  Reserved
00006B  SRBRSSTTA DS A  Pointer to SRTT entry

10-Jan-2018  347/367
### SECRLST

**COPY #SECRLST**

*------------------------------------------------------------------*
| **SECURITY RESOURCE LIST**                                      |
| **------------------------------------------------------------------** |
| **Resource list entry. Note that all resources in the list must** |
| **be of the same class (as specified on the #SECHECK) and, for** |
| **internal checking, must be defined in the same dictionary or** |
| **catalog.**                                                     |
| **------------------------------------------------------------------** |

**COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.**

---

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>SRLSCLNL DS XL1</td>
<td>Length of resource name</td>
</tr>
<tr>
<td>000001</td>
<td>SRLSCLRM DS CL60</td>
<td>Resource name</td>
</tr>
<tr>
<td>000030</td>
<td>SRLSCLLB DS XL1</td>
<td>Length of library name</td>
</tr>
<tr>
<td>00003E</td>
<td>SRLSCLIL DS CL8</td>
<td>Library name</td>
</tr>
<tr>
<td>000046</td>
<td>SRLSRLAUTH DS XL6</td>
<td>Requested authorities</td>
</tr>
<tr>
<td>0004C</td>
<td>SRLSRLEN EQU *-SECRLST</td>
<td>Length of resource id</td>
</tr>
<tr>
<td>0004E</td>
<td>SRLRSLNG EQU *-SECRLST</td>
<td>Total length of entry</td>
</tr>
</tbody>
</table>

---

### SECRTTD

**COPY #SECRTTD**

* * The Resource Type Table (SRTT) is used to select external vs internal processing for resource types and to provide external resource classes for those resource types that are secured externally. *

---

10-Jan-2018
<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>SECRTHD DSECT 06/19/96</td>
</tr>
<tr>
<td>00000</td>
<td>SRTGRPLA DS A Address of group list</td>
</tr>
<tr>
<td>00004</td>
<td>SRTSGRTN DS 0F SIGNON retention (before init)</td>
</tr>
<tr>
<td>00004</td>
<td>SRTSONTA DS A Address of SON table (after init)</td>
</tr>
<tr>
<td>00008</td>
<td>SRTCBBMPA DS A Address of category bitmap pool</td>
</tr>
<tr>
<td>00008</td>
<td>SRTABMPA DS A Address of activity bitmap pool</td>
</tr>
<tr>
<td>00010</td>
<td>SRTSCC DS 0CL4 Startup completion code</td>
</tr>
<tr>
<td>00010</td>
<td>SRTSR15 DS CL2 Return code (R15)</td>
</tr>
<tr>
<td>00012</td>
<td>SRTSR0 DS CL2 Reason code (R0) if applicable</td>
</tr>
<tr>
<td>00014</td>
<td>SRTAPP1L DS XL1 Length of system name</td>
</tr>
<tr>
<td>00015</td>
<td>SRTAPPID DS CL8 System name</td>
</tr>
<tr>
<td>0001D</td>
<td>SRTPENVNL DS XL1 Length of environment name</td>
</tr>
<tr>
<td>0001E</td>
<td>SRTPENVNM DS CL8 Environment name</td>
</tr>
<tr>
<td>00026</td>
<td>SRTSVCNO DS XL1 SVC number for local security</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOINT #FLAG X'01' Internally secured resources</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOINTM EQU X'01'</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOINTI DS 0XL1</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOEXT #FLAG X'02' Externally secured resources</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOEXTM EQU X'02'</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOEXTI DS 0XL1</td>
</tr>
<tr>
<td>00027</td>
<td>SRTDFSN #FLAG X'04' Default SIGNON active</td>
</tr>
<tr>
<td>00027</td>
<td>SRTDFSNI DS 0XL1</td>
</tr>
<tr>
<td>00027</td>
<td>SRTDFSM #FLAG X'04'</td>
</tr>
<tr>
<td>00027</td>
<td>SRTDFSMI DS 0XL1</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOMXD #FLAG SRTOINTM+SRTOEXTM Mixed security if both on</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOMXDI DS 0XL1</td>
</tr>
<tr>
<td>0003</td>
<td>SRTOMXDM EQU SRTOINTM+SRTOEXTM</td>
</tr>
<tr>
<td>00027</td>
<td>SRTESP #FLAG X'10' Process system profiles for ERUs</td>
</tr>
<tr>
<td>00027</td>
<td>SRTESPNI DS 0XL1</td>
</tr>
<tr>
<td>00010</td>
<td>SRTESPM #FLAG X'10' Process user profiles for ERUs</td>
</tr>
<tr>
<td>00027</td>
<td>SRTESPMI DS 0XL1</td>
</tr>
<tr>
<td>00027</td>
<td>SRTEUP #FLAG X'20'</td>
</tr>
<tr>
<td>00027</td>
<td>SRTEUPI DS 0XL1</td>
</tr>
<tr>
<td>00020</td>
<td>SRTEUPE #FLAG X'20'</td>
</tr>
<tr>
<td>00027</td>
<td>SRTEUPEM DS 0XL1</td>
</tr>
<tr>
<td>00027</td>
<td>SRTUCA #FLAG X'40' USER catalog available</td>
</tr>
<tr>
<td>00027</td>
<td>SRTUCAI DS 0XL1</td>
</tr>
<tr>
<td>00040</td>
<td>SRTUCAM #FLAG X'40' SYSTEM catalog available</td>
</tr>
<tr>
<td>00027</td>
<td>SRTSCAI DS 0XL1</td>
</tr>
<tr>
<td>00080</td>
<td>SRTSCAM EQU X'80' Option flags</td>
</tr>
<tr>
<td>00027</td>
<td>SRTOPTNS DS XL1</td>
</tr>
<tr>
<td>00028</td>
<td>SRTTBH DS XL(TBHDLSLEN) Standard table header (see #TBHDS)</td>
</tr>
<tr>
<td>00034</td>
<td>SRTTBHSN DS XL(TBHDLSLEN) Standard table header (see #TBHDS)</td>
</tr>
<tr>
<td>00040</td>
<td>SRTTLTNT DS F Total length of table</td>
</tr>
<tr>
<td>00044</td>
<td>SRTTTLNDS DS F Total entries</td>
</tr>
<tr>
<td>00048</td>
<td>SRTUSRSTK DS H Keyword token</td>
</tr>
<tr>
<td>00048</td>
<td>SRTUSPRF DS CL18 Default USER profile</td>
</tr>
<tr>
<td>0005C</td>
<td>SRTSYSSTK DS H Keyword token</td>
</tr>
<tr>
<td>0005E</td>
<td>SRTSYSPRF DS CL18 Default SYSTEM profile</td>
</tr>
<tr>
<td>00070</td>
<td>SRTRRTKT DS H Keyword token</td>
</tr>
<tr>
<td>00072</td>
<td>SRTERURF DS CL18 Default USER profile for ERUs</td>
</tr>
<tr>
<td>00084</td>
<td>SRTERSTK DS H Keyword token</td>
</tr>
<tr>
<td>00086</td>
<td>SRTERSRF DS CL18 Default SYSTEM profile for ERUs</td>
</tr>
<tr>
<td>00098</td>
<td>SRTVERS DS CL18 SRTT version indicator</td>
</tr>
<tr>
<td>00000</td>
<td>SRTV12 EQU X'00' Up to release 12.01</td>
</tr>
<tr>
<td>00001</td>
<td>SRTV14 EQU X'01' Release 14.0 and above</td>
</tr>
<tr>
<td>00099</td>
<td>SRTDUST #FLAG X'01' Default userid in SRTDUID field R140</td>
</tr>
<tr>
<td>0001</td>
<td>SRTDUSTM EQU X'01'</td>
</tr>
</tbody>
</table>

---

10-Jan-2018 349/367
**SPRFDS**

```
COPY #SPRFDS
******************************************************************************
***                     SPRF DSECT                                      ***
***                      PROFILE                                        ***
***                      COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.***
******************************************************************************

Offset   Value   SPRF     DSECT
03/05/90  000000   SPRFNAME DS CL18     PROFILE NAME
000012   000000   SPRFTYPE DS CL1      TYPE
          *       * 'U' - USER PROFILE
          *       * 'S' - SYSTEM PROFILE
000013   000013   SPRFCTIM DS CL8      TIME/DATA PROFILE WAS CREATED
000018   000018   SPRFUTIM DS CL8      TIME/DATA PROFILE WAS LAST UPDATED
000023   000023   SPRFCAID DS CL18     CREATOR OF THE PROFILE
000035   000035   SPRFUAIMD DS CL18     LAST UPDATOR OF THE PROFILE
000047   000047                        * UNUSED
00058   00058   SPRFDSLN EQU   *-SPRF     * LENGTH OF DSECT
******************************************************************************
```

**SRESDS**

```
COPY #SRESDS
******************************************************************************
***                     SRES DSECT                                      ***
***                       RESOURCE                                     ***
***                       COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.***
******************************************************************************

Offset   Value   SRES     DSECT
000000   000000   SRESRTYP DS CL4      RESOURCE TYPE
000004   000004   SRESRNAME DS CL60     RESOURCE NAME
000040   000040   SRESGTNL DS XL1       Null column indicator
000041   000041   SRESGTYT DP CL4       RESOURCE GROUP TYPE
000045   000045   SRESGNNL DS XL1       Null column indicator
000046   000046   SRESGNAM DS CL60     RESOURCE GROUP NAME
000082   000082   SRESCFNO DS H         CATEGORY OR ACTIVITY NUMBER
          *       * CATEGORY NO IF SRESRTYP='CATA'
          *       * ACTIVITY NO IF SRESRTYP='ACTI'
000084   000084   SRESAMRN DS CL1      RUNNABLE ACCESS MODULE FLAG
          *       * 'N' IF NOT RUNNABLE
          *       * 'Y' IF RUNNABLE
000085   000085   SRESSTAT DS CL1      STATUS FLAG
          *       * 'A' FOR ACTIVE
          *       * 'D' FOR LOGICALLY DELETED
000086   000086   SRESAMGT DS CL1      GRANTABLE ACCESS MODULE FLAG
          *       * 'N' IF NOT GRANTABLE
          *       * 'Y' IF GRANTABLE
000087   000087   SRESAMLC DS CL8      TIME/DATA OF LAST AM-RELATED CHANGE
00008F   00008F   SRESCTIM DS CL8      TIME/DATA RESOURCE WAS CREATED
000097   000097   SRESUTIM DS CL8      TIME/DATA RESOURCE WAS LAST UPDATED
00009F   00009F   SRESCAID DS CL18     CREATOR OF THE RESOURCE
******************************************************************************
```
SRGADS

COPY #SRGADS

*** RESOURCE GROUP AUTHORITY ***

COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.

COPY

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.

SRGADS

COPY #SRGADS

RESERVED.
SRGPDS

COPY #SRGPDS

***************************************************
*** SRGP DSECT ***
*** RESOURCE GROUP ***
*** COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED. ***
***************************************************

Offset Value 000000  SRGP  DSECT
10/29/90                                            
000000 SRGPRRTYP DS  CL4  RESOURCE TYPE
000004 SRGPRNAM DS  CL60  RESOURCE NAME
000040 SRGPOWNR DS  CL18  AUTH. ID OF THE RES. GROUP OWNER
000052 SRGPCATN DS  H  CATEGORY NUMBER
000054 SRGPFASG DS  CL1  FUNCTION NUMBER ASSIGNMENT FLAG
*        * 'S' FOR ASSIGNED BY SYSTEM
*        * 'U' FOR ASSIGNED BY USER
000055 SRGPCCTIM DS  CL8  TIME/DATE AUTH. ID WAS CREATED
00005D SRGPUTIM DS  CL8  TIME/DATE AUTH. ID WAS LAST UPDATED
000077 SRGPUAID DS  CL18  LAST UPDATOR OF THE RESOURCE GROUP
000089 DS  XL19  * UNUSED
0009C SRGPDSLN EQU -SRGP  * LENGTH OF DSECT

***************************************************

SROPDS

COPY #SROPDS

***************************************************
*** SROP DSECT ***
*** RESOURCE OPTION ***
*** COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED. ***
***************************************************

Offset Value 000000  SROP  DSECT
10/29/90                                            
000000 SROPRTYP DS  CL4  RESOURCE TYPE
000004 SROPRTYP DS  CL40  RESOURCE NAME
000040 SROPRTYP DS  CL1  SECURITY METHOD
*  * 'I' FOR IDMS
*  * 'E' FOR EXTERNAL
*  * 'O' FOR OFF
000041 DS  XL11  * unused
000042 SROPMULT DS  CL1  MULTIPLE SIGNON FLAG
*  * '0' FOR MULTIPLE SIGNON NOT ALLOWED
SRSADS

COPY #SRSADS

******************************************************************************

** **
** **
** SRS DSECT
** **
** RESOURCE AUTHORITY
** **
** **
** COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.
**
**
**
******************************************************************************

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>SRSA</th>
<th>DSECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>SRSAID DS CL18</td>
<td>AUTH. ID THE AUTHORITY IS GRANTED TO</td>
<td></td>
</tr>
<tr>
<td>000012</td>
<td>SRSARTYP DS CL4</td>
<td>RESOURCE TYPE THE AUTHORITY IS GRANTED ON</td>
<td></td>
</tr>
<tr>
<td>000016</td>
<td>SRSARNAM DS CL60</td>
<td>RESOURCE NAME THE AUTHORITY IS GRANTED ON</td>
<td></td>
</tr>
<tr>
<td>000052</td>
<td>SRSARA DS H</td>
<td>RUNTIME AUTHORITIES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR SELECT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR EXECUTE/RUN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>2 FOR INSERT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>4 FOR UPDATE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>8 FOR DELETE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>128 FOR OWNER</td>
<td></td>
</tr>
<tr>
<td>000054</td>
<td>SRSARAWG DS H</td>
<td>RUNTIME AUTHORITIES (WITH GRANT OPTION)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR SELECT WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR EXECUTE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>2 FOR INSERT WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>4 FOR UPDATE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>8 FOR DELETE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>128 FOR OWNER WITH GRANT</td>
<td></td>
</tr>
<tr>
<td>000056</td>
<td>SRSADA DS H</td>
<td>DEFINITION AUTHORITIES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR CREATE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>2 FOR ALTER</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>4 FOR DROP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>8 FOR DISPLAY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>16 FOR USE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>31 FOR DEFINE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>32 FOR REFERENCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>128 FOR OWNER</td>
<td></td>
</tr>
<tr>
<td>000058</td>
<td>SRSADAWG DS H</td>
<td>DEFINITION AUTHORITIES (WITH GRANT OPTION)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR CREATE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>2 FOR ALTER WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>4 FOR DROP WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>8 FOR DISPLAY WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>16 FOR USE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>31 FOR DEFINE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>32 FOR REFERENCE WITH GRANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>128 FOR OWNER WITH GRANT</td>
<td></td>
</tr>
<tr>
<td>00005A</td>
<td>SRSAOA DS H</td>
<td>OTHER AUTHORITIES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>1 FOR SYSADMIN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>2 FOR DDAADMIN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>4 FOR DBADMIN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>8 FOR DCAADMIN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>16 FOR SIGNON</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>32 FOR DBREAD</td>
<td></td>
</tr>
</tbody>
</table>
CA IDMS - 19.0

* 64 FOR DBAWRITE

00005C SRSACTIM DS CL8 TIME/DATETIME RESOURCE AUTH. WAS CREATED
000064 SRSAUTIM DS CL8 TIME/DATETIME RESOURCE AUTH. LAST UPDATED
00006C SRSACAID DS CL18 CREATOR OF THE RESOURCE AUTHORITY
00007E SRSUAID DS CL18 LAST UPDATOR OF THE RESOURCE AUTHORITY

* UNUSED

000090 SRSADSLN EQU *-SRSA * LENGTH OF DSECT

SUGPDS

COPY #SUGPDS

******************************************************************************

COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.

******************************************************************************

Offset Value SUGP DSECT

05/16/89 SUGPGPID DS CL18 AUTH. ID OF THE PARENT GROUP
000012 SUGPUSID DS CL18 AUTH. ID OF THE CHILD USER
000024 SUGPOEFG DS CL1 DEFAULT GROUP INDICATOR
* *' ' FOR NOT THE DEFAULT GROUP
* *'D' FOR THE DEFAULT GROUP
000025 SUGPCTIM DS CL8 TIME/DATETIME GROUP WAS CREATED
00002D SUGPUTIM DS CL8 TIME/DATETIME GROUP WAS LAST UPDATED
000035 SUGPCAID DS CL18 CREATOR OF THE GROUP
000047 SUGPUAID DS CL18 LAST UPDATOR OF THE GROUP
000059 DS XL11 * UNUSED

00064 SUGPDSLN EQU *-SUGP * LENGTH OF DSECT

******************************************************************************

SUSDDS

COPY #SUSDDS

******************************************************************************

COPYRIGHT (C) 2007 CA technologies. ALL RIGHTS RESERVED.

******************************************************************************

Offset Value SUSD DSECT

10/29/90 SUSDAUID DS CL18 AUTHORIZATION ID
000012 SUSDCLTH DS H CATEGORY BIT MAP LENGTH
000014 SUSDTYPE DS CL1 TYPE OF USER DATA
* *'C' - CATEGORY
* *'P' - PROFILE
* PROFILE DATA
000015 SUSDSYST DS CL8 SYSTEM NAME
00001D SUSDPROF DS CL18 PROFILE NAME

******************************************************************************

10-Jan-2018 355/367
Privileges Required for Statements

For more information, see the following topics:

- DDL Statements (see page 357)
- SQL Statements (see page 358)
DDL Statements

DISPLAY Statements

To execute a DISPLAY statement on a physical database entity requires one of these privileges if the dictionary and the resource type of the entity has been secured:

- DISPLAY privilege on the entity
- DBADMIN privilege on the dictionary where the entity is defined

This table shows what privilege is required to execute a DISPLAY statement when the indicated resource type has been secured:

<table>
<thead>
<tr>
<th>Statement</th>
<th>Privilege required</th>
<th>Resource type</th>
<th>Resource name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPLAY SEGMENT</td>
<td>DISPLAY</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>DISPLAY FILE</td>
<td>DISPLAY</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>DISPLAY AREA</td>
<td>DISPLAY</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>DISPLAY DMCL</td>
<td>DISPLAY</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>DISPLAY[JOURNAL] BUFFER</td>
<td>DISPLAY</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>DISPLAY ARCHIVE/DISK/TAPE JOURNAL</td>
<td>DISPLAY</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>DISPLAY DBTABLE</td>
<td>DISPLAY</td>
<td>DBTB</td>
<td>dbtable-name</td>
</tr>
<tr>
<td>DISPLAY DBNAME</td>
<td>DISPLAY</td>
<td>DBTB</td>
<td>dbtable-name</td>
</tr>
</tbody>
</table>

Definition Statements

To execute the physical database definition statements in the following table requires:

- DBADMIN privilege on the dictionary where the physical database definition is stored, if that dictionary has been secured.
- The privileges indicated in the following table, if the resource types have been secured.

The privileges must be granted in a session that is connected to the system dictionary.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Privilege required</th>
<th>Resource type</th>
<th>Resource name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE SEGMENT</td>
<td>CREATE</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>ALTER SEGMENT</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>DROP SEGMENT</td>
<td>DROP</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>CREATE FILE</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>ALTER FILE</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>DROP FILE</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>CREATE AREA</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>Statement</td>
<td>Privilege required</td>
<td>Resource type</td>
<td>Resource name</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>ALTER AREA</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>DROP AREA</td>
<td>ALTER</td>
<td>DB</td>
<td>segment-name</td>
</tr>
<tr>
<td>CREATE DMCL</td>
<td>CREATE</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>ALTER DMCL...DBTABLE name</td>
<td>ALTER USE</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>DROP DMCL</td>
<td>DROP</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>GENERATE DMCL</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>CREATE [JOURNAL] BUFFER</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>ALTER [JOURNAL] BUFFER</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>DROP [JOURNAL] BUFFER</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>CREATE ARCHIVE/DISK/TAPE JOURNAL</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>ALTER ARCHIVE/DISK/TAPE JOURNAL</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>DROP ARCHIVE/DISK/TAPE JOURNAL</td>
<td>ALTER</td>
<td>DMCL</td>
<td>dmcl-name</td>
</tr>
<tr>
<td>CREATE DBTABLE</td>
<td>CREATE</td>
<td>DBTB</td>
<td>dbtable-name</td>
</tr>
<tr>
<td>ALTER DBTABLE</td>
<td>ALTER</td>
<td>DBTB</td>
<td>dbtable-name</td>
</tr>
<tr>
<td>DROP DBTABLE</td>
<td>DROP</td>
<td>DBTB</td>
<td>dbtable-name</td>
</tr>
<tr>
<td>GENERATE DBTABLE</td>
<td>ALTER</td>
<td>DBTB</td>
<td>dbtable-name</td>
</tr>
<tr>
<td>CREATE DBNAME</td>
<td>CREATE ALTER</td>
<td>DB</td>
<td>database-name</td>
</tr>
<tr>
<td>ALTER DBNAME</td>
<td>ALTER</td>
<td>DB</td>
<td>database-name</td>
</tr>
<tr>
<td>DROP DBNAME</td>
<td>DROP ALTER</td>
<td>DB</td>
<td>database-name</td>
</tr>
</tbody>
</table>

## SQL Statements

**Contents**

- SQL DDL statements (see page 359)
- SQL DML Statements (see page 360)
- Access Module Management Statements (see page 361)

**Statement Categories**

If the database resource has been secured, users require appropriate privileges to execute SQL statements in these three categories:

- SQL DDL
- SQL DML
- Access module management

**SQL Session Authorization**

A user does not have the authority to issue a CONNECT statement to a dictionary under the central version unless the user holds signon privilege for the system in which the dictionary is defined.

### SQL DDL statements

To execute CA IDMS SQL DDL statements, a user requires the privilege indicated in the following table if the resource type to which the privilege applies has been secured.

**Note:** Resource types QSCH, TABL, and AREA are secured when the DB resource type is secured.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Privilege required</th>
<th>Resource type</th>
<th>Resource name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE CALC</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td>DROP CALC</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td>CREATE CONSTRAINT</td>
<td>ALTER(2) REFERENCES</td>
<td>TABL</td>
<td>referencing-table-identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TABL</td>
<td>referenced-table-identifier</td>
</tr>
<tr>
<td>DROP CONSTRAINT</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>referencing-table-identifier</td>
</tr>
<tr>
<td>CREATE FUNCTION</td>
<td>CREATE(2)</td>
<td>TABL</td>
<td>function-identifier</td>
</tr>
<tr>
<td>ALTER FUNCTION</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>function-identifier</td>
</tr>
<tr>
<td>DROP FUNCTION</td>
<td>DROP(2)</td>
<td>TABL</td>
<td>function-identifier</td>
</tr>
<tr>
<td>CREATE INDEX</td>
<td>ALTER(2) USE</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AREA</td>
<td>segment-name.area-name</td>
</tr>
<tr>
<td>ALTER INDEX</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td>DROP INDEX</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td>CREATE PROCEDURE</td>
<td>CREATE(2)</td>
<td>TABL</td>
<td>procedure-identifier</td>
</tr>
<tr>
<td>ALTER PROCEDURE</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>procedure-identifier</td>
</tr>
<tr>
<td>DROP PROCEDURE</td>
<td>DROP(2)</td>
<td>TABL</td>
<td>procedure-identifier</td>
</tr>
<tr>
<td>CREATE SCHEMA(1)</td>
<td>CREATE</td>
<td>QSCH</td>
<td>scheme-name</td>
</tr>
<tr>
<td>ALTER SCHEMA(1)</td>
<td>ALTER</td>
<td>QSCH</td>
<td>scheme-name</td>
</tr>
<tr>
<td>DROP SCHEMA [CASCADE]</td>
<td>DROP</td>
<td>QSCH</td>
<td>scheme-name</td>
</tr>
<tr>
<td>CREATE TABLE</td>
<td>CREATE(2) USE</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AREA</td>
<td>segment-name.area-name</td>
</tr>
<tr>
<td>ALTER TABLE</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td>Statement</td>
<td>Privilege required</td>
<td>Resource type</td>
<td>Resource name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------------</td>
<td>---------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>DROP TABLE [CASCADE]</td>
<td>DROP(2)</td>
<td>TABL</td>
<td>table-identifier</td>
</tr>
<tr>
<td>CREATE TABLE PROCEDURE</td>
<td>CREATE(2)</td>
<td>TABL</td>
<td>table-procedure-identifier</td>
</tr>
<tr>
<td>ALTER TABLE PROCEDURE</td>
<td>ALTER(2)</td>
<td>TABL</td>
<td>table-procedure-identifier</td>
</tr>
<tr>
<td>DROP TABLE PROCEDURE</td>
<td>DROP(2)</td>
<td>TABL</td>
<td>drop-procedure-identifier</td>
</tr>
<tr>
<td>CREATE VIEW</td>
<td>CREATE(2)</td>
<td>TABL</td>
<td>view-identifier</td>
</tr>
<tr>
<td>DROP VIEW [CASCADE]</td>
<td>DROP(2)</td>
<td>TABL</td>
<td>view-identifier</td>
</tr>
</tbody>
</table>

(1) If reference to a non-SQL-defined schema is made in the CREATE/ALTER SCHEMA statement, then the user must also hold either USE on NONSQL SCHEMA Vnnnn.schema-name or DBADMIN on the dictionary where the non-SQL-defined schema is stored. If DBNAME is specified, the user must hold USE on the named database; if not specified, the user must hold DBADMIN on the system dictionary.

(2) The owner of the associated schema implicitly holds the privilege.

SQL DML Statements

Dynamic SQL Statements

To execute an SQL DML statement dynamically (for example, through the Command Facility), a user requires the access privilege indicated in the following table or must own schema associated with the table-like object, if the database has been secured.

If a DELETE, UPDATE, or INSERT statement contains a query expression or subquery, additional privileges are required for the table-like objects referred to by the query expression or subquery. If a SELECT statement explicitly names a view, additional privileges are required.

Note: For complete documentation of privileges required to execute SQL DML statements, see the CA IDMS SQL Reference section.

Embedded SQL Statements

If the SQL DML statement is embedded in an application program and the database accessed is secured externally, the executing user must hold the applicable privileges on all tables accessed by the statement, whether directly or indirectly through a view.
If the SQL DML statement is embedded in an application program and the database accessed is secured internally, the executing user requires only EXECUTE privilege on the access module; the user inherits the necessary access privileges from the grantor of EXECUTE privilege for the purpose of executing the access module.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Privilege required</th>
<th>Resource type</th>
<th>Resource name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE</td>
<td>DELETE</td>
<td>TABL</td>
<td>schema-name.table-name</td>
</tr>
<tr>
<td>INSERT</td>
<td>INSERT</td>
<td>TABL</td>
<td>schema-name.table-name</td>
</tr>
<tr>
<td>SELECT</td>
<td>SELECT</td>
<td>TABL</td>
<td>schema-name.table-name</td>
</tr>
<tr>
<td>UPDATE</td>
<td>UPDATE</td>
<td>TABL</td>
<td>schema-name.table-name</td>
</tr>
</tbody>
</table>

**Access Module Management Statements**

To execute access module management statements a user requires the privilege indicated in the following table or must own the schema associated with the access module, if the database has been secured.

If the owner lacks one or more applicable table access privileges for SQL statements in the RCMs included in the access module when it is created or altered, a warning is issued. If privileges are lacking at runtime, an error occurs when the user attempts to execute the access module.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Privilege required</th>
<th>Resource type</th>
<th>Resource name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE ACCESS MODULE</td>
<td>CREATE</td>
<td>DACC</td>
<td>schema-name.access-module-name</td>
</tr>
<tr>
<td>ALTER ACCESS MODULE</td>
<td>ALTER</td>
<td>DACC</td>
<td>schema-name.access-module-name</td>
</tr>
<tr>
<td>DROP ACCESS MODULE</td>
<td>DROP</td>
<td>DACC</td>
<td>schema-name.access-module-name</td>
</tr>
<tr>
<td>EXPLAIN ACCESS MODULE</td>
<td>DISPLAY</td>
<td>DACC</td>
<td>schema-name.access-module-name</td>
</tr>
</tbody>
</table>

**User-Defined System Security Rules**

This section provides the information on the following topics:

- Installation Codes (see page 362)
- Using Terminal Autotasks (see page 362)
- Associating an Autotask with Selected Logical Terminals (see page 363)
- Signon and Signoff Functions for an Autotask (see page 364)
- Associating Terminals with Devices (see page 365)
- Checking Authority to Access a Particular Terminal (see page 366)
- Design Considerations (see page 366)
Installation Codes

You can assign an installation code attribute value to the INSTCODE attribute keyword in a user or system profile. If you specify OVERRIDE=NO for INSTCODE, you can use the attribute for site-specific security checking. At runtime, an application program or exit 29 can retrieve the INSTCODE attribute value for the user session by linking to RHDCUF00 and issuing the DCUF SHOW INSTCODE command.

⚠️ Note: For more information on linking to DCUF, see the CA IDMS System Tasks and Operator Reference section.

INSTCODE Considerations

In the logic to retrieve an INSTCODE attribute value, you should account for these possibilities:

- INSTCODE may not be an attribute of a given user session.
- An attribute value may be as many as 32 characters.

Using Terminal Autotasks

What is an Autotask

A terminal autotask is a task that you associate with a logical terminal. Through this association, you direct the CA IDMS system to initiate the preassigned task when either of these conditions occurs:

- The ENTER NEXT TASK CODE prompt would normally be displayed.
- A top-level program or dialog issues a DC RETURN request that specifies no next task code.

Suppose, for example, that a user who signs on to a CA IDMS system is assigned logical terminal LT12012. If this logical terminal is associated with an autotask, the system automatically initiates the autotask when the user connects to the terminal. After the user enters the name of the system, instead of displaying the ENTER NEXT TASK CODE prompt (which allows the user to choose what task to invoke), the terminal displays whatever screen is mapped out by the program associated with the preassigned autotask.

Securing a Terminal Autotask

Since terminal autotasks must be able to execute without a signed on user, they must be unsecured. If tasks are secured internally, you can unsecure terminal autotasks by assigning them to a category on which PUBLIC holds EXECUTE privilege. Alternatively, you can add an occurrence override to turn off security for each terminal autotask in the SRTT.
If tasks are secured externally, you must use an occurrence override in the SRTT to unsecure a
terminal autotask.

**When Should You Use an Autotask**

Use an autotask for workers at one terminal who require access to a *limited number of data
processing functions*.

For example, you can use an autotask for clerks who enter orders into a purchasing system. Or you
can set up an autotask that executes a single manufacturing application. The application itself can
perform a variety of functions, such as bill of material processing and master production scheduling.

In both of these cases, you create a menu for display when the autotask is invoked. This menu lists
each option the user can choose, including a signon and signoff option if appropriate.

You can also use an autotask to invoke a *site-defined signon menu*. Set up this menu to appear when
users connect to the terminal. After signing on to this menu, users invoke any task for which they are
authorized.

**How to Use an Autotask**

To use an autotask, you must do the following:

- Associate an autotask with selected logical terminals.
- Optionally, establish signon and signoff functions as part of the autotask menu.
- Optionally, associate physical terminals with particular devices.
- Optionally, check a user's authority to use a particular terminal.

These topics along with the related design suggestions are discussed as follows.

**Associating an Autotask with Selected Logical Terminals**

**Associating an Autotask with a Logical Terminal**

You associate an autotask with a logical terminal using the AUTOTASK parameter of the system
generation LTERM statement. This statement allows you to specify whether a task will be initiated
automatically for the logical terminal:

- **AUTOTASK IS NULL**, the default, indicates that no task will be initiated automatically for this
  logical terminal. Logical terminals defined as printers *must* use the default value NULL.

- **AUTOTASK IS task-code** specifies the system will execute the specified task code automatically.
The task code specified must be defined in the system dictionary with the system generation
TASK statement. The task must be defined with the NOINPUT option because a terminal autotask
should execute immediately.
Example

To associate LTERM LT12012 with the task code ORDERS, submit this information to the system generation compiler:

```
ADD LTERM LT12012 VERSION 1
   AUTOTASK IS ORDERS
   ENABLED
   PRIORITY IS 0
   PTERM IS PT12012.
```

Signon and Signoff Functions for an Autotask

Forcing Signon Through a Terminal Autotask

If you are using a site-specific signon menu to force a signon and you want users to see the ENTER NEXT TASK CODE prompt after they sign on to the CA IDMS system:

- Clear the autotask field (LTEAUTSK) in the logical terminal element (LTE) when the user signs on to the runtime system.
- Reset the autotask field when the user signs off from the runtime system.

⚠️ **Note:** For more information on the LTE (#LTEDS DSECT), see the *CA IDMS DSECT Reference section*.

In a CA Technologies ADS Environment

If your autotask invokes a CA technologies ADS application, you can specify the following:

- Automatic signon for the application, by using a CA technologies ADS signon menu
- Automatic signoff for the application, by using the CA technologies ADS SIGNOFF function

⚠️ **Note:** For more information on CA technologies ADS signon menus and the CA technologies ADS SIGNOFF function, see the *CA IDMS ADS Reference section*.

In an SQL or DML Environment

If your autotask invokes an SQL or DML application, design the application as follows:

1. When the user signs on through the autotask menu, link to the RHDCSNON program from the autotask program.

⚠️
CA IDMS - 19.0

Note: For more information, see the "SIGNON task" in the CA IDMS System Tasks and Operator Reference section.

Alternatively, if no signon CLIST is invoked, an Assembler program can issue the #SECSGON macro to initiate user signon.

Note: For more information, see #SECSGON (see page 173).

2. When the user signs off through the autotask menu, link to either RHDCSNOF or RHDCBYE:

- RHDCSNOF is the program normally invoked by the CA IDMS system SIGNOFF task (RHDCSNOF leaves the ENTER NEXT TASK CODE prompt displayed on the terminal).

- RHDCBYE is the program normally invoked by the CA IDMS system BYE task (RHDCBYE does not leave the ENTER NEXT TASK CODE prompt displayed on the terminal).

Alternatively, if you do not wish to free resources as RHDCBYE and RHDCSON do, an Assembler program can issue the #SECSGOF macro to initiate user signoff.

Note: For more information, see #SECSGOF (see page 170).

How to Clear and Reset the Autotask Field

To display the ENTER NEXT TASK CODE prompt after the signon menu, you clear the autotask field. You can reset the autotask field at signoff. You can do this by using exit 29 to move the appropriate value to the autotask field (LTEAUTSK) in the logical terminal element (LTE) after a successful signon or signoff. After signon, you move low values (binary zeros) to this field; after signoff, you move the appropriate task code. Note: For more information on the LTE (#LTEDS DSECT), see the CA IDMS DSECT Reference section.

Associating Terminals with Devices

Overview

Once you associate an autotask with selected logical terminals, you need to ensure that these terminals correspond to particular devices. You do this by associating logical terminal/physical terminal pairs with the appropriate devices.

UCF, VTAM, and TCAM
For UCF, VTAM, and TCAM, you can choose whether to associate a logical/physical terminal pair with a particular device. If you do not explicitly associate a terminal pair with a device, the system takes the first available pair when a user signs on.

**Other Access Methods**

For all other access methods, you must associate terminal pairs with devices. In this case, users are always assigned a specific logical and physical terminal when they sign on to a particular device.

**How to Associate Terminals with Devices**

To associate UCF, VTAM, and TCAM terminals with particular devices, you use the `NAME IS` parameter of the system generation PTERM statement.

Suppose, for example, that the system definition includes logical terminal LT12012 and physical terminal PT12012. To associate this terminal pair with device FT068109, submit this information to the system generation compiler:

```
MODIFY PTERM PT12012
   ACQUIRE
   NAME IS FT068109.
```

**Note:** For more information on associating terminals with devices, see the *CA IDMS Administrating* section.

### Checking Authority to Access a Particular Terminal

#### How to Check a User's Authority

When you implement autotasks, users can access only preassigned tasks at particular terminals. If you want to prevent users from accessing terminals that are not associated with a specific task, you must check their authority to use these terminals.

You can do this by:

1. Establishing installation codes that authorize users to access particular physical terminals. For example, you can specify a physical terminal ID, logical terminal ID, or VTAM node name as the attribute value for INSTCODE in the user or system profile.

2. Checking a user's INSTCODE attribute when the user signs on to a terminal.

### Design Considerations

**Secure the DCMT VARY LTERM Command**

Users can override the autotask assigned to a logical terminal with the DCMT VARY LTERM ONLINE command. If you are using autotasks in a secure environment, be sure to secure this DCMT command.
Associate Terminals with Devices Only when Necessary

When using autotasks, you should consider how many people will be using terminals that are restricted to particular tasks. You can use your terminal network more effectively if you associate terminals with devices only when necessary.

Terminal Association Criteria:

<table>
<thead>
<tr>
<th>Need</th>
<th>Terminal configuration</th>
</tr>
</thead>
</table>
| A few people who will be using terminals restricted to an autotask | Associate the restricted terminals with particular devices.  
Keep the other terminals generic (that is, not associated with particular devices).  
Optionally, check a user's authorization to use a particular logical terminal at signon. |
| Many people who will be using terminals restricted to an autotask | Associate the non-restricted terminals with particular devices.  
Keep the restricted terminals generic.  
Optionally, check a user's authorization to use a particular logical terminal at signon. |