Migration

SAP Sybase IQ 16.0

UNIX/Linux

DOCUMENT ID: DC01778-01-1600-01
LAST REVISED: February 2013

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

Read Me First

Although the SAP® Sybase® IQ 16 New Features Summary describes all new SAP Sybase IQ functionality, some features may require additional action on your part to take advantage of the new architecture.

Customers upgrading from a previous release, for example, may need to change some initial compatibility options or rebuild wide columns to accommodate different datatypes. The new load engine provides better performance, but requires changes to the default memory allocation to use all available hardware resources efficiently.

NBit

Continuous `NBit` dictionary compression replaces 1, 2, 3 byte dictionary compression as the default column storage mechanism in 16.0. All datatypes except LOB (character and binary) and BIT datatypes can be `NBit` columns.

The `IQ UNIQUE` column constraint determines whether a column loads as `Flat FP` or `NBit FP`. An `IQ UNIQUE` n value set to 0 loads the column as `Flat FP`. An n value greater than 0 but less than the `FP_NBIT_AUTOSIZE_LIMIT` creates a `NBit` column initially sized to *n*. Columns without an `IQ UNIQUE` constraint implicitly load as `NBit` up to the auto-size limit.

Using `IQ UNIQUE` with an *n* value less than the auto-size limit is not necessary. The load engine automatically sizes all low or medium cardinality columns as `NBit`. Use `IQ UNIQUE` in cases where you want to load the column as `Flat FP` or when you want to load a column as `NBit` when the number of distinct values exceeds the auto-size limits.

Loads and Large Memory

Large memory represents the maximum amount of memory that SAP Sybase IQ can dynamically request from the OS for temporary use. Because some load operations may require more large memory than the 2GB default provides, adjust the startup options that control large and cache memory allocation based on the total amount of available physical memory.

As a general rule, large memory requirements represent one third of the total available physical memory allocated to SAP Sybase IQ. To ensure adequate memory for the main and temporary IQ stores, set the **–iqlm**, **–iqtc**, and **–iqmc** startup parameters so that each parameter receives one third of all available physical memory allocated to SAP Sybase IQ.

In most cases, you should allocate 80% of total physical memory to SAP Sybase IQ to prevent SAP Sybase IQ processes from being swapped out. Adjust actual memory allocation to accommodate other processes running on the same system. For example, on a machine with 32 cores and 128GB of total available physical memory, you would allocate 100GB (approximately 80% of the 128GB total) to SAP Sybase IQ processes. Following the general rule, you would set the **–iqlm**, **–iqtc**, and **–iqmc** parameters to 33GB each.

Database Options

Some database options are not enabled to take advantage of 16.0 features. Maintaining limited compatibility after a database upgrade provides some flexibility to transition existing applications.

Database Option	Description
FP_NBIT_IQ15_COMPATIBILITY	Provides tokenized `FP` support similar to that available in 15.x. This option is ON in all 16 databases upgraded from 15.x and OFF in all newly created databases. • If this option is ON, the database engine uses the `MINIMIZE_STORAGE`, `FP_LOOKUP_SIZE`, and `FP_LOOKUP_SIZE_PPM` options to optimize column storage. These options are ignored in 16.0. • If this option is OFF, the database columns conform to SAP Sybase IQ 16.0 `NBit` storage options. Set this option to OFF to take advantage of the new `NBit` dictionary column compression. See *FP_NBIT_IQ15_COMPATIBILITY Option* in *Reference: Statements and Options*.
CREATE_HG_WITH_EXACT_DISTINCTS	Determines whether newly created `HG` indexes are tiered or non-tiered. This option is ON in databases upgraded from 15.x and all newly created databases. Set this option to OFF to take advantage of the new tiered `HG` index structure. See *CREATE_HG_WITH_EXACT_DISTINCTS Option* in *Reference: Statements and Options*.

Database Option	Description
REVERT_TO_V15_OPTIMIZER	Forces the query optimizer to mimic 15.x behavior. This option is ON in 16.0 databases upgraded from 15.x. and OFF in all newly created 16.0 databases. If you plan to use the new 16.0 hash partitioning options, set this to ON. See *REVERT_TO_V15_OPTIMIZER Option* in *Reference: Statements and Options*.

Index Changes

Changes to `FP` and `HG` indexes take advantage of the new column compression mechanism and improve load performance.

Index	Description
New Fast Projection (FP) Indexes	Take advantage of the new continuous `NBit` dictionary compression, which replaces `FP(1)`, `FP(2)`, and `FP(3)` byte dictionary compression. `FP(1)`, `FP(2)`, and `FP(3)` indexes roll over to `NBit(8)`, `NBit(16)`, and `NBit(24)` respectively. If `FP_NBIT_IQ15_COMPATIBILITY='OFF'`, `IQ UNIQUE` constraints applied to the column determine whether the column loads as `Flat FP` or `NBit`. See *Fast Projection (FP) Index* in *Administration: Database*.
New tiered HG index structure	Decouples load performance from `HG` index size. In 15.x, load throughput could degrade as the amount of data in an `HG` index increased. As the index grew, loading the same amount of data could take more time. The new tiered structure decouples load performance from the `HG` index size to increase throughput. The `CREATE_HG_WITH_EXACT_DISTINCTS` option determines whether newly created `HG` indexes are tiered or non-tiered. This option is ON in all new 16.0 databases and all 16.0 databases migrated from 15.x. To take advantage of the new structure, set this option to OFF. Use **sp_iqrebuildindex** to convert non-tired `HG` indexes to tiered `HG` and vice-versa. See *CREATE_HG_WITH_EXACT_DISTINCTS Option* in *Reference: Statements and Options* .

Stored Procedures

New stored procedures return information about column indexes and constraints.

Procedure	Description
sp_iqindexmetadata	Returns details about column indexes, including the index types (`Flat FP`, `NBit`, `HG`, and tiered `HG`), distinct counts, `IQ UNIQUE` *n* value, and `NBit` dictionary size. See *sp_iqindexmetadata Procedure* in *Reference: Building Blocks, Tables, and Procedures*
sp_iqcolumnmetadata	Returns `FP` index metadata for one or more user tables or all tables in the database. See *sp_iqcolumnmetadata Procedure* in *Reference: Building Blocks, Tables, and Procedures*
sp_iqindexrebuildwidedata	Identifies wide columns that you must rebuild before they are available for read/write activities. Output includes statements that you can use with **sp_iqrebuildindex** to rebuild the columns. See *sp_iqindexrebuildwidedata Procedure* in *Reference: Building Blocks, Tables, and Procedures*
sp_iqrebuildindex	Rebuilds `FP` indexes (`Flat FP as NBit`, or `NBit` as `Flat FP`) and `HG` indexes (single `HG` as tiered `HG`, or tiered `HG` as single `HG`). Before you can insert or update new data, you must rebuild all columns greater than 255 bytes wide. The `index_clause` can reset `IQ UNIQUE` n to an explicit value from 0 (to recast an `NBit` column to `Flat FP`) up to the limits defined in the `FP_NBIT_AUTOSIZE_LIMIT` and `FP_NBIT_LOOKUP_MB` options. **sp_iqrebuildindex** also enables read-write access to columns that contain large object (LOB) data. LOB columns migrated from 15.x databases are read-only until you run **sp_iqrebuildindex**.The estimated cardinality for `NBit` columns with an `IQ UNIQUE` value below or equal to the `FP_NBIT_AUTOSIZE_LIMIT` is stored as 0 regardless of the `FP_NBIT_IQ15_COMPATIBILITY` setting. This affects the value returned from **sp_iqindexmetadata**. See *sp_iqrebuildindex Procedure* in *Reference: Building Blocks, Tables, and Procedures*

Object Names

Reserved words cannot be used as object names.

A SAP Sybase IQ 15.x database could contain tables, columns, and other objects named row. In SAP Sybase IQ 16.0, row is a reserved word and cannot be used as an object name.

To use a reserved word as an object name, enclosed the object name in brackets (regardless of the `QUOTED_IDENTIFIER` setting) or double quotes (if `QUOTED_IDENTIFIER='ON'` [default]):

```
// QUOTED_IDENTIFIER ON | OFF
select * from [row];
alter table row2 rename [row] to col_row;

// QUOTED_IDENTIFIER='ON'
select "row" from row2;
alter table "row" rename rownew;
```

Maintenance Releases

SAP® Sybase® IQ support packages include updates to features that are currently installed on your system.

SAP Sybase IQ support packages are available on the SAP Sybase Product Download Center at *http://downloads.sybase.com/swd/base.do?client=support*. All support packages include a cover letter with specific information about that release. Review the cover letter before you install the upgrade.

Preparing to Install Maintenance Releases

Perform these tasks before you install a maintenance release.

1. In a terminal, change to `$IQDIR16/bin64`, and enter:

   ```
   start_iq -v2
   ```

   If SAP Sybase IQ returns a version string that does not match the baseline version in the cover letter, you cannot perform a rolling upgrade. See *Database Upgrades* for alternate upgrade instructions.
2. Download the maintentance release from the Software Downloads for EBFs and Maintenance site at: *http://downloads.sybase.com/swd/base.do?client=support*.

   For details about SAP Sybase software downloads, see Software Downloads Frequently Asked Questions at: *http://downloads.sybase.com/swd/jsp/faq.jsp*.
3. Back up your current installation and save copies of any changes you made to default login and post-login scripts. Before you proceed, make sure the backups are readable.
4. Check with your operating system vendor for information on the latest operating system patches. Use the recommended operating system patch. Do not use a patch that is earlier than the version suggested for your operating system.
5. In Interactive SQL, run **sp_iqcheckoptions** on each database and capture the output.

   sp_iqcheckoptions generates a list of current database values and options. Use these values to restore your database settings after you upgrade.
6. Validate your license against each database to ensure that your license maintenance support is in (or near) compliance. A server that has not had a maintenance contract for more than a year does not run. Validating your license(s) ensures that your database functions correctly after you install the ESD.

   See the *SySAM* documentation for validation procedures.

See also

- *Installing ESDs* on page 8

- *Restoring Previous Software Versions* on page 8

Installing ESDs

Use these general instructions to install SAP Sybase IQ maintenance releases.

Some steps may differ for simplex and multiplex servers. See the cover letter included with the ESD for release-specific instructions.

Do not run **ALTER DATABASE UPGRADE**. If you do not run this command, you can roll the binary back to the previous version. You must, however, roll the coordinator back first.

1. Do one of the following:

Server	Action
Simplex	Shut down the server.
Multiplex	Shut down the multiplex node. Upgrade the secondary multiplex nodes first, one node at a time. Upgrade the coordinator last. A multiplex can include nodes running different software versions.

   For server shutdown instructions, see *Administration: Database > Run Database Servers > Ways to Start and Stop Databases*.
2. Install the ESD.

   For installation options, see *Installation and Configuration Guide > Server Installations > Installing Server Software*.
3. In a terminal, change to `$IQDIR16/bin64`.
4. Check the server version string, by entering:

   ```
   start_iq -v2
   ```
5. Restart the server or multiplex node.

See also

- *Preparing to Install Maintenance Releases* on page 7
- *Restoring Previous Software Versions* on page 8

Restoring Previous Software Versions

Use these general instructions to restore SAP Sybase IQ to a previous version.

Perform this task only to restore a previous software version after installing an ESD as a rolling upgrade. Some steps differ for simplex and multiplex servers. See the cover letter included with the ESD for release-specific instructions.

1. Do one of the following:

Server	Action
Simplex	Shut down the server.
Multiplex	Shut down the multiplex node. Begin the rollback with the coordinator node. Roll back the secondary nodes one node at a time.

   For server shutdown instructions, see *Administration: Database > Run Database Servers > Ways to Start and Stop Databases*.
2. Uninstall the ESD.

   See *Installation and Configuration Guide for Solaris > Server Installations > Uninstalling UNIX Servers* for details.
3. Reinstall the SAP Sybase IQ standalone version.
4. In a terminal, change to `$IQDIR16/bin64`.
5. To check the server version string, enter:

   ```
   start_iq -v2
   ```
6. Restart the server or multiplex node.

See also

- *Preparing to Install Maintenance Releases* on page 7
- *Installing ESDs* on page 8

Database Upgrades

Use these procedures to upgrade SAP Sybase IQ 15 simplex and multiplex databases.

1. *Preparing for a Database Upgrade*

   Perform these tasks before upgrading a database to version 16.0.
2. *Changes to System Procedures that Perform Privileged Operations*

   As part of the enhanced security of role-based security, the way in which privileged system procedures run has changed. Pre-16.0, a privileged system procedure ran with the privileges of its owner, typically dbo, and is referred to as the SYSTEM PROCEDURE DEFINER model. With 16.0, privileged system procedures run with the privileges of the person executing it, and is referred to as the SYSTEM PROCEDURE INVOKER model.
3. *Upgrading SAP Sybase IQ 15 Databases*

   Perform these steps to upgrade SAP Sybase IQ 15 simplex and multiplex databases to SAP Sybase IQ 16.0.
4. *Post Upgrade Status*

   SAP Sybase IQ 16 databases upgraded from SAP Sybase IQ 15.x are initially set to run in SAP Sybase IQ 15.x compatibility mode. To complete the change from 15.x to 16.0, you must explicitly change several 15.x compatibility settings to complete the 16.0 upgrade.
5. *Regrant the Ability to Run Privileged System Procedures After Upgrade*

   The method to regrant the ability to run privileged system procedures after an upgrade depends on the underlying security model of the procedure.
6. *Logical Servers*

   An SAP Sybase IQ 16.0 multiplex database upgrade changes the way users access multiplex servers. Starting with SAP Sybase IQ 15.4, logical servers provide the only means to access the multiplex server nodes.

Preparing for a Database Upgrade

Perform these tasks before upgrading a database to version 16.0.

1. Disconnect all users from the server.
2. Back up the SAP Sybase IQ 15 database.
3. From the database, drop:
   - All `JOIN` and `LD` indexes
   - (Multiplex users) Logical servers named **AUTO**, **COORDINATOR**, **ALL** or **DEFAULT**

If you do not drop these objects, **ALTER DATABASE UPGRADE** fails. To recover, open the database with the SAP Sybase IQ 15.x binary and drop all join indexes and the named logical servers.

4. Update `DATE` columns that contain a time portion.

   There is a known issue that affects any partition defined on a `DATE` column that contains a time portion:.

   - Use **ALTER TABLE MERGE** to combine the partition with the next partition.
   - Use **ALTER TABLE SPLIT** to divide the resulting partition into a definition with the same criteria, but no time portion.
5. Drop and re-create all SAP Sybase IQ 15.2 `TEXT` indexes that have not already been dropped and recreated as part of a version 15.2 ESD upgrade.

   TEXT indexes created in SAP Sybase IQ 15.2 are incompatible with later versions of SAP Sybase IQ.
6. On database upgrade using the ALTER DATABASE UPGRADE statement, privileged system procedures are dropped and re-created. As part of this process, any explicit EXECUTE privilege granted on system procedures is lost and must be manually regranted post upgrade.

Changes to System Procedures that Perform Privileged Operations

As part of the enhanced security of role-based security, the way in which privileged system procedures run has changed. Pre-16.0, a privileged system procedure ran with the privileges of its owner, typically dbo, and is referred to as the SYSTEM PROCEDURE DEFINER model. With 16.0, privileged system procedures run with the privileges of the person executing it, and is referred to as the SYSTEM PROCEDURE INVOKER model.

Note: This behavior change applies to SAP Sybase IQ privileged system procedures only, not user-defined stored procedures.

In pre-16.0, with the SYSTEM PROCEDURE DEFINER model, when you grant a user explicit EXECUTE privilege on a system procedure, any privileges required to run any authorized tasks associated with the system procedure are automatically inherited from the owner (definer of the system procedure), allowing the user to successfully run the system procedure.

In 16.0, with the SYSTEM PROCEDURE INVOKER model, the EXECUTE privilege for each system procedure is now granted to the PUBLIC role. Since every user, by default, is a member of the PUBLIC role, every user automatically inherits the required EXECUTE privilege. What is not inherited with the grant of EXECUTE privilege are any associated privileges required to run system procedure. These must now be granted directly or indirectly to the user before he or she can successfully run a system procedure.

This behaviour change has the potential to cause loss of functionality on custom stored procedures and applications that explicitly grant EXECUTE privilege on system procedures. For this reason, a default upgrade of a pre-16.0 database uses a combination of the two models. In the combination model, pre-16.0 privileged system procedures continue to run using the SYSTEM PROCEDURE DEFINER model, while any privileged system procedures introduced with 16.0 (or any future release) use the SYSTEM PROCEDURE INVOKER model.

If the potential loss of functionality is not of concern to your installation, you can override the default upgrade behavior so that all privileged system procedures (pre-16.0, new, and any future releases) use the SYSTEM PROCEDURE INVOKER model only. If you are unsure whether the potential loss of functionality will impact your database, upgrade using the default behavior and investigate. If you determine after the fact that it is not an issue, and you want to run all system procedures using the SYSTEM PROCEDURE INVOKER model, you can use the **ALTER DATABASE** statement to change the default security model.

The CREATE DATABASE statement, ALTER DATABASE UPGRADE statement, and Initialization utility (iqinit) have been enhanced to allow specification of a security model.

There is a small subset of pre-16.0 privileged system procedures that has always run with the privileges of the user running the procedure, not the owner of the procedure. To run these system procedures, in addition to requiring EXECUTE privilege on the system procedure, the user must be granted additional system privileges specific to the system procedure. Refer to the documentation for the required system privileges. This behavior remains unchanged in 16.0, regardless of the security model setting.

See also

- *Upgrading SAP Sybase IQ 15 Databases* on page 15

Pre-16.0 Privileged System Procedures

A list of pre-16.0 privileged system procedures.

Pre-16.0 privileged system procedures that use the combined security model

For these privileged system procedures, if the database is configured to run using SYSTEM PROCEDURE DEFINER, you only need EXECUTE privilege on the procedure to run it. If the database is configured to run using SYSTEM PROCEDURE INVOKER, you need the individual privileges that each procedure requires to run successfully. Refer to the documentation for each procedure's required system privileges.

- sa_audit_string
- sa_checkpoint_execute
- sa_clean_database
- sa_column_stats
- sa_conn_activity
- sa_conn_compression_info
- sa_conn_info
- sa_conn_list
- sa_conn_options
- sa_conn_properties
- sa_db_info
- sa_db_list
- sa_db_properties
- sa_disable_auditing_type
- sa_disk_free_space
- sa_enable_auditing_type
- sa_external_library_unload
- sa_flush_cache
- sa_flush_statistics
- sa_get_histogram
- sa_get_request_profile
- sa_get_request_times
- sa_get_table_definition
- sa_get_user_status
- sa_index_density
- sa_index_levels
- sa_install_feature
- sa_java_loaded_classes
- sa_list_external_library
- sa_load_cost_model
- sa_materialized_view_can_be_immediate
- sa_procedure_profile
- sa_procedure_profile_summary
- sa_recompile_views
- sa_refresh_materialized_views
- sa_refresh_text_indexes
- sa_remove_index_consultant_analysis
- sa_text_index_vocab
- sa_text_index_vocab_nchar
- sa_unload_cost_model
- sa_user_defined_counter_add
- sa_user_defined_counter_set
- sa_validate
- sp_iq_reset_identity
- sp_iqaddlogin
- sp_iqbackupdetails
- sp_iqbackupsummary
- sp_iqcardinality_analysis
- sp_iqcheckdb
- sp_iqcheckoptions
- sp_iqclient_lookup
- sp_iqcolumn
- sp_iqcolumnuse
- sp_iqconnection
- sp_iqconstraint
- sp_iqcontext
- sp_iqcopyloginpolicy
- sp_iqcursorinfo
- sp_iqdatatype
- sp_iqdbsize
- sp_iqdbspace
- sp_iqdbspaceinfo
- sp_iqdbspaceobjectinfo
- sp_iqdbstatistics
- sp_iqdroplogin
- sp_iqemptyfile
- sp_iqestdbspaces
- sp_iqestspace
- sp_iqevent
- sp_iqfile
- sp_iqhelp
- sp_iqmodifylogin
- sp_iqmpxcheckdqpconfig
- sp_iqmpxdumptlvlog
- sp_iqmpxfilestatus
- sp_iqmpxincconnpoolinfo
- sp_iqmpxincheartbeatinfo
- sp_iqmpxinfo
- sp_iqmpxversioninfo
- sp_iqobjectinfo
- sp_iqpkeys
- sp_iqprocedure
- sp_iqprocparm
- sp_iqrebuildindex
- sp_iqrename
- sp_iqrestoreaction
- sp_iqrowdensity
- sp_iqsetcompression
- sp_iqsharedtempdistrib
- sp_iqshowcompression
- sp_iqshowpsexe
- sp_iqspaceinfo
- sp_iqspaceused
- sp_iqstatistics
- sp_iqstatus
- sp_iqsysmon
- sp_iqtable
- sp_iqtablesize
- sp_iqtableuse
- sp_iqtransaction
- sp_iqunusedcolumn
- sp_iqunusedindex
- sp_iqunusedtable
- sp_iqversionuse
- sp_iqview
- sp_iqwho
- sp_iqworkmon
- st_geometry_load_shapefile
- xp_cmdshell

- sa_reset_identity
- sa_save_trace_data
- sa_send_udp
- sa_server_option
- sa_table_fragmentation
- sa_table_page_usage
- sa_table_stats
- sa_text_index_stats
- sp_iqindex
- sp_iqindex_alt
- sp_iqindexadvice
- sp_iqindexfragmentation
- sp_iqindexinfo
- sp_iqindexmetadata
- sp_iqindexsize
- sp_iqindexuse
- sp_iqlmconfig
- sp_iqlocks
- sp_iqmodifyadmin
- xp_read_file
- xp_sendmail
- xp_startmail
- xp_startsmtp
- xp_stopmail
- xp_stopsmtp
- xp_write_file

Pre-16.0 privileged system procedures that run with invoker privileges regardless of the security model

These pre-16.0 privileged system procedures run with the privileges of the user running the procedure, not the owner of the procedure, regardless of the security model setting. This means that in addition to requiring EXECUTE privilege on the system procedure, the user must be granted additional system privileges required by the system procedure. Refer to the documentation for the required system privileges.

- sa_describe_shapefile
- sa_get_user_status
- sa_locks
- sa_performance_diagnostics
- sa_report_deadlocks
- sa_text_index_stats

Upgrading SAP Sybase IQ 15 Databases

Perform these steps to upgrade SAP Sybase IQ 15 simplex and multiplex databases to SAP Sybase IQ 16.0.

Warning! Failure to complete this upgrade before you perform any read-write operations in the 16.0 database may result in unintended consequences.

1. Do one of the following:

Server	Action
Simplex	Shut down the server.

Server	Action
Multiplex	Shut down all multiplex nodes.

Note: If the server stops responding during shutdown, do not proceed to the next step. Restart the database with SAP Sybase IQ 15 and shut down the server. Proceed to the next step only on a clean shutdown.

2. Do one of the following:

Sever	Action
Simplex	Start the SAP Sybase IQ 16.0 server using the **-gm 1** and **-iqro 1** startup flags.
Multiplex	Use SAP Sybase IQ 16.0 to restart the coordinator using the **-iqmpx_sn 1**, **-gm 1**, and **-iqro 1** startup flags.

The **-gm** switch controls the number of connections. If Sybase Control Center is running, use **-gm 2** or the upgrade may fail.

3. Start Interactive SQL and connect to the database.
4. Do one of the following to upgrade the database:

System Procedure Security Model	SQL Syntax
Combination model (default)	**ALTER DATABASE UPGRADE**
SYSTEM PROCEDURE INVOKER model only	**ALTER DATABASE UPGRADE SYSTEM PROCEDURE AS DEFINER OFF**

5. Run **sp_iqcheckdb ('allocation database')** to verify that there are no errors.
6. Do one of the following:

Server	Action
Simplex	Shut down and restart the server normally (without the **-gm 1** and **-iqro 1** startup flags).
Multiplex	Shut down and restart the coordinator normally (without the **-iqmpx_sn 1**, **-gm 1**, and **-iqro 1** startup flags). Synchronize and restart all multiplex secondary servers.

7. Back up the database.

See also

- *Changes to System Procedures that Perform Privileged Operations* on page 12

Post Upgrade Status

SAP Sybase IQ 16 databases upgraded from SAP Sybase IQ 15.x are initially set to run in SAP Sybase IQ 15.x compatibility mode. To complete the change from 15.x to 16.0, you must explicitly change several 15.x compatibility settings to complete the 16.0 upgrade.

Indexes

- In Fast Projection (`FP`) indexes, continuous `NBit` dictionary compression replaces `FP(1)`,`FP(2)`, and `FP(3)` byte dictionary compression. `FP(1)`,`FP(2)`, and `FP(3)` indexes roll over to `NBit(8)`,`NBit(16)`, and `NBit(24)` respectively. All data types except LOB (both character and binary) and BIT data types may be **NBit** columns.

  If `FP_NBIT_IQ15_COMPATIBILITY` is OFF, `IQ UNIQUE` determines whether the column loads as `Flat FP` or `NBit`. Setting `IQ UNIQUE` to 0 loads the column as `Flat FP`. Columns without an `IQ UNIQUE` constraint load as `NBit` up to the `NBit` auto-sizing limits.
- New tiered `HG` index structure decouples load performance from `HG` index size. In SAP Sybase IQ 15, load throughput could degrade as the amount of data in an `HG` index increased. As the index grew, loading the same amount of data could take more time. The new tiered structure decouples load performance from the `HG` index size to increase throughput.

  The `CREATE_HG_WITH_EXACT_DISTINCTS` option determines whether newly created `HG` indexes are tiered or non-tiered. If this option is ON, all new `HG` indexes are non-tiered. To take advantage of the new structure, set this option to OFF. Use **sp_iqrebuildindex** to convert non-tiered `HG` indexes to tiered `HG` and vice-versa .

Column Constraints

Constraint	Description
IQ UNIQUE	In SAP Sybase IQ 16.0, `IQ UNIQUE` explicitly defines the expected cardinality of a column and determines whether the column loads as `Flat FP` or `NBit`. Columns retain their `IQ UNIQUE(`*n*`)` value during a 15.x to 16.0 database upgrade. Setting `IQ UNIQUE` to 0 loads the column as `Flat FP`. Columns without an `IQ UNIQUE` constraint or columns with an `IQ UNIQUE` *n* value less that is less than the limit defined by the `FP_NBIT_AUTOSIZE_LIMIT` option is not necessary. Auto-size functionality automatically sizes all low or medium cardinality columns as `NBit`. Use `IQ UNIQUE` in cases where you want to where you want to load the column as `Flat FP` or when you want to load as `NBit` and the number of distinct values exceeds the auto-size limits.

Database Options

Option	Description
FP_NBIT_IQ15_COMPATIBILITY	Provides tokenized **FP** support similar to that available in 15.x. This option is ON by default in all 16.0 databases upgraded from 15.x and OFF in all newly created 16.0 databases. • If this option is ON, the database engine uses the `MINIMIZE_STORAGE`, `FP_LOOKUP_SIZE`, and `FP_LOOKUP_SIZE_PPM` options to optimize column storage. These options are ignored in 16.0. • If this option is OFF, the database engine ignores 15.x options and columns conform to SAP Sybase IQ `NBit` storage options. Set this option to OFF to take advantage of `NBit` column compression.

Option	Description
CREATE_HG_WITH_EXACT_DISTINCTS	Determines whether new HG indexes explicitly created with a **CREATE INDEX** command, or implicitly creating or altering a table with a PRIMARY KEY or a FOREIGN KEY declaration, are tiered or non-tiered. This option is ON 16.0 databases upgraded from 15.x and all newly created 16.0 databases. If this option is ON, all new HG indexes are non-tiered. To take advantage of the new structure, set this option to OFF. To take advantage of the new tiered structure, set this option to OFF. Use **sp_iqrebuildindex** to convert non-tiered HG indexes to tiered HG and vice-versa.
REVERT_TO_V15_OPTIMIZER	`REVERT_TO_V15_OPTIMIZER` forces the query optimizer to mimic SAP Sybase IQ 15.x behavior. `REVERT_TO_V15_OPTIMIZER='ON'` by default in all 16.0 databases upgraded from 15.x. `REVERT_TO_V15_OPTIMIZER='OFF'` by default in all newly created SAP Sybase IQ 16.0 databases. If you plan to use SAP Sybase IQ hash partitioning features, set the REVERT_TO_V15_OPTIMIZER ='OFF' in databases upgraded from 15.x to SAP Sybase IQ.

Startup Options

Some load operations may require more large memory than the 2GB default provides. If memory requirements exceed the default, use the **- iqlm** startup option to increase the memory that SAP Sybase IQ can dynamically request from the OS. Set **–iqlm** as a switch as part of the command or configuration file that starts the server.

As a general rule, large memory requirements represent one third of the total available physical memory allocated to SAP Sybase IQ. To ensure adequate memory for the main and temporary IQ stores, set the **–iqlm**, **–iqtc**, and **–iqmc** startup parameters so that each parameter receives one third of all available physical memory allocated to SAP Sybase IQ.

In most cases, you should allocate 80% of total physical memory to SAP Sybase IQ to prevent SAP Sybase IQ processes from being swapped out. Adjust actual memory allocation to accommodate other processes running on the same system. For example, on a machine with 32 cores and 128GB of total available physical memory, you would allocate 100GB (approximately 80% of the 128GB total) to SAP Sybase IQ processes. Following the general rule, you would set the **–iqlm**, **–iqtc**, and **–iqmc** parameters to 33GB each.

Object Names

Reserved words cannot be used as object names.

A SAP Sybase IQ 15.x database could contain tables, columns, and other objects named row. In SAP Sybase IQ 16.0, row is a reserved word and cannot be used as an object name.

To use a reserved word as an object name, enclosed the object name in brackets (regardless of the `QUOTED_IDENTIFIER` setting) or double quotes (if `QUOTED_IDENTIFIER='ON'` [default]):

```
// QUOTED_IDENTIFIER ON | OFF
select * from [row];
alter table row2 rename [row] to col_row;

// QUOTED_IDENTIFIER='ON'
select "row" from row2;
alter table "row" rename rownew;
```

Stored Procedures

Use these stored procedures to review and change column indexes and constraints:

Procedure	Description
sp_iqcolumnmetadata	Returns index metadata for all columns in one or more tables.
sp_iqindexmetadata	Returns details about column indexes, including the index types (`Flat FP`, `NBit`, `HG`, and tiered `HG`), distinct counts, `IQ UNIQUE` *n* value, and `NBit` dictionary size.

Procedure	Description
sp_iqrebuildindex	Rebuilds `FP` indexes (`Flat FP` as `NBit`, or `NBit` as `Flat FP`) and `HG` indexes (single `HG` as tiered `HG`, or tiered `HG` as single `HG`). Before you can insert or update new data, you must rebuild all columns greater than 255 bytes wide. The `index_clause` can reset `IQ UNIQUE` *n* to an explicit value from 0 (to recast an `NBit` column to `Flat FP`) up to the limits defined in the `FP_NBIT_AUTOSIZE_LIMIT` and `FP_NBIT_LOOKUP_MB` options. **sp_iqrebuildindex** also enables read-write access to columns that contain large object (LOB) data. LOB columns migrated from 15.x databases are read-only until you run **sp_iqrebuildindex**. The estimated cardinality for `NBit` columns with an `IQ UNIQUE` value below or equal to the `FP_NBIT_AUTOSIZE_LIMIT` is stored as 0 regardless of the `FP_NBIT_IQ15_COMPATIBILITY` setting. This affects the value returned from **sp_iqindexmetadata**.
sp_iqindexrebuildwidedata	Identifies wide columns that you must rebuild before they are available for read/write activities. **sp_iqindexrebuildwidedata** also generates a list of statements that you can use to to rebuild the columns. This applies to CHAR, VARCHAR, BINARY, and VARBINARY columns wider than > 255 characters, as well as all Long Varchar and Long Binary columns.

Regrant the Ability to Run Privileged System Procedures After Upgrade

The method to regrant the ability to run privileged system procedures after an upgrade depends on the underlying security model of the procedure.

If you upgraded your database using the default statement, all pre-16 privileged system procedures use the SYSTEM PROCEDURE DEFINER model, while all other privileged system procedures use the SYSTEM PROCEDURE INVOKER model. If you overrode the security model default in the database upgrade statement, all privileged system procedures (pre- and post-16.0) use the SYSTEM PROCEDURE INVOKER model.

Security Model	Regrant Method
SYSTEM PROCEDURE DEFINER model	Grant EXECUTE object-level privilege on the system procedure directly to the user or role to run the procedure.
SYSTEM PROCEDURE INVOKER model	Use **sp_proc_priv()** to identify the system privileges required to run a system procedure. Grant these system privileges to the user or role to run the procedure.

Logical Servers

An SAP Sybase IQ 16.0 multiplex database upgrade changes the way users access multiplex servers. Starting with SAP Sybase IQ 15.4, logical servers provide the only means to access the multiplex server nodes.

Upgrading a multiplex database creates an appropriate logical server for each server-specific login policy. Login policies are updated to use a logical server configuration that provides access to the same set of multiplex servers that they did prior to upgrade.

If a login policy does not allow access to any node (such as when base setting of `LOCKED` is `ON` and there are no multiplex server-level overrides), the login policy is set to a system-defined logical server, `NONE`, instead of creating a new logical server. `NONE` indicates that the login policy does not allow access to any multiplex server.

If a login policy has no explicit setting for the `LOCKED` option, either at the base level or via a multiplex server-level override, no logical server is created for that policy. Such a login policy inherits the logical server assignment of the root login policy.

- Membership configuration of a logical server provides access to the same multiplex nodes as the corresponding 15.x login policy. A logical membership of the coordinator is also added to the logical server if the login policy allowed access to the current coordinator server.

- Logical server names are derived from the login policy names. If the login policy name is fewer than 126 characters, the logical server follows this naming convention: **LS_<login policy name>**. For example, for a login policy named **mpx_grp1**, a logical server **LS_mpx_grp1** is created and assigned to that login policy.

  If the login policy name exceeds 125 characters, a logical server is created with the same name as of the login policy, that is, without adding an **LS_** prefix.
- During the upgrade, some login policy option settings or multiplex server-level overrides are reset or removed. In the root login policy, `LOCKED` and `MAX_CONNECTIONS` overrides are reset to default values ('`OFF`' and 10 respectively).

  Settings for `LOCKED` and `MAX_CONNECTIONS` are removed from user-defined login policies. Multiplex server-level overrides are removed from all login policies.
- The login policy option `LOGIN_REDIRECTION` is added to the root logical server policy with its value set to `'OFF'` to retain pre-upgrade behavior for existing applications.

Note: See *Administration: Multiplex > Manage Resources Through Logical Servers*.

Hardware Changes

Perform these steps to move your software to a new hardware platform.

Moving 32-Bit Databases to 64-bit Platforms

Perform these steps to move a 32-bit database to a 64-bit platform.

Prerequisites

- Review backup and restore procedures:
  - For simplex servers, see *Administration: Backup, Restore, and Data Recovery* .
  - For multiplex servers, see *Administration: Multiplex > Back Up and Restore*.
- Make note of the 32-bit server raw device and IQ store path names. Raw device and IQ store path names on the 64-bit target must match those on the 32-bit machine.

Task

1. Log in to your 32-bit server and back up the database.
2. Copy the backup to the 64-bit machine, and restore the database.

   You may need to rename raw device and path names to ensure they match. See *Administration: Multiplex > Back Up and Restore*.
3. On the 64-bit machine, do one of the following:

Server	Action
Simplex	Start the database with the appropriate startup flags.
Multiplex	Start the coordinator with the **-iqmpx_sn 1**, **-gm 1** , **-iqro 1**, and **-iqmpx_ov 1** startup flags.

4. Start Interactive SQL and connect to the database.
5. Use **DROP MULTIPLEX SERVER** to drop all existing secondary nodes.
6. Use **ALTER DATABASE UPGRADE** to upgrade the database.

   See *Reference: Statements and Options > SQL Statements > ALTER DATABASE Statement*.
7. Run **sp_iqcheckdb ('allocation database')** and verify that the database is error free.

   sp_iqcheckdb checks the validity of the current database. See *Reference: Building Blocks, Tables, and Procedures > System Procedures > sp_iqcheckdb Procedure*.
8. Perform these steps for multiplex servers only:

a) Shut down and restart the coordinator normally (without the **-gm 1** , **-iqro 1**, and **-iqmpx_ov** startup flags).
b) Use **CREATE MULTIPLEX SERVER STATEMENT** to recreate the secondary nodes.

See also

- *Converting to a New Hardware Platform* on page 26

Converting to a New Hardware Platform

Perform these steps to move a database to another platform with the same endian structure.

Platforms must share the same endian structure. Move your database, then migrate your data.

1. Back up the database.
2. Shut down the SAP Sybase IQ server.
3. Install the server on the new platform. Your migration can take place on the same or a different machine.
4. Start the server on the new hardware platform.
5. Connect to the utility database, `utility_db`.
6. Restore the database from the backup you created in step 1.
7. Shut down the server and restart it against the restored database.
8. Start Interactive SQL and run **ALTER DATABASE UPGRADE**.

Note: If the SAP Sybase IQ version is more recent than the version on legacy platform, you must upgrade the database.

See also

- *Moving 32-Bit Databases to 64-bit Platforms* on page 25

SAP Sybase IQ 12.7 Database Migration

Updating a 12.7 catalog to 16.0 requires a database file migration, not a simple database upgrade. Migration tools bundled with SAP Sybase IQ 16.0 can recreate the legacy database schema and database options.

Preparing to Migrate

Perform these tasks before you migrate your database.

1. Upgrade to SAP Sybase IQ 12.7 ESD #5. All migration paths assume that you are migrating from SAP Sybase IQ 12.7 ESD #5 as a minimum.
2. Regenerate any sort-key values. SAP Sybase IQ 12.7 **SORTKEY** function uses a different sort-key value than SAP Sybase IQ 15 and later.
3. Review the collation. SAP Sybase IQ no longer supports custom collations. Custom collations are preserved in database rebuilds only if you rebuild the database in a single step. Use a collation included with SAP Sybase IQ 15.0 or later.
4. Back up your current installation and save copies of any changes you made to default login and post login scripts. Create your backups on removable media, like tape, DVD, or CD. Make sure the backups are readable.
5. Review and understand the database migration utilities. Use **iqunload** to re-create the schema for your database or migrate your 12.7 database. **iqlsunload** utility is available to move 12.7 local stores for 12.7 multiplex servers.
6. Use **DROP TABLE** statements to drop all global temporary tables before you run the **iqunload** utility. You can recreate the global temporary tables after migration.
7. Drop all servers of type `asejdbc` before you run the **iqunload** utility.

   The `asejdbc` server class is deprecated. Servers of type `asejdbc` must be dropped prior to running iqunload. 16.0 12.7 databases with remote server definitions based on the `asejdbc` driver will not have these definitions migrated to by the **iqunload** tool and will not give an error at the beginning of an unload saying that you need to drop any asejdbc servers (if there are any present).
8. Resolve potential migration errors. SAP Sybase IQ no longer supports some 12.7 features and objects. Update these objects before you migrate the database.
9. Use **sp_iqcheckdb** to verify that your 12.7 database is clean and error free.

   For information on **sp_iqcheckdb** output, see *Administration: Backup, Restore, and Data Recovery > System Recovery and Database Repair*.

Migration Utilities

Utilities and support tools bundled with SAP Sybase IQ.

iqunload Utility

iqunload is a command line utility for SAP Sybase IQ 12.6 and 12.7 database migration. **iqunload** re-creates the legacy catalog on the new database catalog in the current installation.

iqunload does not change SAP Sybase IQ data and temp dbspaces. The utility preserves all legacy database options and applies them to the new database. SAP Sybase IQ ignores any legacy options that no longer apply to the current version of the software.

Syntax

```
iqunload [ options ]  directory [ @data ]
```

```
data:[ environment variable | file ]
```

Parameters

iqunload takes one or more parameters.

- **-ap <*size*> –** (optional) Sets the page size for the new catalog store.
- **-au –** required for migration mode. Migrates the database.

  Specify an **-au** argument to start **iqunload** in migration mode. or **-n** argument, but not both.
- **-c "*keyword* = *value*, ..." –** (required) Supply database connection parameters. You must specify a DBF parameter to specify the name of the database file for migration. The file path is either absolute or relative to the server startup directory.
- **-dc –** (optional) Recalculate computed columns in the database.
- **-ms_filename –** optional for simplex migration; required for multiplex migration. Use **-ms_filename** to specify a file name for the new empty `IQ_SYSTEM_MAIN` store created during the migration.

  If not specified, the default new main store is a file system file called `new_main_store.iq`
- **-ms_reserve –** (optional) Specifies the size of the new `IQ_SYSTEM_MAIN` reserve, in MB. If unspecified, defaults to zero.
- **-ms_size –** (optional) specifies the size of the new `IQ_SYSTEM_MAIN` store, in MB, based on the database size. The minimum, assuming a default page size, is 200MB. If you specifiy an **–ms_size** value smaller than the computed value, SAP Sybase IQ uses the computed value; otherwise the specified value is used.

- **-n –** required for schema unload only. Unloads the schema definition only. The **-n** parameter requires 12.7 ESD #5 or later. Specify **-au** or **-n** argument but not both.
- **-new_startline –** (optional) specify startup switches for the new server that is the migration target. For a complete list of server startup switches, see the *Utility Guide*.
- **-o *file name* –** (optional) logs output messages to *file name*.
- **-q –** (optional) suppresses messages and windows.
- **-r *file name* –** (optional) specifies the file name.
- **-t *list* –** (optional) outputs listed tables only. Can specify `OwnerName.TableName` or `TableName` alone.
- **-v –** (optional) returns verbose messages.
- **-y –** (optional) replaces existing reload schema SQL script with new output without confirmation.

Examples

- **Example 1 –** migrates a simplex database to a current server.

  ```
  iqunload -au -c
  “UID=DBA;PWD=SQL;DBF=/mydevice/test_dir/test2.db”
  ```

- **Example 2 –** unloads a legacy database schema and renames the generated SQL script to `test2_reload.sql`:

  ```
  iqunload -n -c “UID=DBA;PWD=SQL;DBF=D:\\test_dir\\test2\
  \test2.db;
  ```

- **Example 3 –** migrates database `test3.db`. The **START** connection parameter specifies switches for starting the database being reloaded. The **-new_startline** parameter specifies switches for starting the `utility_db` to create the new catalog store.

  ```
  iqunload -au -c
  "UID=DBA;PWD=SQL;DBF=test3.db;START=-ch 128M -iqmc 50" -
  new_startline "-ch 256M -iqtc 400"
  ```

  Do not include **-iqnotemp** *XYZ* in the new start line or migration fails. As part of the migration process, temp files are added to `IQ_SYSTEM_TEMP`. If you start the server with the **-iqnotemp** option, **iqunload** cannot add these temp files

- **Example 4 –** migrates the legacy database, `asiqdemo.db`, using a raw device for the `IQ_SYSTEM_MAIN` store:

  ```
  iqunload -au -c
  "UID=DBA;PWD=SQL;DBF=asiqdemo.db" -ms_filename "/dev/rdsk/
  c4t0d0s3"
  ```

Usage

iqunload has two working modes: schema unload and migration.

Schema Unload Mode

iqunload requires an **-n** argument to start in schema unload mode. Schema unload mode unloads a 12.6 ESD #11 or 12.7 ESD #5 database schema, and generates a script (`reload.sql`) that can re-create the schema for a database in a current version of the software. A **-c** argument is required for connection parameters:

```
iqunload -n -c "UID=DBA;PWD=SQL;ENG=my_engine;DBN=my_dbname"
```

Schema unload mode re-creates the schema, but does not migrate data. To migrate data, extract the legacy data and load the new database.

Migration Mode

iqunload requires an **-au** argument to start in migration mode. **iqunload** migration mode interfaces with the 12.7 support engine (**iqunlspt**) and the current database server (**iqsrv16**):

- Start the legacy database and generate the schema
- Start the current SAP Sybase IQ server
- Create a new database and apply the legacy schema

General Usage

- Insufficient cache memory causes migration errors. **iqunload** uses default values for various cache sizes (catalog cache, main cache, temp cache). If the legacy database requires higher cache values, use the **–ch** and **-cl** options as part of the **START** connection parameter to increase the cache size. See the *Utility Guide* for details.
- During database migration, the server creates a message file (`*.iqmsg.R`) as it reloads the generated schema. This file is normally deleted as part of a cleanup operation for successful migrations. If the migration fails during the reload stage, cleanup does not occur, and `*.iqmsg.R` remains in the `unload` directory. `*.iqmsg.R` may contain information that can help solve your migration problems.
- **iqunload** writes some temporary files to the $IQTMP16 directory. If you set the $IQTMP16 environment variable, set it to a valid directory name.
- Users with wide tables (large numbers of column/null values) should not decrease the catalog store page size for database migration.
- If the legacy database contains invalid views, SAP Sybase IQ completes the migration but issues warnings. A warning may occur, for example, if the tables involved in a view are dropped.
- If the legacy database is encrypted, use the **DBKEY** connection parameter to provide the encryption key. The migrated database uses the same encryption key.

Permissions

DBA

See also

- *iqlsunload Utility* on page 31
- *Support Processes* on page 33

iqlsunload Utility

In current multiplex configurations, multiple nodes can write to the main store, which eliminates the need for local stores. **iqlsunload** is a command line utility that you can use to unload a 12.7 local store. **iqlsunload** is used only in 12.7 ESD #5 multiplex migrations.

iqlsunload is bundled with all versions of SAP Sybase IQ starting with 12.7 ESD #5.

Syntax

```
iqlsunload [ options ]  directory [ @data ]
data:[ environment variable | file ]
```

Parameters

- **directory –** (required) identifies the directory where **iqlsunload** unloads the data files. Create this directory before you run **iqlsunload**, or point to an existing directory. This directory must be relative to the database on the database server.
- **-al –** (optional) unloads IQ local store schema and data.
- **-c “*keyword*=*value*;...” –** (optional) supplies database connection parameters.
- **-h –** (optional) prints out the syntax (help) for the utility.
- **-o *filename* –** (optional) logs output messages, including errors, to *filename*.
- **-q –** (optional) suppresses messages and windows.
- **-r *directory* –** (optional) specifies the directory where SQL scripts are generated. The default reload command file is `reload.sql` in the current directory. The directory is relative to the current directory of the client application, not the server.
- **-t *list* –** (optional) outputs listed tables only. Can specify `OwnerName.TableName` or `TableName` alone. Cannot be specified with **al** argument.
- **-v –** (optional) outputs verbose messages.
- **-y –** (optional) replaces existing reload schema SQL script without confirmation.

Examples

- **Example 1 –** unload local stores from a database called `mpxtest2`, extracting any table data to the directory `/mydevice/test_dir/unload_dir` :

  ```
  iqlsunload -o iqunload_624.out -al
  -c “UID=DBA;PWD=SQL;ENG=myserver_mpxtest02”
  /mydevice/test_dir/unload_dir
  ```

Usage

General Notes

- Run **iqlsunload** from the `$IQDIR16/lsunload` directory to pick up updated libraries before resolving any IQ 12.7 libraries.

Unloaded Objects

Running **iqlsunload** with an **-al** argument unloads these persistent objects:

- Base tables
- Global temporary tables
- Indexes
- Domains (user-defined data types)
- Constraints (column check constraint, table constraint, primary key, foreign key, unique, default, IQ unique, not null)
- Views
- Stored procedures and functions
- Messages
- Remote servers and external logins
- Events

Empty User Names

SAP Sybase IQ no longer allow users with empty user names. You cannot drop or migrate users with empty user names the 12.6 or 12.7 server. The schema reload operation warns that an empty user name has been encountered and that the user will not be re-created. The reload operation ignores such users and any associated objects.

Unloading Tables

iqlsunload ignores any system tables or nonexistent tables:

- If you extract table schema and data only, the legacy database collation must match the collation of the current database collation.
- If you do not qualify table names with owner names, **iqlsunload** extracts table data from all tables with that table name.

Output Files

iqlsunload generates these output files:

Script Name	Description
`reload_schema.sql`	Recreates schema for unloaded objects (either objects from local store or tables selected by the user.) This script is executed against a node that writes to the multiplex. This node can be either the existing writer node, or a writer or coordinator for the multiplex after migration, depending on where you plan to recreate the schema.
`extract_data.sql`	Extracts table data for the unloaded tables from the local store. Execute this script in Interactive SQL while connected to the query node from which it was generated. When this script executes, it generates the data files into the directory data.

Script Name	Description
`reload_data.sql`	Loads extracted table data. This script is executed on the node where you ran `reload_schema.sql`, and reloads the data extracted from the `extract_data.sql` file.

Permissions

DBA

See also

- *iqunload Utility* on page 28
- *Support Processes* on page 33

Support Processes

Running **iqunload** in migration mode (**-au**) starts **iqunlspt** and **iqsrv16**.

iqunlspt

iqunlspt is a self-contained subset of the SAP Sybase IQ 12.7 (ESD #5) database engine. It runs as a background process and provides support for legacy database unloads. **iqunlspt** starts on your legacy database with these options as defaults:

```
iqunlspt -iqnotemp 100 -iqro 1 -c 48MB -gc 20 -gd
all -gk all -gl all -gm 1 -gu all -ti 4400 -x shmem .
```

If your database requires special switches or memory setting, **iqunlspt** accepts additional startup arguments. See the *Utility Guide*.

Default cache settings are sufficient for most migrations. At migration, data queries execute against the system catalogs, not IQ data, so the **iqunlspt** engine needs lower cache levels than complex queries or multiple concurrent users. The amount of time required to start the legacy database is the same as to start **iqunlspt**. This time is included in the **iqunload** startup time.

iqsrv16

iqunload starts **iqsrv16** with these options:

```
iqsrv16 -gp 4096 -c 40p -gc 20 -gd all -gk all -gl all
-gm 1 -gu all -qi -qs -ti 4400
```

iqsrv16 also includes the **-n** parameter followed by a special randomly generated server name. The **-c 40p** setting provides a larger cache for the catalog store, allowing the server engine to execute many schema DDL statements. Both server start commands use the default values for **-iqmc** and **-iqtc**. If the legacy server requires larger startup values, use the **-c** switch to increase the server cache memory.

See also

- *iqunload Utility* on page 28

- *iqlsunload Utility* on page 31

Migration Issues

SAP Sybase IQ no longer supports some legacy features. Run **iqunload** in schema unload mode to generate a script `(reload.sql)` that contains the entire database schema. Compare the contents of this file to find unsupported syntax and metadata.

Unsupported Objects

Check the schema for objects SAP Sybase IQ no longer supports.

Table 1. Unsupported metadata

Object	Details	Action
Invalid database, table, or user names	Table names cannot contain double quote characters. User names and database names cannot contain double quote characters, single quote, or semicolon characters. User names and database names cannot start or end with a space.	Change the object name.
Reserved logical server names	A logical server cannot be named ALL, AUTO, COORDINATOR, DEFAULT, OPEN, or SERVER.	Drop the logical server before upgrading.
Join indexes	Join indexes are no longer supported.**iqunload** does not run if the database to be migrated contains join indexes.	Drop all join indexes before migrating data.
LD indexes	LD indexes are no longer supported.	Drop all LD indexes before migrating data.
Database with BLANK PADDING OFF	**iqunload** searches BLANK PADDING OFF databases for any indexes that would become invalid after migration. **iqunload** fails and lists indexes and constraints that must be dropped and in which order.	Drop these indexes and constraints before the schema reloads and recreate the indexes and constraints after schema reload has been completed.
Unenforced constraints	**iqunload** fails and lists unenforced constraints that must be dropped.	Drop unenforced constraints before proceeding with migration.
Old 1-byte FP or old 2-byte FP indexes	Databases created with SAP Sybase IQ 12.4.2 or earlier may have these indexes. Because these indexes were automatically created by SAP Sybase IQ, you cannot drop and recreate them; you must rebuild them.	Allow **iqunload** to check for these and list them. Rebuild these indexes using **sp_iqrebuildindex** before migration.The rebuilt indexes are upgraded.

See also

- *Syntax Changes* on page 35
- *Schema Size* on page 36
- *Output Logs* on page 37
- *Data Storage Changes* on page 38
- *Post-Migration Files* on page 39

Syntax Changes

Review the reload script `(reload.sql)` for legacy syntax that can cause **iqunload** to fail.

Table 2. Troubleshooting Syntax Changes

Problem	Solution
A **DECLARE LOCAL TEMPORARY TABLE** statement in a procedure or trigger causes a syntax error if the table name is prefixed with an owner name.	Remove the owner name.
If a **CREATE TRIGGER** statement does not include an owner name for the table on which the trigger is defined, and the table must be qualified with an owner when referenced by the user executing the reload.sql file, the statement fails with a `‘table-name’ not found` error.	Prefix the table name with the owner name.

If an object name (such as a table, column, variable, or parameter name) corresponds to a reserved word introduced in a later version of SAP Sybase IQ, the reload fails. (For reserved words, see *Reference: Building Blocks, Tables, and Procedures*. For example:
`CREATE PROCEDURE p( )`
`BEGIN`
`DECLARE NCHAR INT;`
`SET NCHAR = 1;`
`END`
	Change all references to the reserved word to use a different name. For variable names, prefixing the name with @ is a common convention that prevents naming conflicts.

Views that use Transact-SQL® outer joins (by specifying *= or =*) may not be created properly when they are reloaded.	Add the following line to the reload script: `SET TEMPORARY OPTION tsql_outer_joins='on'` Also set this option for your database. Rewrite any views or stored procedures that use Transact–SQL outer joins.
Stored procedures that use Transact–SQL outer joins may not work correctly.	Rewrite views and stored procedures.

Problem	Solution
Functions that have `OUT` or `INOUT` parameters cannot be reloaded.	`OUT` and `INOUT` parameters are no longer supported. Drop these functions before reloading.

See also

- *Unsupported Objects* on page 34
- *Schema Size* on page 36
- *Output Logs* on page 37
- *Data Storage Changes* on page 38
- *Post-Migration Files* on page 39

Schema Size

Increase the cache memory to migrate large and extremely large schemas.

Large Schemas

Default cache settings for large schemas may be too small and can exhaust dynamic memory in the **iqsrv16** server. Use the **-c** switch to increase the server cache memory and **-new_startline** to pass the switch to the server.

A diagnostic example includes these switches:

```
-ca 1
-c 1000m
-o /iq15outputdir/iq16console.out
```

- **-ca 1** – enables dynamic catalog cache sizing, and logs memory cache statistics to the console.
- **-c 1000m** – sets the initial catalog cache at 1GB.
- **-o /outputdir/iq16console.out** – specifies the log file for console output.

Use a text editor to xamine the `.out` file log entries. Watch how the catalog store adjusts the cache and determines if the setting is appropriate.

Note: The value shown for **-c** is in bytes. Set switches appropriately for your system. To specify megabytes, use the m suffix, as shown.

Extremely Large Schemas

Running **iqunload** in schema unload mode (**iqunload -n**) generates a single script (`reload.sql`) that includes the entire legacy schema. In some cases, you may need to break a very large `reload.sql` file into pieces that can be executed sequentially. This also helps the server manage the cache.

If **iqunload** fails in migration mode (**iqunload -au**) because dynamic memory is exhausted, set the cache settings as high as your hardware and operating system limitations allow. If the failure continues, contact SAP Sybase for assistance.

See also

- *Unsupported Objects* on page 34
- *Syntax Changes* on page 35
- *Output Logs* on page 37
- *Data Storage Changes* on page 38
- *Post-Migration Files* on page 39

Output Logs

Check the output logs to isolate migration problems.

SAP Sybase IQ 16.0 Engine Logs

Use the **-new_startline** " **-z -zr all**" argument to start **iqsrv16** with extra logging:

```
iqunload -au -c "UID=DBA;PWD=SQL;DBF=/iq-15/
unload/127/db/iq127db.db" -new_startline "-z -zr all"
-o iq15db.out
```

SAP Sybase IQ 12.7 Engine Logs

Use the **START** = **-z -zr all** argument to start the 12.7 engine with extra logging:

```
iqunload -v -au -c "UID=DBA;PWD=SQL;DBF=/iq-15/
unload/127/db/iq127db.db;START=-z -zr all
-o iq127db.out"
```

See the *Utility Guide* for details about the **-z** and **-zr all** parameters.

Server Not Found

A message similar to this indicates that **iqunload** started the database but could not connect to the server:

```
SQL error:Database server not found
```

Check to see if **iqunlspt** is running and stop the process before retrying **iqunload**.

For example, here is **top** output:

```
load averages:  1.45,  1.19,  0.80; up 3+16:22:31
10:2
172 processes: 168 sleeping, 2 zombie, 2 on cpu
CPU states: 79.1% idle, 18.9% user,  1.9% kernel,  0.0%
iowait,  0.0% swap
Memory: 16G phys mem, 13G free mem, 16G swap, 16G free
swap

PID USERNAME LWP PRI NICE  SIZE   RES STATE    TIME    CPU COMMAND
21223 ybrown 1  59    0 2908K 1844K cpu      0:00  0.12% top
21172 ybrown 476  59    0  319M  264M sleep    0:01  0.05% iqunlspt
24890 ybrown 14  29   10   79M   43M sleep    0:49  0.03% java
20103 ybrown 1  59    0 7916K 2080K sleep    0:00  0.00% sshd
```

To stop the process, enter the command **kill -9** and supply the process ID:

```
kill -9 21172
```

Trying to run **iqunload** without killing an orphaned **iqunlspt** process, may generate this error:

```
SQL error: Unable to start specified database: autostarting database
failed.
```

Obsolete Stored Procedures

Migration replaces 12.7 login procedures with new login management functions.

sp_login_environment replaces the 12.7 default login procedure **DBA.sp_iq_process_login** and **dbo.sa_post_login_procedure** replaces the 12.7 default post-login procedure **DBA.sp_iq_process_post_login**. **iqunload** generally replaces obsolete options with new defaults, but if the 12.7 option is set on a specific user instead of PUBLIC (the default), the log file may report errors:

```
E. 10/31 16:53:40. Login procedure
'DBA.sp_iq_process_login' caused SQLSTATE '52W09'
E. 10/31 16:53:40. Procedure 'sp_iq_process_login' not
found
```

See also

- *Unsupported Objects* on page 34
- *Syntax Changes* on page 35
- *Schema Size* on page 36
- *Data Storage Changes* on page 38
- *Post-Migration Files* on page 39

Data Storage Changes

SAP Sybase IQ 16.0 migration creates a new catalog store and changes some legacy options.

Dbspaces

In current versions of SAP Sybase IQ, all user data should reside in a user dbspace comprised of one or more files. Migration converts main dbspaces into files under one user dbspace: `IQ_MAIN`, for the SAP Sybase IQ main store, and temporary dbspaces into files under a single temporary dbspace, `IQ_SYSTEM_TEMP`, for a single SAP Sybase IQ temporary store. Existing catalog store dbspaces remain as dbspaces with a single file. All of the old main dbspaces become files in the new `iq_main` user main dbspace. Migration sets the `PUBLIC.default_dbspace` option to the value `iq_main`.

Logical names for files created from converted dbspaces are the dbspace name followed by an underscore and the file ID. For example, a main dbspace with file ID 16384 becomes `IQ_SYSTEM_MAIN_16384`.

Main Store

Migration creates a new system file for the `IQ_SYSTEM_MAIN` dbspace that contains no tables. By default, the name of this file is `new_main_store.iq`, but you can use the **ms_filename** argument to specify a different file name. The **iqunload** utility computes the size of the new `IQ_SYSTEM_MAIN` based on the size of your existing database.

If you accept the default settings for **iqunload -au -c**, the new store marked as `MAIN` has `DBSpaceName = IQ_SYSTEM_MAIN`, `DBFileName = IQ_SYSTEM_MAIN` and `path = new_main_store.iq`. For multiplex migration, the location of the new main store must be visible to all nodes on the multiplex, and you must use the **-ms_filename** argument to specify the path instead of the default value of `new_main_store.iq`.

When you migrate a database, specify the file to use for the new `IQ_SYSTEM_MAIN` dbspace, its name, whether or not to use a raw device, and the size of the main store and its reserve.

Migrating IQ_SYSTEM_MAIN

Specify the `IQ_SYSTEM_MAIN` size in the database migration command. The **-ms_size** parameter requires a value in MB, not GB. Omit **-ms_size** and **-ms_reserve** to specify a raw device. For a raw device, you must specify an unused raw partition.

This statement creates an `IQ_SYSTEM_MAIN` on a raw device:

```
iqunload -au -ms_filename /dev/rdsk/c1t0d1 -c
"UID=DBA;PWD=SQL;DBF=latest.db"
```

See also

- *Unsupported Objects* on page 34
- *Syntax Changes* on page 35
- *Schema Size* on page 36
- *Output Logs* on page 37
- *Post-Migration Files* on page 39

Post-Migration Files

iqunload generates a set of files derived from the legacy database. .

Table 3. Pre-Migration and Post-Migration Files

Pre-Migration	Post-Migration Files	Description
`asiqdemo.db`	`asiqdemo.db.before_schema_reload`	The 12.7 catalog database. This file is copied at the OS level upon successful migration; it is not a result of the SQL **backup** command.
`asiqdemo.log`	`asiqdemo.log`	The database log file is regenerated when the migrated database is used with the 16.0 server.
`asiqdemo.iq`	`asiqdemo.iq`	The old SAP Sybase IQ 12.7 `IQ_SYSTEM_MAIN` dbspace. This file and all other user dbspaces are unaffected by the migration process. This dbspace is added as a file to a user main dbspace.

Pre-Migration	Post-Migration Files	Description
`asiqdemo.iqtmp`	`asiqdemo.iqtmp`	The `IQ_SYSTEM_TEMP` dbspace. No operations are performed on this dbspace during migration. This file becomes the IQ 16.0 database temporary store.
`asiqdemo.iqmsg`	`asiqdemo.iqmsg.before_schema_reload`	The IQ 12.7 message file. This file is copied at the OS level upon successful migration.
	`asiqdemo.db`	The new 16.0 migrated catalog database.
	`new_main_store.iq`	The new `IQ_SYSTEM_MAIN` dbspace for the migrated database.

See also

- *Unsupported Objects* on page 34
- *Syntax Changes* on page 35
- *Schema Size* on page 36
- *Output Logs* on page 37
- *Data Storage Changes* on page 38

Unloading Legacy Schemas

To unload legacy schema, run **iqunload** in schema unload mode (**iqunload -n**) on the same machine as the legacy schema.

1. Copy these files from *$IQDIR16*`/unload` to *$ASDIR*`/scripts`:
   - `unloadold.sql`
   - `unload.sql`
   - `optdeflt.sql`
   - `opttemp.sql`
2. Start the legacy server.
3. Run **iqunload** in schema unload mode (**iqunload -n**).

   Include the appropriate connection parameters and other startup options. Schema unload mode creates a SQL script (`reload.sql`) in the current directory that contains the legacy database schema. `reload.sql` does not contain any checkpoints.For very large schemas, edit `reload.sql`, to add a few checkpoints. If you do not include extra checkpoints, IQ generates additional metadata objects that requires extra (**-iqmc**) main cache memory.

`reload.sql` also contains a **CREATE DATABASE** template command that is commented out.

4. Create a new 16.0 database.

   Set the IQ SIZE and TEMPORARY SIZE clauses to create an IQ_SYSTEM_MAIN of 10GB and IQ_SYSTEM_TEMP of 5GB. For example:

   ```
   CREATE DATABASE 'test.db'
   IQ PATH 'test.iq'
   IQ SIZE 10240
   TEMPORARY PATH 'test.iqtmp'
   TEMPORARY SIZE 5120
   ```
5. Start and connect to the new database.
6. Run the `reload.sql` against the new database.

   Execution time roughly approximates the actual time to allow for database migration, excluding validation checks. Correct any errors. Perform this process iteratively until you can cleanly load the legacy schema.

Migrating Legacy Databases

Run **iqunload** in database migration mode (**iqunload -au**) to migrate a legacy database.

Simplex Migration

Migrate a 12.7 database simplex database to 16.0.

1. *Migrating the Legacy Database*

   Make sure that the database file is not in use, and run the iqunload utility with the -au (migrate database) and -c (connection parameters).
2. *Verifying the Migrated Database*

   To verify simplex migration, start the migrated database in read-only mode and perform post-migration tasks.

Migrating the Legacy Database

Make sure that the database file is not in use, and run the **iqunload** utility with the **-au** (migrate database) and **-c** (connection parameters).

This command migrates the simplex database `mytest` and saves output in `unload.out` in the current directory:

```
iqunload -au -c "uid=DBA;pwd=SQL;dbf=mytest" -o unload.out
```

The database and the **iqunload** utility must be on the same machine to migrate the database, or **iqunload** returns an error. **-o** is an optional switch that sends a copy of the console output to the specified log file, here named `unload.out`.

Because the example specified `DBF=mytest.db`, the **iqunload** utility attempts to connect to this database in the current directory. You can also specify the full path to the database, as shown here:

```
iqunload -au -c
"dbf=/ybrown/iq-15/unload/iq127db.db;uid=DBA;pwd=SQL"
Output:
    Sybase IQ Unload Utility Version 15.0.0.5533
    Connecting and initializing
    Unloading user and group definitions
    Unloading table definitions
    Unloading index definitions
    Unloading functions
    Unloading view definitions
    Unloading procedures
    Unloading triggers
    Unloading SQL Remote definitions
    Creating new database
    Creating indexes for (1/14)
        "DBA"."sales_order"
```

```
    Creating indexes for (2/14)
        "DBA"."sales_order_items"
    Creating indexes for (3/14) "DBA"."contact"
    Creating indexes for (4/14) "DBA"."customer"
    Creating indexes for (5/14) "DBA"."fin_code"
    Creating indexes for (6/14) "DBA"."fin_data"
    Creating indexes for (7/14) "DBA"."product"
    Creating indexes for (8/14) "DBA"."department"
    Creating indexes for (9/14) "DBA"."employee"
    Creating indexes for (10/14)"DBA"."alt_sales_order"
    Creating indexes for (11/14) "DBA"."alt_sales_order_items"
    Creating indexes for (12/14) "DBA"."iq_dummy"
    Creating indexes for (13/14) "DBA"."emp1"
    Creating indexes for (14/14) "DBA"."sale"
```

```
Successfully backed up file "/ybrown/iq-15/
unload/127/db/iq127db.db" by renaming it to "/ybrown/iq-15/unload/
127/db/iq127db.db.before_schema_reload".
Successfully backed up file "/ybrown/iq-15/unload/127/db/
iq127db.iqmsg"
by renaming it to "/ybrown/iq-15/unload/127/
db/iq127db.iqmsg.before_schema_reload"
Successfully reloaded schema of database "/ybrown/iq-15/unload/127/
db/iq127db.db".
```

Perform post migration tasks. Make sure that the migration completed correctly. Back up your new databases.

Verifying the Migrated Database

To verify simplex migration, start the migrated database in read-only mode and perform post-migration tasks.

1. Start the 16.0 database in read-only mode:

   ```
   start_iq -iqro 1
   ```

When starting the coordinator in 16.0, use the same port as the 12.7 writer server.

2. Issue a **CHECKPOINT** command.
3. Run the 16.0 version of **sp_iqcheckdb** in verify mode:

   ```
   sp_iqcheckdb ('verify database')
   ```

   If you run the procedure from Interactive SQL, redirect output to a file by entering:

   ```
   dbisql -c "..." "sp_iqcheckdb ('verify database')" >& filename
   ```

   where “...” represents startup parameters for your database.
4. Issue a **COMMIT** statement.
5. Check **sp_iqcheckdb** results for errors.

   If there is an error, you can revert to the previous database as long as you do not restart the database in write mode. To revert to the 12.7 catalog , copy all the `.before_schema_reload` files to the same file without the `.before_schema_load file` extension.
6. After you perform the read-only checks, stop the database server and restart in write mode.

Note: For information on interpreting **sp_iqcheckdb** results and corrective action, see *Administration: Backup, Restore, and Data Recovery > System Recovery and Database Repair*.

Multiplex Migration

Migrate multiplex databases, performing all steps in sequence.

1. *Synchronizing the Multiplex Nodes*

   Check the SQL Remote and multiplex server log files for synchronization problems.
2. *Migrate Local Stores*

   To move the 12.7 local stores before migration, use iqlsunload.
3. *Start the Multiplex Write Server*

   To clean the internal state information, start the write server in single-node mode.
4. *Multiplex Migration Parameters*

   To migrate the multiplex, run iqunload with the appropriate parameters.
5. *Verifying the Migrated Multiplex Database*

   Verify the migrated database in read-only mode and correct any errors.
6. *Starting the Coordinator*

   Starting the multiplex coordinator in single-node mode (-iqmpx_sn) and read-only (-iqro) performs some initial database checks. For coordinators, the server must reset an identity cookie before you can use the multiplex.
7. *Manually Synchronize the Secondary Nodes*

To start the secondary nodes, install SAP Sybase IQ 16.0, then synchronize from the coordinator node. When you migrate a query node, it becomes a reader node.

8. *Start the Secondary Nodes*

   To start the secondary nodes, all nodes of the multiplex must be running.

9. *Set the Failover Node*

   After you migrate the multiplex data, connect to the coordinator, and set the failover node.

10. *Troubleshooting Multiplex Migration*

    If you cannot migrate your multiplex database, try this alternate method.

Synchronizing the Multiplex Nodes

Check the SQL Remote and multiplex server log files for synchronization problems.

1. Start the multiplex server.
2. Start SQL Remote on all multiplex nodes.

   Give the multiplex time to propagate any changes throughout the multiplex. To do this, look at the write server console log file and check that the events starting with `ev_iqmpx` have successfully executed. By default, the server console log file is created in `$ASDIR/logfiles`.

   For example:

   ```
   Now accepting requestsOS Available: 933096K, Working Set: 83988K,
   Cache Target: 11483K
   OS Available: 860680K, Working Set: 83996K, Cache Target: 11483K
   Next time for 'ev_iqmpxq2w' is 2008/11/23 22:03:00.000
   Next time for 'ev_iqmpxstatus' is 2008/11/23 22:03:00.000
   OS Available: 859232K, Working Set: 84112K, Cache Target: 11489K
   OS Available: 861052K, Working Set: 84424K, Cache Target: 11489K
   OS Available: 860972K, Working Set: 84428K, Cache Target: 11489K
   OS Available: 850248K, Working Set: 85540K, Cache Target: 11579K
   OS Available: 850104K, Working Set: 85568K, Cache Target: 11579K
   Next time for 'ev_iqmpxq2w' is 2008/11/23 22:04:00.000
   Next time for 'ev_iqmpxstatus' is 2008/11/23 22:04:00.000
   OS Available: 850120K, Working Set: 85600K, Cache Target: 11579K
   Next time for 'ev_iqmpxq2w' is 2008/11/23 22:05:00.000
   Next time for 'ev_iqmpxstatus' is 2008/11/23 22:05:00.000
   OS Available: 852668K, Working Set: 85604K, Cache Target: 11579K
   ```

3. Wait for SQL Remote to scan the log files, then view the logs.

   Wait for SQL Remote to process any messages:

   ```
   I. 11/23 22:06:10. Scanning logs starting at offset 0001787252
   I. 11/23 22:06:10. Hovering at end of active log
   ```

4. Shut down SQL Remote and multiplex servers.

   If you simply shut down the multiplex servers, the SQL Remote servers detect that the multiplex servers are no longer running and shut themselves down. By default, the SQL Remote servers should shut themselves down within 60 seconds.

5. Shut down query servers in the multiplex. They are no longer required.
6. If the logs report no errors, verify the database.

Migrate Local Stores

To move the 12.7 local stores before migration, use **iqlsunload**.

To migrate your local store, consolidate node-specific information into either the existing 12.7 writer or the new SAP Sybase IQ 16.0 main store. Customize the process to meet your data requirements.

For query nodes with node- or department-specific information, use tablespaces and partitioning to achieve the same results.

If information is duplicated across your query nodes, you may need to migrate only a single query server's local store. The duplicated information on the other query servers becomes redundant and can be ignored for multiplex migration.

See also

- *Start the Multiplex Write Server* on page 49

Partitioning Query Server Data

If the same table exists on multiple query nodes, and each node has its own subset of the data, manually edit the local store migration scripts.

For a department-specific `employee` table on each query server, follow these basic steps:

1. Unload the schema and data from the query nodes.

   The `reload_schema.sql` script produced for each query node contains the same schema definition for `employee`.
2. Execute the `reload_schema.sql` from one of the query nodes against either the existing 12.7 writer or the new SAP Sybase IQ 16.0 main store.
3. Execute the `reload_data.sql` script from each of the query nodes against the same server.

   This procedure creates the `employee` table once but loads each query node data set.

See also

- *Addressing Overlapping Query Server Data* on page 46
- *Moving Local Stores* on page 49

Addressing Overlapping Query Server Data

If the same table exists on each query node with overlapping data sets, you must resolve the issue. Extract the data files to ensure that data sets are unique, or rename the tables and then reload all the unique tables.

1. Run **iqlsunload** against all query servers with local stores that have the data to consolidate.
2. Modify the `reload_schema.sql` and `reload_data.sql` files to use the new table names. Do not modify `extract_data.sql;` it references the table found in the query server's local store.
3. Run `extract_data.sql` from each node.

The following example shows modifications to the **iqlsunload** output to carry out step 2. Suppose that the `reload_schema.sql` script contains:

```
CREATE TABLE "DBA"."sales_order"

    "id"                unsigned int NOT NULL  IQ UNIQUE (648),
    "cust_id"               unsigned int NOT NULL  IQ UNIQUE (111),
    "order_date"            "datetime" NOT NULL  IQ UNIQUE (376),
    "fin_code_id"           char(2) NULL  IQ UNIQUE (1),
    "region"                char(7) NULL  IQ UNIQUE (5),
    "sales_rep"             unsigned int NOT NULL  IQ UNIQUE (75),
PRIMARY KEY ("id"),
```

Modify `reload_schema.sql` to:

```
CREATE TABLE "DBA"."q1_sales_order"

    "id"                unsigned int NOT NULL  IQ UNIQUE (648),
    "cust_id"               unsigned int NOT NULL  IQ UNIQUE (111),
    "order_date"            "datetime" NOT NULL  IQ UNIQUE (376),
    "fin_code_id"           char(2) NULL  IQ UNIQUE (1),
    "region"                char(7) NULL  IQ UNIQUE (5),
    "sales_rep"             unsigned int NOT NULL  IQ UNIQUE (75),
    PRIMARY KEY ("id"),
```

`extract_data.sql` contains:

```
---- Extract Table Data for table sales_order
-- NOTE: Approximately 57672 bytes of storage space.
-- will be required to extract the data for this table.
--
-- The following will unload the data for table
sales_order, row group 1, column group 1
```

```
SET TEMPORARY OPTION temp_extract_name1 =
'DBA_sales_order_1_1_DATA_1.inp';
SET TEMPORARY OPTION temp_extract_name2 =
'DBA_sales_order_1_1_DATA_2.inp';
SET TEMPORARY OPTION temp_extract_name3 =
'DBA_sales_order_1_1_DATA_3.inp';
SET TEMPORARY OPTION temp_extract_name4 =
'DBA_sales_order_1_1_DATA_4.inp';
SET TEMPORARY OPTION temp_extract_name5 =
'DBA_sales_order_1_1_DATA_5.inp';
SET TEMPORARY OPTION temp_extract_name6 =
'DBA_sales_order_1_1_DATA_6.inp';
SET TEMPORARY OPTION temp_extract_name7 =
'DBA_sales_order_1_1_DATA_7.inp';
SET TEMPORARY OPTION temp_extract_name8 =
'DBA_sales_order_1_1_DATA_8.inp';

SELECT id, cust_id, order_date,
IFNULL(fin_code_id, @null_string, fin_code_id),
IFNULL(region, @null_string, region), sales_rep
FROM "DBA"."sales_order"
WHERE rowid( "sales_order" ) >= 1
AND  rowid( "sales_order" ) <= 648;
```

```

SET TEMPORARY OPTION temp_extract_name1 = '';
SET TEMPORARY OPTION temp_extract_name2 = '';
SET TEMPORARY OPTION temp_extract_name3 = '';
SET TEMPORARY OPTION temp_extract_name4 = '';
SET TEMPORARY OPTION temp_extract_name5 = '';
```

Leave `extract_data.sql` code unchanged to extract the `sales_order` table from the query server.

Suppose that `reload_data.sql` contains:

```
-- Reload Table Data for table "sales_order"
-------------------------------------------------
ALTER TABLE "DBA"."sales_order" MODIFY cust_id NULL;
ALTER TABLE "DBA"."sales_order" MODIFY order_date NULL;
ALTER TABLE "DBA"."sales_order" MODIFY sales_rep NULL;

SET @max_row_id =
( SELECT MAX( rowid( "sales_order" ) )+1
FROM "DBA"."sales_order" );
```

```
SET @load_statement =
'LOAD TABLE "DBA"."sales_order"
(id, cust_id, order_date, fin_code_id NULL(
'''||@null_string||''' ) , region NULL(
'''||@null_string||''' ) , sales_rep)
FROM
'''||@extract_directory||'DBA_sales_order_1_1_DATA_1.
inp'',
'''||@extract_directory||'DBA_sales_order_1_1_DATA_2.
inp'', '''||@extract_directory||'DBA_sales_order_1_1_DATA_3.
inp'',
'''||@extract_directory||'DBA_sales_order_1_1_DATA_4.
inp'',
'''||@extract_directory||'DBA_sales_order_1_1_DATA_5.
inp'', '''||@extract_directory||'DBA_sales_order_1_1_DATA_6.
inp'', '''||@extract_directory||'DBA_sales_order_1_1_DATA_7.
inp'', '''||@extract_directory||'DBA_sales_order_1_1_DATA_8.
inp'' ROW DELIMITED BY ''\n'' QUOTES ON
```

```
ESCAPES OFF DEFAULTS OFF FORMAT ASCII
IGNORE CONSTRAINT ALL 0 START ROW ID
'||@max_row_id;
```

```
CALL IqExecuteCommand( @load_statement );
ALTER TABLE "DBA"."sales_order" MODIFY cust_id NOT
NULL;
ALTER TABLE "DBA"."sales_order" MODIFY order_date NOT
NULL;
ALTER TABLE "DBA"."sales_order" MODIFY sales_rep NOT
NULL;
```

Change `reload_data.sql` to:

```
-- Reload Table Data for table
"q1_sales_order"
-------------------------------------------------
ALTER TABLE "DBA"."q1_sales_order" MODIFY cust_id NULL;
ALTER TABLE "DBA"."q1_sales_order" MODIFY order_date
NULL;
ALTER TABLE "DBA"."q1_sales_order" MODIFY sales_rep
NULL;

SET @max_row_id = ( SELECT MAX( rowid( "q1_sales_order"
) )+1 FROM "DBA"."q1_sales_order" );
```

```
SET @load_statement =
'LOAD TABLE "DBA"."q1_sales_order"
(id, cust_id, order_date, fin_code_id NULL(
'''||@null_string||''' ) , region NULL(
'''||@null_string||''' ) , sales_rep) FROM
'''||@extract_directory||'DBA_q1_sales_order_1_1_DATA_
1.inp'',
'''||@extract_directory||'DBA_q1_sales_order_1_1_DATA_
2.inp'',
'''||@extract_directory||'DBA_q1_sales_order_1_1_DATA_
3.inp'',
'''||@extract_directory||'DBA_q1_sales_order_1_1_DATA_
4.inp'', '''||
@extract_directory||'DBA_q1_sales_order_1_1_DATA_5.inp'', '''||
@extract_directory||'DBA_q1_sales_order_1_1_DATA_
6.inp'', '''||@extract_directory||'DBA_q1_sales_order_1_1_DATA_
7.inp'', '''||@extract_directory||'DBA_q1_sales_order_1_1_DATA_
8.inp'' ROW DELIMITED BY ''\n'' QUOTES ON ESCAPES OFF
DEFAULTS OFF FORMAT ASCII IGNORE CONSTRAINT ALL 0
START ROW ID '||@max_row_id;
```

```
CALL IqExecuteCommand( @load_statement );ALTER TABLE
"DBA"."q1_sales_order" MODIFY cust_id NOT
NULL;
ALTER TABLE "DBA"."q1_sales_order" MODIFY order_date
NOT NULL;
ALTER TABLE "DBA"."q1_sales_order" MODIFY sales_rep NOT
NULL;
```

This example shows query server schema and data that require intervention during migration. Your situation may vary, but you have complete control of the content of the final `reload_schema.sql` and `reload_data.sql` files.

See also

- *Partitioning Query Server Data* on page 45
- *Moving Local Stores* on page 49

Moving Local Stores

Unload and move the 12.7 local stores.

Prerequisites

Upgrade to SAP Sybase IQ 12.7 ESD #5 or later.

Task

1. Source the `ASIQ-12_7.sh` or `.csh` file.
2. Run the 12.7 **iqlsunload** utility against each query server with a local store.
3. Edit `reload_schema.sql`:
   - Delete unwanted objects.
   - Change any commented objects in the `reload_schema.sql` that you want to reload.
   - Add commands to define any objects that you defined in **sp_mpxcfg_**`<servername>` procedures.
4. Edit `extract_data.sql` to remove objects you do not want to migrate. These objects are generally the same ones you removed from `reload_schema.sql`.
5. Use Interactive SQL to run `extract_data.sql` from your 12.7 local store.

   You now have unloaded the schema and data for local objects in the 12.7 local store.
6. Run the `reload_schema.sql` and `reload_data.sql` scripts against the 12.7 write server.

   Note: If you prefer, wait until the write server has been migrated to version 16.0 and run `reload_schema.sql` and `reload_data.sql` against the new coordinator.

See also

- *Partitioning Query Server Data* on page 45
- *Addressing Overlapping Query Server Data* on page 46

Start the Multiplex Write Server

To clean the internal state information, start the write server in single-node mode.

Note: You must specify your login and password as arguments to the start_server script.

Start the writer node with the server arguments **-gm 1** and **-iqmpx_sn 1**:

```
-gm 1 -iqmpx_sn 1
```

If you use administrative startup scripts, create a copy of the `start_server.sh` script to start the write server you want to migrate. For example, copy the existing file `start_server.sh` to a new file called `start_server_single_node.sh`.

Suppose that `start_server.sh` contains this startup command:

```
start_asiq -STARTDIR /work/iq-127/mpx/main @/work/iq-
127/mpx/main/params.cfg -n mpx_main $readonly $nomain -
x tcpip{port=62631} /work/iq-127/mpx/main/main.db
$dbkey
```

Add the two single node startup arguments to change the preceding command as follows in `start_server_single_node.sh`:

```
start_asiq -STARTDIR /work/iq-127/mpx/main @/work/iq-
127/mpx/main/params.cfg -n mpx_main -gm 1 -iqmpx_sn 1
$readonly $nomain -x tcpip{port=62631} /work/iq-127/
mpx/main/main.db $dbkey
```

There are now two script files, `start_server.sh` and `start_server_single_node.sh` to make the server ready for migration:

1. Start the writer node with `start_server_single_node.sh`.
2. Shut down the writer node.
3. Start the writer node with `start_server.sh`.
4. Shut down the writer node.
5. Shut down the SAP Sybase IQ 12.7 server.

See also

- *Migrate Local Stores* on page 45

Multiplex Migration Parameters

To migrate the multiplex, run **iqunload** with the appropriate parameters.

Minimum required parameters for a multiplex writer are **-au** (migrate database), **-c** (connection parameters), **ENG=** connection parameter and **-ms_filename**. The **ENG=** value must match the existing server name in SAP Sybase IQ 12.7, and the **-ms_filename** specifies the new main store for the migrated writer. This path must be the same for all nodes in the multiplex.

There are two differences in the way you will execute the **iqunload** utility for multiplex:

- Specify the engine name in the **-c** connection parameters. This is the same name that your `<mpx_dir>/<writer_node>/start_server` script file uses to start the writer node. The **iqunload** utility initially attempts to start the database server as simplex. This start requires the name of the server to match the naming conventions for the multiplex nodes. Once **iqunload** detects that the server is a multiplex node, it shuts the node down and restarts it using the **-iqmpx_sn 1** option.

- The name of the new main store must be visible and accessible by all nodes of the multiplex. This is important because the main store file name defaults to `new_system_main.iq`, and its location is relative to the catalog database file (.db). Later, when you synchronize the SAP Sybase IQ 16.0 multiplex, the catalog is replicated to the secondary nodes, formerly known as the query nodes. If you leave the default value for the main store name unchanged, the path remains `new_system_main.iq` and secondary nodes cannot find the shared main store.

For multiplex writers, required arguments are:

- **ENG** – argument specifies the multiplex main engine name. **iqunload** attempts to start the database and determine whether the database is a simplex or multiplex. If multiplex, the server name is enforced. If you are unsure of the server name, check the administrative script `start_server` in the database directory.
- **DBF** – argument must specify the actual path used to create the multiplex. If you are unsure of this, look at the `SYSIQFILE` table in your 12.7 server to verify the database path.
- **-ms_filename** – argument specifies the location of the new main store. This path must be visible and accessible by all servers in the multiplex.

For example:

```
iqunload -au -v -c
"uid=DBA;pwd=SQL;dbf=/sunx5prod/users/marshall/mpx127/
w1/w1.db;eng=w1_1234" -ms_filename
../shared/new_main_store.iq
```

```
 Sybase IQ Unload Utility Version 15.2.0.5533
Connecting and initializing
    2008-11-23 22:32:07 Unloading user and group
        definitions
    2008-11-23 22:32:08 Unloading table definitions
    2008-11-23 22:32:09 Unloading index definitions
    2008-11-23 22:32:09 Unloading functions
    2008-11-23 22:32:09 Unloading view definitions
    2008-11-23 22:32:09 Unloading procedures
    2008-11-23 22:32:09 Unloading triggers
    2008-11-23 22:32:09 Unloading SQL Remote         definitions
    2008-11-23 22:32:09 Unloading MobiLink definitions
    2008-11-23 22:32:10 Creating new database
    2008-11-23 22:32:48 Reloading user and group
       definitions    2008-11-23 22:32:48 Reloading table definitions
    2008-11-23 22:32:53 Reloading index definitions
    2008-11-23 22:32:53 Reloading functions
    2008-11-23 22:32:53 Reloading view definitions
    2008-11-23 22:32:53 Reloading procedures
    2008-11-23 22:32:53 Reloading triggers
    2008-11-23 22:32:53 Reloading SQL Remote
        definitions
    2008-11-23 22:32:53 Reloading MobiLink definitions
```

```
Successfully backed up file "/sunx5prod/users/marshall/mpx127/w1/
w1.db" by
```

```
renaming it to
"/sunx5prod/users/marshall/mpx127/w1/w1.db.before_schema_reload".
Successfully backed up file
"/sunx5prod/users/marshall/mpx127/main.db" by renaming it to
"/sunx5prod/users/marshall/mpx127/main.db.before_schema_reload".
Successfully backed up file
/sunx5prod/users/marshall/mpx127/main.iqmsg" by renaming it to
"/sunx5prod/users/marshall/mpx127/main.iqmsg.before_schema_reload".
Successfully reloaded schema of database
"/sunx5prod/users/marshall/mpx127/main.db".
```

Verifying the Migrated Multiplex Database

Verify the migrated database in read-only mode and correct any errors.

1. Start the database using the read-only switch, **-iqro 1**. Start the coordinator (the 12.7 write server) using both **-iqro 1** and single node mode, **-iqmpx_sn 1**.

   When starting the coordinator in 16.0, use the same port used by the 12.7 writer server.
2. Issue a **CHECKPOINT** command.
3. Run **sp_iqcheckdb** in verify mode:

   ```
   sp_iqcheckdb 'verify database'
   ```
4. Issue a **COMMIT** statement.

The server is currently in read-only mode, and cannot complete some post migration tasks. Additionally, the verification reports some problems with Block Count Mismatch, Blocks Leaked, and Unallocated Blocks in Use. No other segments of the verify database should report any errors.

For example:

```
'** Block Count Mismatch','79','*****'
'** Blocks Leaked','25','*****'
'** Unallocated Blocks in Use','104','*****'
```

Examine the **sp_iqcheckdb** report for errors. If you need to contact SAP Sybase Technical Support, you must provide the output from **sp_iqcheckdb**.

Starting the Coordinator

Starting the multiplex coordinator in single-node mode (**-iqmpx_sn**) and read-only (**-iqro**) performs some initial database checks. For coordinators, the server must reset an identity cookie before you can use the multiplex.

Once you successfully restart the coordinator with **iqro 1** and **iqmpx_sn 1**, shut it down and restart it without any special switches.

For example:

```
start_iq @params.cfg -n mpx_main -iqmpx_ov 1 -x 'tcpip{port=62631}' /
workserver/work/iq-127/mpx/main.db
```

Manually Synchronize the Secondary Nodes

To start the secondary nodes, install SAP Sybase IQ 16.0, then synchronize from the coordinator node. When you migrate a query node, it becomes a reader node.

1. Back up the query node files. Back up existing catalog `.db`, catalog `.log` and `iqmsg` files.

   For example:

   ```
   rename /sunx5prod/users/work/iq-127/mpx/q1/q1.db
   /sunx5prod/users/work/iq-127/mpx/q1/q1.db.before_schema_reload
   rename /sunx5prod/users/work/iq-127/mpx/q1/q1.log /sunx5prod/
   users/work/iq-127/mpx/q1/q1.log.before_schema_reload
   rename /sunx5prod/users/work/iq-127/mpx/q1/q1.iqmsg
   /sunx5prod/users/work/iq-127/mpx/q1/q1.iqmsg.before_schema_reload
   ```

2. Issue a **dbbackup** command to synchronize servers. You might have a different name for the query node's catalog file, depending on your configuration. In the following example, `q1.db` is the catalog file name on the query node:

   ```
   dbbackup -y -x -c
   "uid=dba;pwd=sql;eng=mpx_main;dbf=/sunx5prod/users/
   work/iq-127/mpx/main/main.db"
   /sunx5prod/users/work/iq-127/mpx/q1
   ```

   ```
   SQL Anywhere Backup Utility Version 11.0.1.5533 Debug
   (702 of 699 pages, 100% complete)
   Transaction log truncated
   Database backup completed
   ```

3. If your query nodes do not use a different catalog database name, skip to step 4.

   Step 2 synchronizes the catalog database file from the coordinator. If you prefer to use the same catalog database file name as the coordinator, adjust any server start and stop administration scripts on the secondary nodes to use the new name.

   To retain the same catalog database file names:

   - Rename the synchronized coordinator catalog database file name. For example, assuming the coordinator file was called `main.db` and the secondary server was called `q1.db`, enter:

     ```
     mv main.db q1.db
     ```

     ```
     rename main.db q1.db
     ```

   - Rename the log file for the query node. This is necessary as the file renamed above still contains an internal pointer to `main.log`:

     ```
     dblog -t q1.log q1.db
     ```

4. Start the secondary server in normal mode:

   ```
   start_iq @params.cfg -n mpx_q1 -x
   'tcpip{port=62632}' -o /worksrver/work/
   iq-127/mpx/q1/o.out -Z -zr all -zo /worksrver/
   ```

```
iq-127/mpx/q1/zo.out /workserver/work/
iq-127/mpx/q1/main.db
```

The above command line is derived from your existing query server `start_server` administration script.

5. Repeat these steps on the remaining secondary nodes that you want to migrate.

Start the Secondary Nodes

To start the secondary nodes, all nodes of the multiplex must be running.
Start the secondary servers with the command line startup utility.

For example:

```
start_iq @params.cfg -n <server_name> database_file.db
```

Where <*server_name*> specifies the secondary server. You can obtain the name from the existing start server administration script. The specified `database_file.db` is the name resulting after you performed the secondary node synchronization.

Set the Failover Node

After you migrate the multiplex data, connect to the coordinator, and set the failover node.
Use a command like this to set the failover node:.

```
ALTER MULTIPLEX SERVER servername ASSIGN AS FAILOVER SERVER
```

Where *servername* is one of the secondary nodes.

Troubleshooting Multiplex Migration

If you cannot migrate your multiplex database, try this alternate method.

- Drop all query nodes, to change the SAP Sybase IQ 12.7 multiplex to a simplex database.
- Follow the steps for simplex databases to migrate the database to SAP Sybase IQ 16.0.
- Convert the simplex SAP Sybase IQ 16.0 database to multiplex, following the steps in *Administration: Multiplex > Create Multiplex Servers > Converting Databases to Multiplex*.

Postmigration Tasks

SAP Sybase IQ 16 databases upgraded from SAP Sybase IQ 12.7 are initially set to run in SAP Sybase IQ 15.x compatibility mode. To complete the change from 15.x to 16.0, you must explicitly change several 15.x compatibility settings to complete the 16.0 upgrade.

Indexes

- In Fast Projection (`FP`) indexes, continuous `NBit` dictionary compression replaces `FP(1)`,`FP(2)`, and `FP(3)` byte dictionary compression. `FP(1)`,`FP(2)`, and `FP(3)` indexes roll over to `NBit(8)`,`NBit(16)`, and `NBit(24)` respectively. All data types except LOB (both character and binary) and BIT data types may be **NBit** columns.

If FP_NBIT_IQ15_COMPATIBILITY is OFF, IQ UNIQUE determines whether the column loads as Flat FP or NBit. Setting IQ UNIQUE to 0 loads the column as Flat FP. Columns without an IQ UNIQUE constraint load as NBit up to the NBit auto-sizing limits.

- New tiered HG index structure decouples load performance from HG index size. In SAP Sybase IQ 15, load throughput could degrade as the amount of data in an HG index increased. As the index grew, loading the same amount of data could take more time. The new tiered structure decouples load performance from the HG index size to increase throughput.

  The CREATE_HG_WITH_EXACT_DISTINCTS option determines whether newly created HG indexes are tiered or non-tiered. If this option is ON, all new HG indexes are non-tiered. To take advantage of the new structure, set this option to OFF. Use **sp_iqrebuildindex** to convert non-tiered HG indexes to tiered HG and vice-versa .

Constraints

Constraint	Description
IQ UNIQUE	In SAP Sybase IQ 16.0, IQ UNIQUE explicitly defines the expected cardinality of a column and determines whether the column loads as Flat FP or NBit. Columns retain their IQ UNIQUE(*n*) value during a 15.x to 16.0 database upgrade. Setting IQ UNIQUE to 0 loads the column as Flat FP. Columns without an IQ UNIQUE constraint or columns with an IQ UNIQUE *n* value less that is less than the limit defined by the FP_NBIT_AUTOSIZE_LIMIT option is not necessary. Auto-size functionality automatically sizes all low or medium cardinality columns as NBit. Use IQ UNIQUE in cases where you want to where you want to load the column as Flat FP or when you want to load as NBit and the number of distinct values exceeds the auto-size limits.

Options

Option	Description
FP_NBIT_IQ15_COMPATIBILITY	Provides tokenized **FP** support similar to that available in 15.x. This option is ON by default in all 16.0 databases upgraded from 15.x and OFF in all newly created 16.0 databases. • If this option is ON, the database engine uses the `MINIMIZE_STORAGE`, `FP_LOOKUP_SIZE`, and `FP_LOOKUP_SIZE_PPM` options to optimize column storage. These options are ignored in 16.0. • If this option is OFF, the database engine ignores 15.x options and columns conform to SAP Sybase IQ `NBit` storage options. Set this option to OFF to take advantage of `NBit` column compression.
CREATE_HG_WITH_EXACT_DISTINCTS	Determines whether new `HG` indexes explicitly created with a **CREATE INDEX** command, or implicitly creating or altering a table with a PRIMARY KEY or a FOREIGN KEY declaration, are tiered or non-tiered. This option is ON 16.0 databases upgraded from 15.x and all newly created 16.0 databases. If this option is ON, all new `HG` indexes are non-tiered. To take advantage of the new structure, set this option to OFF. To take advantage of the new tiered structure, set this option to OFF. Use **sp_iqrebuildindex** to convert non-tiered `HG` indexes to tiered `HG` and vice-versa.
REVERT_TO_V15_OPTIMIZER	`REVERT_TO_V15_OPTIMIZER` forces the query optimizer to mimic SAP Sybase IQ 15.x behavior. `REVERT_TO_V15_OPTIMIZER='ON'` by default in all 16.0 databases upgraded from 15.x. `REVERT_TO_V15_OPTIMIZER='OFF'` by default in all newly created SAP Sybase IQ 16.0 databases. If you plan to use SAP Sybase IQ hash partitioning features, set the REVERT_TO_V15_OPTIMIZER ='OFF' in databases upgraded from 15.x to SAP Sybase IQ.

Object Names

Reserved words cannot be used as object names.

A SAP Sybase IQ 15.x database could contain tables, columns, and other objects named row. In SAP Sybase IQ 16.0, row is a reserved word and cannot be used as an object name.

To use a reserved word as an object name, enclosed the object name in brackets (regardless of the `QUOTED_IDENTIFIER` setting) or double quotes (if `QUOTED_IDENTIFIER='ON'` [default]):

```
// QUOTED_IDENTIFIER ON | OFF
select * from [row];
alter table row2 rename [row] to col_row;

// QUOTED_IDENTIFIER='ON'
select "row" from row2;
alter table "row" rename rownew;
```

Stored Procedures

Use these stored procedures to review and change column indexes and constraints:

Procedure	Description
sp_iqcolumnmetadata	Returns index metadata for all columns in one or more tables.
sp_iqindexmetadata	Returns details about column indexes, including the index types (`Flat FP`, `NBit`, `HG`, and tiered `HG`), distinct counts, `IQ UNIQUE` *n* value, and `NBit` dictionary size.

Procedure	Description
sp_iqrebuildindex	Rebuilds `FP` indexes (`Flat FP` as `NBit`, or `NBit` as `Flat FP`) and `HG` indexes (single `HG` as tiered `HG`, or tiered `HG` as single `HG`). Before you can insert or update new data, you must rebuild all columns greater than 255 bytes wide. The `index_clause` can reset `IQ UNIQUE` *n* to an explicit value from 0 (to recast an `NBit` column to `Flat FP`) up to the limits defined in the `FP_NBIT_AUTOSIZE_LIMIT` and `FP_NBIT_LOOKUP_MB` options. **sp_iqrebuildindex** also enables read-write access to columns that contain large object (LOB) data. LOB columns migrated from 15.x databases are read-only until you run **sp_iqrebuildindex**. The estimated cardinality for `NBit` columns with an `IQ UNIQUE` value below or equal to the `FP_NBIT_AUTOSIZE_LIMIT` is stored as 0 regardless of the `FP_NBIT_IQ15_COMPATIBILITY` setting. This affects the value returned from **sp_iqindexmetadata**.
sp_iqindexrebuildwidedata	Identifies wide columns that you must rebuild before they are available for read/write activities. **sp_iqindexrebuildwidedata** also generates a list of statements that you can use to to rebuild the columns. This applies to CHAR, VARCHAR, BINARY, and VARBINARY columns wider than > 255 characters, as well as all Long Varchar and Long Binary columns.

Re-create Indexes for EUC_TAIWAN Data

In SAP Sybase IQ 15 and later, the character encoding specification for the EUC–TAIWAN collation now uses the EUC_TW character set. You must re-create indexes on data in version 12.7 or earlier databases that use the EUC_TAIWAN collation to make them work with SAP Sybase IQ 16.

Update Configuration Files

Compare your existing `params.cfg` files with the new `default.cfg` file created by the installation. The installation does not update or overwrite existing `params.cfg` files. In each `params.cfg` file, update any parameter defaults that differ from those in the `default.cfg` file, while maintaining any customized parameter settings that are appropriate for your system. Add any new startup parameters in `default.cfg` to your `params.cfg` file. The **-gl** parameter, for example, is required for server startup in version 12.5 and later.

Preserve Database Options

SAP Sybase IQ preserves the settings of all 12.7 database options that are still valid in migrated databases. Check for deprecated features.

Back Up Your Databases

- Back up your databases again with the **BACKUP** statement. If you use the **BACKUP** statement instead of a system–level backup, you can run backups and queries concurrently.
- For a multiplex migration, back up only the coordinator only in this manner. For secondary servers, run the **dbbackup** utility from the secondary server directory.

Additional Information

- *Administration: Database > Index SAP Sybase IQ Columns > Index Types Comparison > Fast Projection (FP) Index*
- *Administration: Database > Index SAP Sybase IQ Columns > Index Types Comparison > High_Group (HG) Index*
- *Reference: Statements and Options > SQL Statements > ALTER TABLE*
- *Reference: Statements and Options > Database Options > Alphabetical List of Options > FP_NBIT_IQ15_ COMPATIBILITY_MODE*
- *Reference: Statements and Options > Database Options > Alphabetical List of Options > CREATE_HG_WITH_EXACT_DISTINCTS*
- *Reference: Building Blocks, Tables, and Procedures > System Procedures > Alphabetical List of System Stored Procedures > sp_iqindexmetadata*
- *Reference: Building Blocks, Tables, and Procedures > System Procedures > Alphabetical List of System Stored Procedures > sp_iqrebuildindex*

Upgrading to Role-Based Security

Role-based security replaces the authority-based security model used in versions of SAP Sybase IQ earlier than 16.0.

What Happened to Authorities, Permissions, and Groups?

SAP Sybase IQ 16.0 introduces a role-based security model. Whereas before you had authorities, permissions, object-level permissions, and groups, you now have roles, system privileges, object-level privileges, and user-extended roles.

Note: You can use a SAP Sybase IQ 16.0 database server with a pre-16.0 database. When you do, full backwards compatibility is provided for that database, and its security model is not changed.

In pre-16.0 databases, authorities were database-level permissions. For example, a user with BACKUP authority could back up the database. Some authorities also bundled object-level permissions. For example, a user with PROFILE authority could perform application profiling and database tracing tasks, which involve using system procedures that aren't otherwise available for use. You could not create new authorities, alter the permissions they comprised, or drop them. You could grant administrative rights (WITH GRANT), but could not limit the grant to only being an administrator.

Now, roles replace authorities in functionality with the added benefit that you can create new roles, alter the privileges they comprise, and drop them. Switching to roles and privileges means you have more granular control over the privileges you want to grant to a user, and an easier way to grant them to other users. You can also grant the role to a user with administrative rights only, which means the user can grant and revoke the role, but cannot exercise the underlying privileges.

In pre-16.0 databases, permissions allowed you to create, modify, query, use, or delete database objects such as tables, views, and users. For example, you might have SELECT privilege on a table.

Now, privileges replace permissions in functionality, with the added benefit that there are far more privileges than permissions. For every privileged operation that can be performed on a database object, there is a grantable privilege. You can grant privileges individually to users, or grant a role to them. The term permission has not gone away; however, it has changed slightly. Previously, the word permission meant a grantable capability. Now, the word permission means the result of an evaluation of whether an operation can be performed. For example, you have permission to alter the table if you are the owner or you have the ALTER ANY TABLE system privilege.

In pre-16.0 databases, groups were a collection of one or more users whose authorities and permissions were determined by what is set at the group level. A user was granted group status, and then other users were granted membership in that group.

Now, the group paradigm is achieved using user-extended roles. If you have a user with a set of privileges that you want to grant to other users, you can extend the user to become a user-extended role, and then grant that role to other users.

When you upgrade a pre-16.0 database, the upgrade process automatically converts your existing authority, permission, and group hierarchy into an equivalent role, privilege, and user-extended role hierarchy. For every pre-16.0 authority, there is a compatibility role. These roles are easily identifiable in the database because their names start with SYS_AUTH. Compatibility roles contain the system privileges required for pre-16.0 users to perform the same operations they could perform using an authority.

To take full advantage of the control and granularity of privileges available with role-based security, it is strongly recommended that you review the compatibility role grants of each user post-migration and adjust membership and system privilege grants as necessary.

Authorities Become Compatibility Roles

When you upgrade a database, users that were granted authorities in pre-16.0 databases are automatically granted an equivalent compatibility role for that authority. If a user had the ability to administer the previous authority, the user has the ability to administer the compatibility role.

For ease of transition, the naming convention for each compatibility role retains the original authority name, but prefaces it with "SYS_AUTH_" and suffixes it with "_ROLE". For example, the authority BACKUP becomes the role SYS_AUTH_BACKUP_ROLE, authority RESOURCE becomes role SYS_AUTH_RESOURCES_ROLE, and so on.

You cannot modify compatibility roles. However, you can migrate them to a user-defined role, and then modify them. Once each underlying system privilege has been granted to at least one other role, you can drop the original compatibility role. When you migrate a compatibility role to a user-defined role, all users that were granted the compatibility role are automatically granted the new user-defined role. The compatibility role is automatically dropped once it has been migrated. However, you can restore compatibility roles using the CREATE ROLE statement.

Backwards compatibility for SQL statements has been provided so applications that grant or revoke authorities continue to work. However, the old syntax is deprecated and you should consider changing your applications to use the new SQL syntax for roles.

The following table shows authorities and the compatibility roles they become when a database is upgraded.

Pre-16.0 Authority	Equivalent Role	Description
BACKUP authority	SYS_AUTH_BACKUP_ROLE compatibility role	Allows a user to back up databases and transaction logs with archive or image backups by using the BACKUP statement or dbbackup utility.
DBA authority	SYS_AUTH_DBA_ROLE compatibility role SYS_AUTH_SA_ROLE compatibility role SYS_AUTH_SSO_ROLE compatibility role	Allows users to perform all possible privileged operations. Users with the SYS_AUTH_DBA_ROLE role can create database objects and assign ownership of these objects to other user IDs, change table structures, create new user IDs, revoke permissions from users, back up the database, and so on. Of the possible privileged operations that the SYS_AUTH_DBA_ROLE compatibility role can perform, the SYS_AUTH_SA_ROLE compatibility role allows the user to perform all database administration-related activities, such as creating tables, and backing up data. Of the possible privileged operations that the SYS_AUTH_DBA_ROLE compatibility role can perform, the SYS_AUTH_SSO_ROLE compatibility role allows the user to perform the security and access-related administration activities, such as creating users, and granting privileges on objects.
PROFILE authority	SYS_AUTH_PROFILE_ROLE compatibility role	Allows a user to perform profiling, tracing, and diagnostic operations.
READCLIENTFILE authority	SYS_AUTH_READCLIENTFILE_ROLE compatibility role	Allows a user to read files on the client computer, for example when loading data from a file on a client computer.
READFILE authority	SYS_AUTH_READFILE_ROLE compatibility role	Allows a user to use the OPENSTRING clause in a SELECT statement to read a file.
REMOTE DBA authority	SYS_RUN_REPLICATION_ROLE system role SYS_REPLICATION_ADMIN_ROLE system role	Allows a SQL Remote user to perform replication activities by using the dbremote utility, and a MobiLink user to perform synchronization activities by using the dbmlsync utility. It does not allow administration of replication, however. The SYS_REPLICATION_ADMIN_ROLE system role is provided for replication administration.

Pre-16.0 Authority	Equivalent Role	Description
RESOURCE authority	SYS_AUTH_RESOURCE_ROLE compatibility role	Allows a user to create database objects, such as tables, views, stored procedures, and triggers.
VALIDATE authority	SYS_AUTH_VALIDATE_ROLE compatibility role	Allows a user to perform database, table, index, and checksum validation by using the VALIDATE statement or dbvalid utility.
WRITECLIENTFILE authority	SYS_AUTH_WRITECLIENTFILE_ROLE compatibility role	Allows a user to write to files on a client computer, for example when using the UNLOAD TABLE statement to write data to a client computer.
WRITEFILE authority	SYS_AUTH_WRITEFILE_ROLE compatibility role	Allows a user to execute the xp_write_file system procedure.

With an authority-based security model, if a user did not need all of the permissions vested in an authority, there was no way to limit the grant. As a result, users were often granted more permissions than necessary, a potential security concern. The role-based security model addresses this concern, allowing privileges to be granted at a granular level.

Since the migration process ensures that all of a user's privileges are preserved during migration, it is strongly recommended that you review the compatibility role grants and of each user post-migration and adjust membership as necessary.

Permissions Become Privileges

In pre-16.0 databases, there were object-level permissions such as ALTER and SELECT for tables and views, and so on. While statements that grant or revoke these permissions still work, these permissions are now referred to as privileges, but retain the same name.

In addition to object-level privileges, there is a grantable system privilege for every operation that requires authorization to perform. When you upgrade your database, users that had permissions are automatically updated to have the equivalent privileges they need to perform the tasks they could perform before.

Groups Become Roles

During the upgrade of a pre-16.0 database, each group is converted to a user-extended role of the same name. Members of the original group are automatically granted the new role and all of its underlying privileges. Authorities and object-level permissions that were granted to the original group are converted to their equivalent roles and system privileges and granted to the user-extended role.

If an authority was inheritable, the compatibility role will be inherited by grantees of the new user-extended role. If the authority was non-inheritable, the grantees of the user-extended role do not inherit the compatibility role. If the legacy group had a password, only the extended user of the user-extended role inherits the underlying system privileges of the non-inheritable compatibility role.

The following table shows the system users and groups and the roles they are converted to.

Pre-16.0 Group	Role	Description
dbo	dbo	This role owns many system stored procedures, views, and tables.
diagnostics	diagnostics	This role owns the diagnostic tables and views, and can perform operations on them.
PUBLIC	PUBLIC	This role has SELECT permission on the system tables. Any new user ID is automatically granted the PUBLIC role.
ra_systabgroup	rs_systabgroup	This role allows users to perform replication server functionality.
SYS	SYS	This role owns the system tables and views (IQ catalog) for the database, and can perform operations on them.
SYS_SPATIAL_ADMIN_ROLE	SYS_SPATIAL_ADMIN_ROLE	This role allows users to create, alter, or drop spatial objects.

Change to Concept of a Super-User (DBA Authority)

In pre-16.0 databases, you could create a super-user by granting them DBA authority. Users with DBA authority could perform any privileged task in the system. When you upgrade your database, any users that had DBA authority gets the SYS_AUTH_DBA_ROLE compatibility role, and automatically receives exercise and administration rights for all roles and privileges that are present at the time of upgrade.

When you create a new role and don't specify an administrator at creation time, users with the MANAGE ROLES system privilege (global administrators) can administer the role. Since MANAGE ROLES is one of the system privileges granted to the SYS_AUTH_DBA_ROLE compatibility role, super-users can administer new roles.

However, if you create a new role and assign administrators as part of role creation, administration is then limited to those administrators. Therefore, with SAP Sybase IQ 16.0 and later, if you want your super-user to have administrative rights for new roles, you must explicitly grant it by making them an administrator of the role.

In SAP Sybase IQ 16.0, the SYS_AUTH_DBA_ROLE compatibility role can be migrated to a user-defined role, and once each underlying system privilege has been granted to at least one other role, can be dropped. Therefor, in order to preserve the ability of a super-user to perform any privileged task in the system, before dropping the SYS_AUTH_DBA_ROLE compatibility role, each of its underlying system privileges must be granted directly or indirectly to the super-user.

In pre-16.0 databases, the DBA user was often considered a super-user by virtue of being granted the DBA authority. The DBA user continues to exist with 16.0, and after migration is granted the SYS_AUTH_DBA_ROLE compatibility role. However, the DBA will be unable to administer any role with administrators assigned as part of role creation unless explicitly granted.

Changes to the GRANT Statement Syntax

If you have applications that use the pre-16.0 GRANT statement syntax for authorities, permissions, and groups, you should modify them to use the updated syntax for roles and privileges. The table below shows you what the statements should be changed to. Use of the old GRANT syntax for authorities, permissions, and groups is supported, but deprecated.

In pre-16.0 databases, DBA, REMOTE DBA, RESOURCE, and VALIDATE authorities were non-inheritable. When your database is upgraded, the WITH NO SYSTEM PRIVILEGE INHERITANCE clause is specified to ensure that inheritance behavior remains consistent with previous releases.

Also, in pre-16.0 databases, users that were granted DBA and REMOTE DBA authorities automatically could grant them to others. The WITH ADMIN clause in the new syntax ensures that administration rights behavior remains consistent with previous releases.

Table 4. NON-INHERITABLE AUTHORITIES

Pre-16.0 Syntax	New Syntax
GRANT DBA TO *<grantee>[,...]*	**GRANT ROLE SYS_AUTH_DBA_ROLE TO** *<grantee> [,...]* **WITH ADMIN OPTION** **WITH NO SYSTEM PRIVILEGE INHERITANCE**
GRANT REMOTE DBA TO *<grantee>[,...]*	**GRANT ROLE SYS_RUN_REPLICATION_ROLE TO** *<grantee> [,...]* **WITH NO ADMIN OPTION** **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Pre-16.0 Syntax	New Syntax
GRANT BACKUP TO *<grantee>[,...]*	**GRANT ROLE SYS_AUTH_BACKUP_ROLE TO** *<grantee> [,...]* **WITH NO SYSTEM PRIVILEGE INHERITANCE**
GRANT RESOURCE TO *<grantee>[,...]*	**GRANT ROLE SYS_AUTH_RESOURCE_ROLE TO** *<grantee> [,...]* **WITH NO SYSTEM PRIVILEGE INHERITANCE**
GRANT VALIDATE TO *<grantee>[,...]*	**GRANT ROLE SYS_AUTH_VALIDATE_ROLE TO** *<grantee> [,...]* **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Table 5. INHERITABLE AUTHORITIES

Pre-16.0 SYNTAX	NEW SYNTAX
GRANT Multiplex Admin TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_MULTIPLEX_ADMIN_ROLE TO** *<grantee> [,...]*
GRANT Operator TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_OPERATOR_ROLE TO** *<grantee> [,...]*
GRANT Perms Admin TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_PERMS_ADMIN_ROLE TO** *<grantee> [,...]*
GRANT PROFILE TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_PROFILE_ROLE TO** *<grantee> [,...]*
GRANT READCLIENTFILE TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_READCLIENTFILE_ROLE TO** *<grantee> [,...]*
GRANT READFILE TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_READFILE_ROLE TO** *<grantee> [,...]*
GRANT Space Admin TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_SPACE_ADMIN_ROLE TO** *<grantee> [,...]*
GRANT Spatial Admin TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_SPATIAL_ADMIN_ROLE TO** *<grantee> [,...]*
GRANT WRITECLIENTFILE TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_WRITECLIENTFILE_ROLE TO** *<grantee> [,...]*
GRANT WRITEFILE TO *<grantee> [,...]*	**GRANT ROLE SYS_AUTH_WRITEFILE_ROLE TO** *<grantee> [,...]*
GRANT CONNECT TO *<username>* [**IDENTIFIED BY** *<pwd>*]	No change

Pre-16.0 SYNTAX	NEW SYNTAX
GRANT GROUP TO *<user>*	**CREATE OR REPLACE** *<rolename>* **FOR USER** *<user>*
GRANT MEMBERSHIP IN GROUP *<groupname>[,...]* **TO** *<grantee>[,...]*	**GRANT ROLE** *<groupname>[,...]* **TO** *<grantee>[,...]*
GRANT PUBLISH TO *<grantee>*	No change. However, you can also set the new PUBLIC option, db_publisher: **SET OPTION PUBLIC.db_publisher=***<grantee_id>*
GRANT *<permission>[,...]* **ON** *[owner.]object-name* **TO** *<grantee>[,...]* **[WITH GRANT OPTION]** *<permission>*: ALL [PRIVILEGES] | ALTER | DELETE | INSERT | REFERENCES [(column-name, ...)] | SELECT [(column-name, ...)] | UPDATE [(column-name, ...)]	No Change
GRANT EXECUTE ON *[owner.]{ procedure-name / user-defined-function* } **TO** *<grantee>[,...]*	No Change
GRANT INTEGRATED LOGIN TO *<user-profile-name>[,...]* **AS USER** *<user>*	No Change
GRANT KERBEROS LOGIN **TO** *client-Kerberos-principal [, …]* **AS USER** *<user>*	No Change
GRANT CREATE ON *<dbspacename> [,...]* **TO** *<grantee> [,...]*	No Change

Changes to the REVOKE Statement Syntax

If you have applications that use the pre-16.0 REVOKE statement syntax for authorities, permissions, and groups, you should modify them to use the updated syntax for roles and privileges. The table below shows you what the statements should be changed to. Use of the old REVOKE syntax for authorities, permissions, and groups is supported but deprecated.

Pre-16.0 Syntax	New Syntax
REVOKE CONNECT FROM *<user>*	No change
REVOKE GROUP FROM *<user>*	**DROP** *<rolename>* **FROM USER** *<user>* **WITH REVOKE**
REVOKE MEMBERSHIP IN GROUP *<groupname> [,...]* **FROM** *<grantee> [,...]*	**REVOKE ROLE** *<groupname>[,...]* **FROM** *<grantee> [,...]*
REVOKE *<authority>[,...]* **FROM** *<grantee> [,...]* *<authority>*: BACKUP |DBA |Multiplex Admin |Operator |Perms Admin |PROFILE |READCLIENTFILE |READFILE |REMOTE DBA |RESOURCE | ALL |Space Admin |Spatial Admin |User Admin |VALIDATE |WRITECLIENTFILE |WRITEFILE	**REVOKE** *<rolename>[,...]* **FROM** *<grantee> [,...]* *<rolename>*: SYS_AUTH_BACKUP_ROLE |SYS_AUTH_DBA_ROLE |SYS_AUTH_MULTIPLEX_ADMIN_ROLE |SYS_AUTH_OPERATOR_ROLE |SYS_AUTH_PERMS_ADMIN_ROLE |SYS_AUTH_PROFILE_ROLE |SYS_READCLIENTFILE_ROLE |SYS_AUTH_READFILE_ROLE |SYS_RUN_REPLICATION_ROLE |SYS_AUTH_RESOURCE_ROLE |SYS_AUTH_SPACE_ADMIN_ROLE |SYS_AUTH_SPATIAL_ADMIN_ROLE |SYS_AUTH_USER_ADMIN_ROLE |SYS_AUTH_VALIDATE_ROLE |SYS_AUTH_WRITECLIENTFILE_ROLE |SYS_AUTH_WRITEFILE_ROLE

Pre-16.0 Syntax	New Syntax
REVOKE PUBLISH FROM *grantee*	No change. However, you can also set the new PUBLIC option, db_publisher: **SET OPTION PUBLIC.db_publisher=grantee**
REVOKE *<permission>[,...]* **ON** *[owner.]object-name* **FROM** *<grantee>[,...]* *<permission>*: ALL [PRIVILEGES] | ALTER | DELETE | INSERT | REFERENCES [(column-name, ...)] | SELECT [(column-name, ...)] | UPDATE [(column-name, ...)]	No change, except to naming convention. Object-level permissions are now object-level privileges.

REVOKE EXECUTE ON [*owner.*]{ *procedure-name* | *user-defined-function* }
FROM *<grantee> [,...]*
	No Change

REVOKE INTEGRATED LOGIN FROM *<user>*	No Change
REVOKE KERBEROS LOGIN FROM *<user> [,...]* **AS USER** *<user>*	No Change
REVOKE CREATE ON *<dbspacename> [,...]* **FROM** *<grantee> [,...]*	No Change

Changes to REMOTE DBA

In pre-16.0 databases, REMOTE DBA authority allowed a user to perform replication and synchronization operations using **dbremote** and **dbmlsync**.

The REMOTE DBA authority has been replaced by the SYS_RUN_REPLICATION_ROLE system role. Change your applications to grant this role, instead of REMOTE DBA.

The GRANT REMOTE DBA statement syntax is still supported but deprecated. Another replication-related role has also been introduced: the SYS_REPLICATION_ADMIN_ROLE system role. This role allows user to administer replication.

Changes in Inheritance Behavior for Some Authorities That Became Compatibility Roles

In pre-16.0 databases, if you granted the DBA, REMOTE DBA, BACKUP, RESOURCE, and VALIDATE authorities to a group, the underlying permissions were not inherited by members of the group.

Now, however, the default behavior when granting one of these roles (now called SYS_AUTH_DBA_ROLE, SYS_RUN_REPLICATION_ROLE, SYS_AUTH_BACKUP_ROLE, SYS_AUTH_RESOURCE_ROLE, and SYS_AUTH_VALIDATE_ROLE) to a user-defined role is to allow those who have been granted the user-defined role to inherit the underlying system privileges of the role.

Suppose you have a user, userA. You grant userA the ALTER ANY OBJECT system privilege. You then decide to extend userA to become a role, and then grant userA to userB. Now you want to grant the SYS_AUTH_DBA_ROLE role to userA, but you don't want userB to inherit all the privileges that the SYS_AUTH_DBA_ROLE role gives. You would therefore grant the SYS_AUTH_DBA_ROLE role as follows:

```
GRANT ROLE SYS_AUTH_DBA_ROLE TO userA WITH NO SYSTEM PRIVILEGE
INHERITANCE;
```

In this scenario, userB inherits only the ALTER ANY OBJECT system privilege from userA.

To retain the non-inheritance behavior of these roles after upgrading, include the WITH NO SYSTEM PRIVILEGE INHERITANCE clause in the GRANT ROLE statement. Likewise, if you have applications that you are changing to use the new GRANT syntax, you must specify this clause as well. This clause is only for use with these specific roles.

Note: The **WITH NO SYSTEM PRIVILEGE INHERITANCE** clause is only supported with these specific roles; any other use results in an error.

Changes in administering the database publisher

In pre-16.0 databases, the database publisher was controlled by granting the PUBLISH authority by using the GRANT PUBLISH and REVOKE PUBLISH statements. The current publisher could be determined by querying the CURRENT PUBLISHER special value.

he PUBLISH authority has been replaced by the PUBLIC.db_publisher database option, which requires the SET ANY SYSTEM OPTION system privilege to be set. Changing the publisher can be achieved by changing the database option, but for backwards compatibility, you can still change it using GRANT PUBLISH and REVOKE PUBLISH. You can also still query the CURRENT PUBLISHER to find out the current publisher.

Changes to System Procedures that Perform Privileged Operations

As part of the enhanced security of role-based security, the way in which privileged system procedures run has changed. Pre-16.0, a privileged system procedure ran with the privileges of its owner, typically dbo, and is referred to as the SYSTEM PROCEDURE DEFINER model. With 16.0, privileged system procedures run with the privileges of the person executing it, and is referred to as the SYSTEM PROCEDURE INVOKER model.

Note: This behavior change applies to SAP Sybase IQ privileged system procedures only, not user-defined stored procedures.

In pre-16.0, with the SYSTEM PROCEDURE DEFINER model, when you grant a user explicit EXECUTE privilege on a system procedure, any privileges required to run any authorized tasks associated with the system procedure are automatically inherited from the owner (definer of the system procedure), allowing the user to successfully run the system procedure.

In 16.0, with the SYSTEM PROCEDURE INVOKER model, the EXECUTE privilege for each system procedure is now granted to the PUBLIC role. Since every user, by default, is a member of the PUBLIC role, every user automatically inherits the required EXECUTE privilege. What is not inherited with the grant of EXECUTE privilege are any associated privileges required to run system procedure. These must now be granted directly or indirectly to the user before he or she can successfully run a system procedure.

This behaviour change has the potential to cause loss of functionality on custom stored procedures and applications that explicitly grant EXECUTE privilege on system procedures. For this reason, a default upgrade of a pre-16.0 database uses a combination of the two models. In the combination model, pre-16.0 privileged system procedures continue to run using the SYSTEM PROCEDURE DEFINER model, while any privileged system procedures introduced with 16.0 (or any future release) use the SYSTEM PROCEDURE INVOKER model.

If the potential loss of functionality is not of concern to your installation, you can override the default upgrade behavior so that all privileged system procedures (pre-16.0, new, and any future releases) use the SYSTEM PROCEDURE INVOKER model only. If you are unsure whether the potential loss of functionality will impact your database, upgrade using the default behavior and investigate. If you determine after the fact that it is not an issue, and you want to run all system procedures using the SYSTEM PROCEDURE INVOKER model, you can use the **ALTER DATABASE** statement to change the default security model.

The CREATE DATABASE statement, ALTER DATABASE UPGRADE statement, and Initialization utility (iqinit) have been enhanced to allow specification of a security model.

There is a small subset of pre-16.0 privileged system procedures that has always run with the privileges of the user running the procedure, not the owner of the procedure. To run these system procedures, in addition to requiring EXECUTE privilege on the system procedure, the user must be granted additional system privileges specific to the system procedure. Refer to the documentation for the required system privileges. This behavior remains unchanged in 16.0, regardless of the security model setting.

Grant Compatibility Roles

Granting a compatibility role is semantically equivalent to granting each of its underlying system privileges and roles.

You can drop compatibility roles once each of the system privileges granted to a compatibility role have been granted to at least one user-defined role. You cannot modify individual system privileges within each compatibility role. With the exception of the SYS_AUTH_SA_ROLE, SYS_AUTH_SSO_ROLE, and SYS_AUTH_DBA_ROLE roles, compatibility roles can be dropped at any time, if not required. You can re-create any dropped compatibility role, if needed.

Use the compatibility roles SYS_AUTH_SA_ROLE and SYS_AUTH_SSO_ROLE to administer and grant all individual system privileges in a new database. The union of the system privileges of these two roles are granted to the compatibility role SYS_AUTH_DBA_ROLE. By default, SYS_AUTH_DBA_ROLE is granted to the DBA user with administrative privileges. Thus, all system privileges are initially granted to the DBA user.

To migrate all system privileges within a specific compatibility role to a single user-defined role, use the **ALTER ROLE** statement with the **MIGRATE** clause.

You can grant and revoke users or other roles to compatibility roles.

Granting SYS_AUTH_SA_ROLE

Allows users to perform authorized tasks pertaining to data and system administration responsibilities.

Prerequisites

Administrative privilege over SYS_AUTH_SA_ROLE role.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_SA_ROLE role, execute one of these statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_SA_ROLE TO** *grantee [,...]* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_SA_ROLE TO** *grantee [,...]* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_SA_ROLE TO** *grantee [,...]* **WITH NO ADMIN OPTION**

System Privileges Granted to SYS_AUTH_SA_ROLE

System privileges granted to the SYS_AUTH_SA_ROLE role. Each system privilege is granted with the **WITH ADMIN OPTION** clause.

- ACCESS SERVER LS system privilege
- ALTER ANY INDEX system privilege
- ALTER ANY MATERIALIZED VIEW system privilege
- ALTER ANY OBJECT system privilege
- ALTER ANY PROCEDURE system privilege
- ALTER ANY SEQUENCE system privilege
- ALTER ANY TEXT CONFIGURATION system privilege
- ALTER ANY TABLE system privilege
- ALTER ANY TRIGGER system privilege
- ALTER ANY VIEW system privilege
- ALTER DATABASE system privilege
- ALTER DATATYPE system privilege
- BACKUP DATABASE system privilege
- CHECKPOINT system privilege
- COMMENT ANY OBJECT system privilege
- CREATE ANY INDEX system privilege
- CREATE ANY MATERIALIZED VIEW system privilege
- CREATE ANY OBJECT system privilege
- CREATE ANY PROCEDURE system privilege
- CREATE ANY SEQUENCE system privilege
- CREATE ANY TABLE system privilege
- CREATE ANY TEXT CONFIGURATION system privilege
- CREATE ANY TRIGGER system privilege
- CREATE ANY VIEW system privilege
- CREATE DATATYPE system privilege

- CREATE EXTERNAL REFERENCE system privilege
- CREATE MATERIALIZED VIEW system privilege
- CREATE MESSAGE system privilege
- CREATE PROCEDURE system privilege
- CREATE PROXY TABLE system privilege
- CREATE TABLE system privilege
- CREATE TEXT CONFIGURATION system privilege
- CREATE VIEW system privilege
- DEBUG ANY PROCEDURE system privilege
- DELETE ANY TABLE system privilege
- DROP ANY INDEX system privilege
- DROP ANY MATERIALIZED VIEW system privilege
- DROP ANY OBJECT system privilege
- DROP ANY PROCEDURE system privilege
- DROP ANY SEQUENCE system privilege
- DROP ANY TABLE system privilege
- DROP ANY TEXT CONFIGURATION system privilege
- DROP ANY VIEW system privilege
- DROP DATATYPE system privilege
- DROP MESSAGE system privilege
- EXECUTE ANY PROCEDURE system privilege
- INSERT ANY TABLE system privilege
- LOAD ANY TABLE system privilege
- MANAGE ANY DBSPACE system privilege
- MANAGE ANY EVENT system privilege
- MANAGE ANY EXTERNAL ENVIRONMENT system privilege
- MANAGE ANY EXTERNAL OBJECT system privilege
- MANAGE ANY MIRROR SERVER system privilege
- MANAGE ANY SPATIAL OBJECT system privilege
- MANAGE ANY STATISTICS system privilege
- MANAGE ANY WEB SERVICE system privilege
- MANAGE MULTIPLEX system privilege
- MANAGE PROFILING system privilege
- MANAGE REPLICATION system privilege
- MONITOR system privilege
- READ CLIENT FILE system privilege
- READ FILE system privilege
- REORGANIZE ANY OBJECT system privilege
- SELECT ANY TABLE system privilege

- SERVER OPERATOR system privilege
- SET ANY PUBLIC OPTION system privilege
- SET ANY SYSTEM OPTION system privilege
- SET ANY USER DEFINED OPTION system privilege
- TRUNCATE ANY TABLE system privilege
- UPDATE ANY TABLE system privilege
- UPGRADE ROLE system privilege
- USE ANY SEQUENCE system privilege
- VALIDATE ANY OBJECT system privilege
- WRITE CLIENT FILE system privilege
- WRITE FILE system privilege

Granting SYS_AUTH_SSO_ROLE

Grant to allow users to perform authorized tasks pertaining to security and access control responsibilities.

Prerequisites

Administrative privilege over SYS_AUTH_SSO_ROLE role.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

To grant the role, execute one of these statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_SSO_ROLE TO** *grantee [,...]* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_SSO_ROLE TO** *grantee [,...]* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_SSO_ROLE TO** *grantee [,...]* **WITH NO ADMIN OPTION**

System Privileges Granted to SYS_AUTH_SSO_ROLE

System privileges granted to the SYS_AUTH_SSO_ROLE role. Each system privilege is granted with the **WITH ADMIN OPTION** clause.

- ALTER ANY OBJECT OWNER system privilege
- ANY USER system privilege
- CHANGE PASSWORD system privilege
- DROP CONNECTION system privilege
- MANAGE ANY OBJECT PRIVILEGES system privilege
- MANAGE ANY LDAP SERVER system privilege
- MANAGE ANY LOGIN POLICY system privilege
- MANAGE ANY USER system privilege
- MANAGE AUDITING system privilege
- MANAGE ROLES system privilege
- SET ANY SECURITY OPTION system privilege
- SET USER system privilege (granted with the WITH ADMIN ONLY OPTION clause)

Granting SYS_AUTH_DBA_ROLE

Grant to allow users to perform all authorized tasks.

Prerequisites

Administrative privilege over SYS_AUTH_DBA_ROLE role.

Task

This role indirectly grants all compatibility roles, as well as some system roles to a user. It is the union of the underlying system privileges of each of these roles that makes the SYS_AUTH_DBA_ROLE role the "super" role.

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

Note: If you are migrating from SAP Sybase IQ 15.4 or earlier, the concept of inheritance of the underlying system privileges of this system role represents a change in behavior with SAP Sybase IQ 16.0 or later. For SAP Sybase IQ 15.4 and earlier behavior, use the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

The WITH ADMIN ONLY OPTION clauses is invalid when using the WITH NO SYSTEM PRIVILEGE INHERITANCE. clause. The WITH NO ADMIN OPTION clause is valid, but not required, as it is semantically equivalent to the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

To grant the SYS_AUTH_DBA_ROLE role, execute one of these statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_DBA_ROLE TO** *grantee [,...]* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_DBA_ROLE TO** *grantee [,...]* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_DBA_ROLE TO** *grantee [,...]* **WITH NO ADMIN OPTION**
With full administrative rights, **but no system privilege inheritance**	**GRANT ROLE SYS_AUTH_REMOTE_DBA_ROLE** TO *user_ID* **WITH ADMIN OPTION** **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Roles Granted to SYS_AUTH_DBA_ROLE

Roles granted to the SYS_AUTH_DBA_ROLE role.

These compatibility roles are granted with the WITH ADMIN OPTION clause:

- SYS_AUTH_SA_ROLE
- SYS_AUTH_SSO_ROLE

These compatibility roles are granted with the WITH ADMIN ONLY OPTION clause:

- SYS_AUTH_RESOURCE_ROLE
- SYS_AUTH_BACKUP_ROLE
- SYS_AUTH_VALIDATE_ROLE
- SYS_AUTH_READFILE_ROLE
- SYS_AUTH_PROFILE_ROLE
- SYS_AUTH_READCLIENTFILE_ROLE
- SYS_AUTH_WRITECLIENTFILE_ROLE
- SYS_AUTH_WRITEFILE_ROLE
- SYS_AUTH_USER_ADMIN_ROLE
- SYS_AUTH_SPACE_ADMIN_ROLE
- SYS_AUTH_MULTIPLEX_ADMIN_ROLE
- SYS_AUTH_OPERATOR_ROLE
- SYS_AUTH_PERMS_ADMIN_ROLE

These system roles are granted with the WITH ADMIN ONLY OPTION clause:

- SYS_SPATIAL_ADMIN_ROLE
- diagnostics
- rs_systabgroup
- SYS
- DBO
- PUBLIC

System Privileges Granted to SYS_AUTH_DBA_ROLE

System privileges granted to the SYS_AUTH_DBA_ROLE role.

Through the granting of all compatibility roles and select system roles, these system privileges are indirectly granted to the SYS_AUTH_DBA_ROLE role. The underlying system privileges of the SYS_AUTH_SA_ROLE and SYS_AUTH_SSO_ROLE roles are indirectly granted with the WITH ADMIN OPTION clause, which grants full administrative rights. All other compatibility roles and system roles are indirectly granted with the WITH ADMIN ONLY OPTION clause.

- ACCESS SERVER LS system privilege
- ALTER ANY INDEX system privilege
- ALTER ANY MATERIALIZED VIEW system privilege
- ALTER ANY OBJECT system privilege
- ALTER ANY OBJECT OWNER system privilege
- ALTER ANY PROCEDURE system privilege
- ALTER ANY SEQUENCE system privilege
- ALTER ANY TABLE system privilege
- ALTER ANY TEXT CONFIGURATION system privilege
- ALTER ANY TRIGGER system privilege
- ALTER ANY VIEW system privilege
- ALTER DATABASE system privilege
- ALTER DATATYPE system privilege
- BACKUP DATABASE system privilege
- CHANGE PASSWORD system privilege
- CHECKPOINT system privilege
- COMMENT ANY OBJECT system privilege
- CREATE ANY INDEX system privilege
- CREATE ANY MATERIALIZED VIEW system privilege
- CREATE ANY OBJECT system privilege
- CREATE ANY PROCEDURE system privilege
- CREATE ANY SEQUENCE system privilege
- CREATE ANY TABLE system privilege
- CREATE ANY TEXT CONFIGURATION system privilege

- CREATE ANY TRIGGER system privilege
- CREATE ANY VIEW system privilege
- CREATE DATATYPE system privilege
- CREATE EXTERNAL REFERENCE system privilege
- CREATE MATERIALIZED VIEW system privilege
- CREATE MESSAGE system privilege
- CREATE PROCEDURE system privilege
- CREATE PROXY TABLE system privilege
- CREATE TABLE system privilege
- CREATE TEXT CONFIGURATION system privilege
- CREATE VIEW system privilege
- DEBUG ANY PROCEDURE system privilege
- DELETE ANY TABLE system privilege
- DROP ANY INDEX system privilege
- DROP ANY MATERIALIZED VIEW system privilege
- DROP ANY OBJECT system privilege
- DROP ANY PROCEDURE system privilege
- DROP ANY SEQUENCE system privilege
- DROP ANY TABLE system privilege
- DROP ANY TEXT CONFIGURATION system privilege
- DROP ANY VIEW system privilege
- DROP CONNECTION system privilege
- DROP DATATYPE system privilege
- DROP MESSAGE system privilege
- EXECUTE ANY PROCEDURE system privilege
- LOAD ANY TABLE system privilege
- INSERT ANY TABLE system privilege
- MANAGE ANY DBSPACE system privilege
- MANAGE ANY EVENT system privilege
- MANAGE ANY EXTERNAL ENVIRONMENT system privilege
- MANAGE ANY EXTERNAL OBJECT system privilege
- MANAGE ANY LDAP SERVER system privilege
- MANAGE ANY LOGIN POLICY system privilege
- MANAGE ANY MIRROR SERVER system privilege
- MANAGE ANY OBJECT PRIVILEGES system privilege
- MANAGE ANY SPATIAL OBJECT system privilege
- MANAGE ANY STATISTICS system privilege
- MANAGE ANY USER system privilege
- MANAGE ANY WEB SERVICE system privilege

- MANAGE AUDITING system privilege
- MANAGE MULTIPLEX system privilege
- MANAGE PROFILING system privilege
- MANAGE REPLICATION system privilege
- MANAGE ROLES system privilege
- MONITOR system privilege
- READ CLIENT FILE system privilege
- READ FILE system privilege
- REORGANIZE ANY OBJECT system privilege
- SELECT ANY TABLE system privilege
- SERVER OPERATOR system privilege
- SET ANY PUBLIC OPTION system privilege
- SET ANY SECURITY OPTION system privilege
- SET ANY SYSTEM OPTION system privilege
- SET ANY USER DEFINED OPTION system privilege
- SET USER system privilege (granted with ADMIN ONLY clause)
- TRUNCATE ANY TABLE system privilege
- UPDATE ANY TABLE system privilege
- UPGRADE ROLE system privilege
- USE ANY SEQUENCE system privilege
- VALIDATE ANY OBJECT system privilege
- WRITE CLIENT FILE system privilege
- WRITE FILE system privilege

Granting SYS_AUTH_BACKUP_ROLE

Grant to allow users to perform all backups.

Prerequisites

Administrative privilege over SYS_AUTH_BACKUP_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

Note: For users migrating from SAP Sybase IQ 15.4 and earlier, the concept of inheritance of the underlying system privileges of this system role represents a change in behavior with SAP Sybase IQ 16.0 or later. For SAP Sybase IQ 15.4 and earlier behavior, use the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

The WITH ADMIN ONLY OPTION and WITH ADMIN OPTION clauses are invalid when using the WITH NO SYSTEM PRIVILEGE INHERITANCE. clause. The WITH NO ADMIN OPTION clause is valid, but not required, as it is semantically equivalent to the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

To grant the SYS_AUTH_BACKUP_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_BACKUP_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_BACKUP_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_BACKUP_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**
With no system privilege inheritance	**GRANT ROLE SYS_AUTH_BACKUP_ROLE** TO *user_ID* **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Example:

This example grants the SYS_AUTH_BACKUP_ROLE to Mary and Joe, in two ways. Mary is granted administrative rights to the role and inherits the underlying system privileges of the role while Joe is granted neither.

```
GRANT ROLE SYS_AUTH_BACKUP_ROLE TO Mary WITH ADMIN OPTION
```

```
GRANT ROLE SYS_AUTH_BACKUP_ROLE TO Joe
WITH NO SYSTEM PRIVILEGE INHERITANCE
```

System Privileges Granted to SYS_AUTH_BACKUP_ROLE

The SYS_AUTH_BACKUP_ROLE role is granted the BACKUP DATABASE system privilege with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_MULTIPLEX_ADMIN_ROLE

Grant to allow users to perform authorized tasks to manage Multiplex.

Prerequisites

Administrative privilege over SYS_AUTH_MULTIPLEX_ADMIN_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the

underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_MULTIPLEX_ADMIN_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_MULTIPLEX_ADMIN_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_MULTIPLEX_ADMIN_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_MULTIPLEX_ADMIN_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_MULTIPLEX_ADMIN_ROLE to Mary, with no administrative options.

```
GRANT ROLE SYS_AUTH_MULTIPLEX_ADMIN_ROLE TO Mary WITH NO ADMIN
OPTION
```

System Privileges Granted to SYS_AUTH_MULTIPLEX_ADMIN_ROLE

The SYS_AUTH_MULTIPLEX_ADMIN_ROLE role is granted the ACCESS SERVER LS and MANAGE MULTIPLEX system privileges with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_OPERATOR_ROLE

Grant to allow users to checkpoint databases, drop connections (including those for users with SYS_AUTH_DBA_ROLE), back up databases, and monitor the system.

Prerequisites

Administrative privilege over SYS_AUTH_OPERATOR_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

Note: For users migrating from SAP Sybase IQ 15.4 and earlier, the concept of inheritance of the underlying system privileges of this system role represents a change in behavior with SAP Sybase IQ 16.0 or later. For SAP Sybase IQ 15.4 and earlier behavior, use the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

The WITH ADMIN ONLY OPTION and WITH ADMIN OPTION clauses are invalid when using the WITH NO SYSTEM PRIVILEGE INHERITANCE. clause. The WITH NO ADMIN OPTION clause is valid, but not required, as it is semantically equivalent to the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

To grant the SYS_AUTH_OPERATOR_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_OPERATOR_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_OPERATOR_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_OPERATOR_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**
With no system privilege inheritance	**GRANT ROLE SYS_AUTH_OPERATOR_ROLE** TO *user_ID* **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Example:

This example grants the SYS_AUTH_OPERATOR_ROLE to Mary and Joe, in two ways. Mary is granted administrative rights to the role and inherits the underlying system privileges of the role while Joe is granted neither.

```
GRANT ROLE SYS_AUTH_OPERATOR_ROLE TO Mary WITH ADMIN OPTION
```

```
GRANT ROLE SYS_AUTH_OPERATOR_ROLE TO Joe
WITH NO SYSTEM PRIVILEGE INHERITANCE
```

System Privileges Granted to SYS_AUTH_OPERATOR_ROLE

The SYS_AUTH_OPERATOR_ROLE role is granted several system privileges with the **WITH NO ADMIN OPTION** clause.

- ACCESS SERVER LS System Privilege

- BACKUP DATABASE System Privilege
- CHECKPOINT System Privilege
- DROP CONNECTION System Privilege
- MONITOR System Privilege

Granting SYS_AUTH_PERMS_ADMIN_ROLE

Grant to allow users to manage data privileges, groups, authorities, and passwords.

Prerequisites

Administrative privilege over SYS_AUTH_PERMS_ADMIN_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_PERMS_ADMIN_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_PERMS_ADMIN_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_PERMS_ADMIN_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_PERMS_ADMIN_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_PERMS_ADMIN_ROLE to Mary, with only administrative options.

```
GRANT ROLE SYS_AUTH_PERMS_ADMIN_ROLE TO Mary WITH ADMIN ONLY OPTION
```

Roles Granted to SYS_AUTH_PERMS_ADMIN_ROLE

List of roles granted to this SYS_AUTH_PERMS_ADMIN_ROLE role.

The following compatibility roles are granted with the **WITH ADMIN OPTION** clause:

- SYS_AUTH_BACKUP_ROLE
- SYS_AUTH_OPERATOR_ROLE
- SYS_AUTH_USER_ADMIN_ROLE
- SYS_AUTH_SPACE_ADMIN_ROLE
- SYS_AUTH_MULTIPLEX_ADMIN_ROLE
- SYS_AUTH_RESOURCE_ROLE
- SYS_AUTH_VALIDATE_ROLE
- SYS_AUTH_PROFILE_ROLE
- SYS_AUTH_READFILE_ROLE
- SYS_AUTH_READCLIENTFILE_ROLE
- SYS_AUTH_WRITEFILE_ROLE
- SYS_AUTH_WRITECLIENTFILE_ROLE

System Privileges Granted to SYS_AUTH_PERMS_ADMIN_ROLE

The SYS_AUTH_PERMS_ADMIN_ROLE role is granted several system privileges with the **WITH NO ADMIN OPTION** clause.

- CHANGE PASSWORD System Privilege
- MANAGE ANY OBJECT PRIVILEGES System Privilege
- MANAGE ROLES System Privilege

Granting SYS_AUTH_PROFILE_ROLE

Grant to allow users to enable/disable server tracing for application profiling.

Prerequisites

Administrative privilege over SYS_AUTH_PROFILE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges. By default, the SYS_AUTH_PROFILE_ROLE is granted the diagnostics system role with no administrative rights.

To grant the SYS_AUTH_PROFILE_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_PROFILE_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_PROFILE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_PROFILE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_PROFILE_ROLE to Mary, with administrative options.

```
GRANT ROLE SYS_AUTH_PROFILE_ROLE TO Mary WITH ADMIN OPTION
```

System Privileges Granted to SYS_AUTH_PROFILE_ROLE

the SYS_AUTH_PROFILE_ROLE role is granted the MANAGE PROFILING system privilege with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_READFILE_ROLE

Grant to allow users to read to a file resident on the server machine.

Prerequisites

Administrative privilege over SYS_AUTH_READFILE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_READFILE_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT SYS_AUTH_READFILE_ROLE** TO *user_ID* **WITH ADMIN OPTION**

Administrative Option	Statement
With administrative rights only	**GRANT SYS_AUTH_READFILE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT SYS_AUTH_READFILE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_READFILE_ROLE to Mary, with no administrative options.

```
GRANT ROLE SYS_AUTH_READFILE_ROLE TO Mary WITH NO ADMIN OPTION
```

System Privileges Granted to SYS_AUTH_READFILE_ROLE

The SYS_AUTH_READFILE_ROLE role is granted the READ FILE system privilege with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_READCLIENTFILE_ROLE

Grant to allow users to read to a file resident on the client machine.

Prerequisites

Administrative privilege over SYS_AUTH_READCLIENTFILE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

To grant the SYS_AUTH_READCLIENTFILE_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_READCLIENTFILE_ROLE** TO *user_ID* **WITH ADMIN OPTION**

Administrative Option	Statement
With administrative rights only	**GRANT ROLE SYS_AUTH_READCLIENTFILE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_READCLIENTFILE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_READCLIENTFILE_ROLE to Mary, with only administrative options.

```
GRANT ROLE SYS_AUTH_READCLIENTFILE_ROLE TO Mary WITH ADMIN ONLY
OPTION
```

System Privileges Granted to SYS_AUTH_READCLIENTFILE_ROLE

The SYS_AUTH_READCLIENTFILE_ROLE role is granted the READ CLIENT FILE system privilege with the **WITH NO ADMIN OPTION** clause.

Granting SYS_RUN_REPLICATION_ROLE

This role is required for performing replication tasks using **dbremote** and synchronization tasks using **dbmlsync**.

Prerequisites

MANAGE REPLICATION system privilege.

Task

The SYS_RUN_REPLICATION_ROLE system role is active only for users connecting through the **dbremote** or **dbmlsync** utilities.

The SYS_RUN_REPLICATION_ROLE system role is granted the SYS_AUTH_DBA_ROLE compatibility role with the WITH ADMIN OPTION clause. It is also granted these system privileges with the WITH NO ADMIN OPTION clause.

- SELECT ANY TABLE
- SET ANY USER DEFINED OPTION
- SET ANY SYSTEM OPTION
- BACKUP DATABASE
- MONITOR

By default, when granting SYS_RUN_REPLICATION_ROLE, the underlying system privileges were inherited by members of the receiving group. To prevent inheritance, the

WITH NO SYSTEM PRIVILEGE INHERITANCE clause can be included for this system role only.

This default set of system privileges cannot be revoked from the system role. Additional system privileges and roles can be granted and revoked from this system role.

The minimum number of role administrators (**MIN_ROLE_ADMINS**) database option ensures that a designated number of users always exist in the database who can grant and revoke the MANAGE REPLICATION system privilege to other users.

The SYS_AUTH_DBA_ROLE compatibility role is granted by default to the SYS_RUN_REPLICATION_ROLE system role to address any possible requirements for additional system privileges to perform other replication related authorized tasks over and above the above-noted explicitly granted system privileges. It is recommended, however, that the SYS_AUTH_DBA_ROLE compatibility role be revoked from SYS_RUN_REPLICATION_ROLE system role and those specific additional system privileges or roles identified be explicitly granted to the SYS_RUN_REPLICATION_ROLE system role.

The WITH ADMIN OPTION or WITH ADMIN ONLY OPTION clauses are not valid when granting the SYS_RUN_REPLICATION_ROLE system role.

To grant the SYS_RUN_REPLICATION_ROLE system role, execute one of these statements:

Inheritance Type	Statement
With inheritance	**GRANT ROLE SYS_RUN_REPLICATION_ROLE TO** *grantee [,...]*
With no inheritance	**GRANT ROLE SYS_RUN_REPLICATION_ROLE TO** *grantee [,...]* **WITH NO SYSTEM PRIVILEGE INHERITANCE**

System Privileges and Roles Granted to SYS_RUN_REPLICATION_ROLE

The SYS_RUN_REPLICATION_ROLE role is granted the SYS_AUTH_DBA_ROLE role with the **WITH ADMIN OPTION** clause. It is also granted several system privileges with the **WITH NO ADMIN OPTION** clause.

- SELECT ANY TABLE
- SET ANY USER DEFINED OPTION
- SET ANY SYSTEM OPTION
- BACKUP DATABASE
- MONITOR

This default set of system privileges granted cannot be revoked from the role. Additional system privileges and roles can be granted and revoked from this role.

Note: The SYS_AUTH_DBA_ROLE role is granted by default to the SYS_RUN_REPLICATION_ROLE role to address any possible requirements for additional system privileges to perform other replication related authorized tasks over and above the above-noted explicitly granted system privileges. It is recommended, however, that the SYS_AUTH_DBA_ROLE role be revoked from SYS_RUN_REPLICATION_ROLE role and those specific additional system privileges or roles identified be explicitly granted to the SYS_RUN_REPLICATION_ROLE role.

Granting SYS_AUTH_RESOURCE_ROLE

Grant to allow users to create new database objects, such as tables, views, indexes, or procedures.

Prerequisites

Administrative privilege over SYS_AUTH_RESOURCE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

Note: For users migrating from SAP Sybase IQ 15.4 and earlier, the concept of inheritance of the underlying system privileges of this system role represents a change in behavior with SAP Sybase IQ 16.0 or later. For SAP Sybase IQ 15.4 and earlier behavior, use the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

The WITH ADMIN ONLY OPTION and WITH ADMIN OPTION clauses are invalid when using the WITH NO SYSTEM PRIVILEGE INHERITANCE. clause. The WITH NO ADMIN OPTION clause is valid, but not required, as it is semantically equivalent to the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

To grant the SYS_AUTH_RESOURCE_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_RESOURCE_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_RESOURCE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**

Administrative Option	Statement
With no administrative rights	**GRANT ROLE SYS_AUTH_RESOURCE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**
With no system privilege inheritance	**GRANT ROLE SYS_AUTH_RESOURCE_ROLE** TO *user_ID* **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Example:

This example grants the SYS_AUTH_RESOURCE_ROLE to Mary and Joe, in two ways. Mary is granted administrative rights to the role and inherits the underlying system privileges of the role while Joe is granted neither.

```
GRANT ROLE SYS_AUTH_RESOURCE_ROLE TO Mary WITH ADMIN OPTION
```

```
GRANT ROLE SYS_AUTH_RESOURCE_ROLE TO Joe
WITH NO SYSTEM PRIVILEGE INHERITANCE
```

System Privileges Granted to SYS_AUTH_RESOURCE_ROLE

The SYS_AUTH_RESOURCE_ROLE role is granted several system privileges granted with the **WITH NO ADMIN OPTION** clause.

- CREATE TABLE system privilege
- CREATE PROXY TABLE system privilege
- CREATE VIEW system privilege
- CREATE MATERIALIZED VIEW system privilege
- CREATE PROCEDURE system privilege
- CREATE DATATYPE system privilege
- CREATE MESSAGE system privilege
- CREATE TEXT CONFIGURATION system privilege
- CREATE ANY SEQUENCE system privilege
- CREATE ANY TRIGGER system privilege
- ALTER ANY TRIGGER system privilege
- CREATE ANY OBJECT system privilege

Granting SYS_AUTH_SPACE_ADMIN_ROLE

Grant to allow users to manage dbspaces.

Prerequisites

Administrative privilege over SYS_AUTH_SPACE_ADMIN_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_SPACE_ADMIN_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_SPACE_ADMIN_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_SPACE_ADMIN_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_SPACE_ADMIN_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_SPACE_ADMIN_ROLE to Mary, with no administrative options.

```
GRANT ROLE SYS_AUTH_SPACE_ADMIN_ROLE TO Mary WITH NO ADMIN OPTION
```

System Privileges Granted to SYS_AUTH_SPACE_ADMIN_ROLE

The SYS_AUTH_SPACE_ADMIN_ROLE role is granted the ACCESS SERVER LS and MANAGE ANY DBSPACE system privileges with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_USER ADMIN_ROLE

Grant to allow users to manage external logins, login policies, and other users.

Prerequisites

Administrative privilege over SYS_AUTH_USER ADMIN_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can

manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_USER ADMIN_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_USER_ADMIN_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_USER_ADMIN_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_USER_ADMIN_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_USER_ADMIN_ROLE to Mary, with administrative options.

```
GRANT ROLE SYS_AUTH_USER_ADMIN_ROLE TO Mary WITH ADMIN OPTION
```

System Privileges Granted to SYS_AUTH_USER_ADMIN_ROLE

The SYS_AUTH_USER_ADMIN_ROLE role is granted the MANAGE ANY LOGIN POLICY and MANAGE ANY USER system privileges with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_VALIDATE_ROLE

Grant to allow users to validate or check tables, materialized views, indexes or databases in the system store that are owned by any user.

Prerequisites

Administrative privilege over SYS_AUTH_VALIDATE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.

Note: For users migrating from SAP Sybase IQ 15.4 and earlier, the concept of inheritance of the underlying system privileges of this system role represents a change in behavior with SAP Sybase IQ 16.0 or later. For SAP Sybase IQ 15.4 and earlier behavior, use the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

The WITH ADMIN ONLY OPTION and WITH ADMIN OPTION clauses are invalid when using the WITH NO SYSTEM PRIVILEGE INHERITANCE. clause. The WITH NO ADMIN OPTION clause is valid, but not required, as it is semantically equivalent to the WITH NO SYSTEM PRIVILEGE INHERITANCE clause.

To grant the SYS_AUTH_VALIDATE_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_VALIDATE_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_VALIDATE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_VALIDATE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**
With no system privilege inheritance	**GRANT ROLE SYS_AUTH_VALIDATE_ROLE** TO *user_ID* **WITH NO SYSTEM PRIVILEGE INHERITANCE**

Example:

This example grants the SYS_AUTH_VALIDATE_ROLE to Mary and Joe, in two ways. Mary is granted administrative rights to the role and inherits the underlying system privileges of the role while Joe is granted neither.

```
GRANT ROLE SYS_AUTH_VALIDATE_ROLE TO Mary WITH ADMIN OPTION
```

```
GRANT ROLE SYS_AUTH_VALIDATE_ROLE TO Joe
WITH NO SYSTEM PRIVILEGE INHERITANCE
```

System Privileges Granted to SYS_AUTH_VALIDATE_ROLE

The SYS_AUTH_VALIDATE_ROLE role is granted the VALIDATE ANY OBJECT system privilege with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_WRITEFILE_ROLE

Grant to allow users to write to a file resident on the server machine.

Prerequisites

Administrative privilege over SYS_AUTH_WRITEFILE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_WRITEFILE_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_WRITEFILE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_WRITEFILE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_WRITEFILE_ROLE to Mary, with no administrative options.

```
GRANT ROLE SYS_AUTH_WRITEFILE_ROLE TO Mary WITH NO ADMIN OPTION
```

System Privileges Granted to SYS_AUTH_WRITEFILE_ROLE

The SYS_AUTH_WRITEFILE_ROLE role is granted the WRITE FILE system privilege with the **WITH NO ADMIN OPTION** clause.

Granting SYS_AUTH_WRITECLIENTFILE_ROLE

Grant to allow users to write to a file resident on the client machine.

Prerequisites

Administrative privilege over SYS_AUTH_WRITECLIENTFILE_ROLE.

Task

You can grant this role with or without administrative rights. When granted with administrative rights, a user can manage (grant and revoke) the role, as well as use any of the underlying system privileges. When granted with administrative rights only, a user can

manage the role, but not use its underlying system privileges. Finally, when granted with no administrative rights, a user can only use its underlying system privileges.
To grant the SYS_AUTH_WRITECLIENTFILE_ROLE role, execute one of the following statements:

Administrative Option	Statement
With full administrative rights	**GRANT ROLE SYS_AUTH_WRITECLIENTFILE_ROLE** TO *user_ID* **WITH ADMIN OPTION**
With administrative rights only	**GRANT ROLE SYS_AUTH_WRITECLIENTFILE_ROLE** TO *user_ID* **WITH ADMIN ONLY OPTION**
With no administrative rights	**GRANT ROLE SYS_AUTH_WRITECLIENTFILE_ROLE** TO *user_ID* **WITH NO ADMIN OPTION**

Example:

This example grants the SYS_AUTH_WRITECLIENTFILE_ROLE to Mary, with only administrative options.

```
GRANT ROLE SYS_AUTH_WRITECLIENTFILE_ROLE TO Mary WITH ADMIN ONLY
OPTION
```

System Privileges Granted to SYS_AUTH_WRITECLIENTFILE_ROLE

The SYS_AUTH_WRITEFILECLIENT_ROLE role is granted the WRITE CLIENT FILE system privilege with the **WITH NO ADMIN OPTION** clause.

Revoking a Compatibility Role

Revoke a compatibility role from a user or role.

Prerequisites

Requires administrative privilege over the compatibility role being revoked.

Task

To revoke a compatibility role, execute one of these statements:

Administrative Option	Statement
Administrative rights only	**REVOKE ADMIN OPTION FOR ROLE** *compatibility_role* **FROM** *grantee [,...]*
Membership in the role and any administrative rights	**REVOKE ROLE** *compatibility_role* **FROM** *grantee [,...]*

Migrating a Compatibility Role

Migrate all underlying system privileges of a compatibility role to a user-defined role.

Prerequisites

Administrative privilege over the role being migrated, and the MANAGE ROLES system privilege.

Task

Compatibility roles are immutable, but they can be migrated in their entirety to a new user-defined role. Once migrated, the compatibility role is automatically dropped. This process is systematically equivalent to individually granting each underlying system privilege to a user-defined role, then manually dropping the compatibility role.

During migration:

- A new user-defined role is created.
- All of the system privileges currently granted to the migrating compatibility role are automatically granted to the new user-defined role.
- All users and roles currently granted to the migrating compatibility role are automatically granted to the new user-defined role.
- Administrators of the compatibility role continue to be the administrators of the new migrated role.
- The compatibility role is dropped.

You cannot use **ALTER ROLE** to migrate the compatibility roles SYS_AUTH_SA_ROLE and SYS_AUTH_SSO_ROLE. SYS_AUTH_SA_ROLE and SYS_AUTH_SSO_ROLE are automatically migrated when SYS_AUTH_DBA_ROLE is migrated.
To migrate a compatibility role, execute one of the following statements:

Compatibility Role	Statement
SYS_AUTH_DBA_ROLE **SYS_AUTH_DBA_ROLE is successfully migrated if:** • **SYS_AUTH_DBA_ROLE has not already been dropped.** • **The names of the new roles do not begin with the prefix SYS_ or end with the suffix _ROLE.** • **The names of the three new roles do not already exist in the database.**	**ALTER ROLE** *SYS_AUTH_DBA_ROLE* **MIGRATE TO** *new_role_name, new_sa_role_name, new_sso_role_name*
Any other compatibility role **The compatibility role is successfully migrated if:** • **The compatibility role being migrated has not already been dropped.** • **The name of the new role does not begin with the prefix SYS_ or end with the suffix _ROLE.** • **The name of the new role does not already exist in the database.**	**ALTER ROLE** *compatibility_sys_role_name* **MIGRATE TO** *new_role_name*

The following statements migrate the `SYS_AUTH_DBA_ROLE` to the new roles `Custom_DBA`, `Custom_SA`, and `Custom_SSO`, respectively, and migrate the `SYS_AUTH_OPERATOR_ROLE` role to the new role `Operator_role`. All users, underlying system privileges, and roles granted to the original roles are automatically migrated to the new roles. Finally, SYS_AUTH_DBA_ROLE, SYS_AUTH_SA_ROLE, SYS_AUTH_SSO_ROLE and SYS_AUTH_OPERATOR_ROLE are all dropped.

```
ALTER ROLE SYS_AUTH_DBA_ROLE
MIGRATE TO Custom_DBA, Custom_SA, Custom_SSO
```

```
ALTER ROLE SYS_AUTH_OPERATOR_ROLE
MIGRATE TO Operator_role
```

Dropping a Compatibility Role

All compatibility roles, with the exception of SYS_AUTH_SA_ROLE and SYS_AUTH_SSO_ROLE can be dropped. SYS_AUTH_SA_ROLE and

SYS_AUTH_SSO_ROLE are dropped automatically when SYS_AUTH_DBA_ROLE is dropped.

Prerequisites

Administrative privilege over the role being dropped.

Task

Unlike user-defined roles, compatibility roles cannot be user-extended roles, nor can they own objects. Therefore, only the **WITH REVOKE** clause is valid when you are dropping a compatibility role. As with user-defined roles, the **WITH REVOKE** clause is required when dropping a compatibility role to which users have been granted the underlying system privileges of the role.

To drop a compatibility role, execute one of the following statements:

Drop Condition	Statement
Compatibility role that does not have its underlying system privileges granted to any user **The role is successfully dropped if:** • **No users are currently granted the underlying system privileges of the role.** • **The role being dropped is not SYS_AUTH_SA_ROLE, SYS_AUTH_SSO_ROLE or SYS_AUTH_DBA_ROLE.**	**DROP ROLE** *role_name*
Compatibility role that does have underlying system privileges granted to users **The role is successfully dropped if:** • **The role being dropped is not SYS_AUTH_SA_ROLE, SYS_AUTH_SSO_ROLE or SYS_AUTH_DBA_ROLE.**	**DROP ROLE** *role_name* **WITH REVOKE**

Re-creating Compatibility Roles

To re-create dropped compatibility roles, use the **CREATE ROLE** statement and specify the compatibility role name.

Prerequisites

- The MANAGE ROLES system privilege.
- Administrative privileges on all of the system privileges granted to the compatibility role being recreated.

Task

Re-creating SYS_AUTH_DBA_ROLE is semantically equivalent to re-creating both the SYS_AUTH_SA_ROLE and SYS_AUTH_SSO_ROLE roles; you cannot re-create these two roles separately.

When you re-create any compatibility role other than SYS_AUTH_DBA_ROLE, administrative privileges on the re-created compatibility role are automatically granted to SYS_AUTH_DBA_ROLE , as long as SYS_AUTH_DBA_ROLE has not been dropped.

When you re-create any compatibility role other than SYS_AUTH_DBA_ROLE, or SYS_AUTH_PERMS_ADMIN_ROLE, administrative privileges on the re-created compatibility role are automatically granted to SYS_AUTH_PERMS_ADMIN_ROLE, as long as SYS_AUTH_PERMS_ADMIN_ROLE has not been dropped.

To re-create a compatibility role, execute:
CREATE ROLE *compatibility_role_name* [**WITH ADMIN [ONLY]** *userid [, ...]*]

```
CREATE ROLE SYS_AUTH_OPERATOR_ROLE
WITH ADMIN ONLY user1, user2
```

This statement:

a. Recreates the compatibility role SYS_AUTH_OPERATOR_ROLE.
b. Grants SYS_AUTH_OPERATOR_ROLE with administrative privileges to the compatibility role SYS_AUTH_DBA_ROLE, if SYS_AUTH_DBA_ROLE exists.
c. Grants SYS_AUTH_OPERATOR_ROLE with administrative privileges to the compatibility role SYS_AUTH_PERMS_ADMIN_ROLE, if SYS_AUTH_PERMS_ADMIN_ROLE exists.
d. Grants the following system privileges to SYS_AUTH_OPERATOR_ROLE with the NO ADMIN option:
- BACKUP DATABASE
- DROP CONNECTION
- CHECKPOINT
- MONITOR
- ACCESS SERVER LS

e. Grants the system role SYS_AUTH_OPERATOR_ROLE to User1 and User2 with the ADMIN ONLY option.

DBO System Role in a Multiplex Environment

By default, the DBO system role is granted the SYS_AUTH_DBA_ROLE compatibility role, ensure that the DBO system role is granted all privileges necessary to execute multiplex management stored procedures.

If you use the **ALTER ROLE** statement to migrate the SYS_AUTH_DBA_ROLE compatibility role to a new user-defined role, the new role is automatically granted to the DBO system role, provided that SYS_AUTH_DBA_ROLE has not been revoked from the DBO system role.

The SYS_AUTH_DBA_ROLE is immutable. However, once migrated to a new user-defined role, any underlying system privileges can be individually revoked from the new role and granted to other user-defined roles. When this occurs, either the user-defined role to which the system privileges are granted or each individually revoked system privileges must be granted to the DBO system role.

This ensures that all system privileges required to execute multiplex management stored procedures remain granted to the DBO system role.

Backward Compatibility in SAP Sybase IQ 16.0

Grant and revoke syntax for role-based security differs significantly from authority-based security. However, SAP Sybase IQ 16.0 is fully backward compatible with authority-based syntax.

SAP Sybase IQ 16.0 provides well-documented mappings and stored procedures to assist in transition. All stored procedures, functions, and queries created in pre-16.0 databases will continue to run after upgrading.

Stored Procedure to Map Authorities to System Roles

The **sp_auth_sys_role_info** stored procedure generates a report, which maps each authority to a corresponding system role name.

A separate row is generated for each authority. No permission is required to execute the procedure.

Connecting to SAP Sybase IQ 15.x Databases with SAP Sybase IQ 16.0

Role-based syntax is not supported in SAP Sybase IQ 15.x databases.

When using SAP Sybase IQ 16.0 to connect to a 15.x database, only authority-based syntax is valid. Using role-based syntax returns errors. For example, GRANT ROLE returns an error message; GRANT MEMBERSHIP IN GROUP does not.

Beyond this limitation, there should be no change in functionality and no noticeable change in performance using SAP Sybase IQ 16.0 with a 15.x database.

Index