



IQ 15 Best Practices Guide



TABLE OF CONTENTS

Executive Summary	5
What's New in IQ 15.....	6
Engine Parallelism.....	7
Understanding Query Optimizer Changes	8
IQ DATABASE OPTIONS and The Optimizer	8
Summary of Optimizer Changes	9
Additional Query Engine Improvements	9
Improved Tokenization.....	9
Multicolumn HG Indexes	9
Component Integration Services	9
Transaction Out of Space Behavior	10
Storage Changes	11
Hardware Considerations	13
Client Side Loading	14
Other Loading Changes.....	14
User Defined Functions	15
Some tips for using UDFs.....	15
How to disable and enable UDFs.....	15
Time Series	15
Full Text Search.....	16
Multiplex Changes	17
Key Changes from 12.7	17
Coordinator Node.....	17
How to Implement Local Store TYPE Functionality in 15.x.....	18
Inter-node Communication (INC) and SQLRemote	18
MIPC	19
Dynamic Collision	19
Multiplex Do's and Don'ts	20
Distributed Query Processing	21

SA changes.....	22
Catalog (SA) Version Change IQ 15.....	22
New RECOMPILE Clause FOR ALTER PROCEDURE/ALTER FUNCTION	22
Custom Collations.....	22
Message Log Management.....	22
Monitor the Catalog's Disk Space.....	22
Views Dependencies and Schema-Altering Changes.....	23
The REQUEST LOGGING File Content has Changed on 15.....	23
Auditing	24
Temporary Files	24
Client Access and Network Connectivity	26
Client Access	26
ODBC Driver Manager on Unix	27
DBISQL Changes.....	27
IMPORTANT INFORMATION: SDK has been added to the IQ 15.3 install.....	27
Network Connectivity	27
Database and Server Options	28
Purpose of Database Options	28
How to Set Database Options.....	28
How to Control Database Options.....	28
Tips on Usage.....	29
How to Display Database Options Using sp_IQCheckOptions.....	30
Tip: Run sp_IQCheckOptions Before and After a Migration	30
How to Reset Database Options.....	31
Database Options that Affect Performance	31
Tip: For Increased Performance, Consider the Following Database Options	31
Database Options for Collecting Query Performance Data	32
Tip: To Collect Query Plan Information, Set the Following Database Options	32
New Options for IQ 15.3.....	33
Summary of Changed Options for IQ 15.3, 15.2, 15.1 AND 15.0.....	33
Summary of New Options for IQ 15.0, 15.1 AND 15.2.	35
Listing of Deprecated Options for IQ 15.0	35

Operational Management	36
Sybase Central	36
Sybase Control Center	37
Backup/Restore Samples and Scenarios.....	38
Maintenance Operations.....	39
Create events to monitor main and temp DBSpace Usage and Take Corrective Action	40
Data Model Recommendations	42
Security Recommendations.....	44
Authorities	44
User Login Policies	45
IPV6.....	45
Authentication for Sybase Central to Communicate with IQ Agent	45
Advanced Security Option	45
Diagnostics Checklist for Troubleshooting a Problem in IQ.....	47
Information That Should be Collected for ALL Problems	47
Additional Diagnostics That Should be Collected for Some Specific Problems	47
Sysam and Licensing Changes.....	51
Licensing Options in IQ 15	51
Legacy Licensing	52
Tips on SYSAM in IQ.....	52
Appendices	55
APPENDIX 1: Listing of New Options FOR IQ 15.0, 15.1, 15.2 and 15.3.	55
APPENDIX 2: Listing of Deprecated Options for IQ 15.0, 15.1, 15.2 and 15.3	60
APPENDIX 3: sp_dropConnOnMainUsed Procedure	61
APPENDIX 4: sp_dropConnOnTempUsed Procedure.....	63

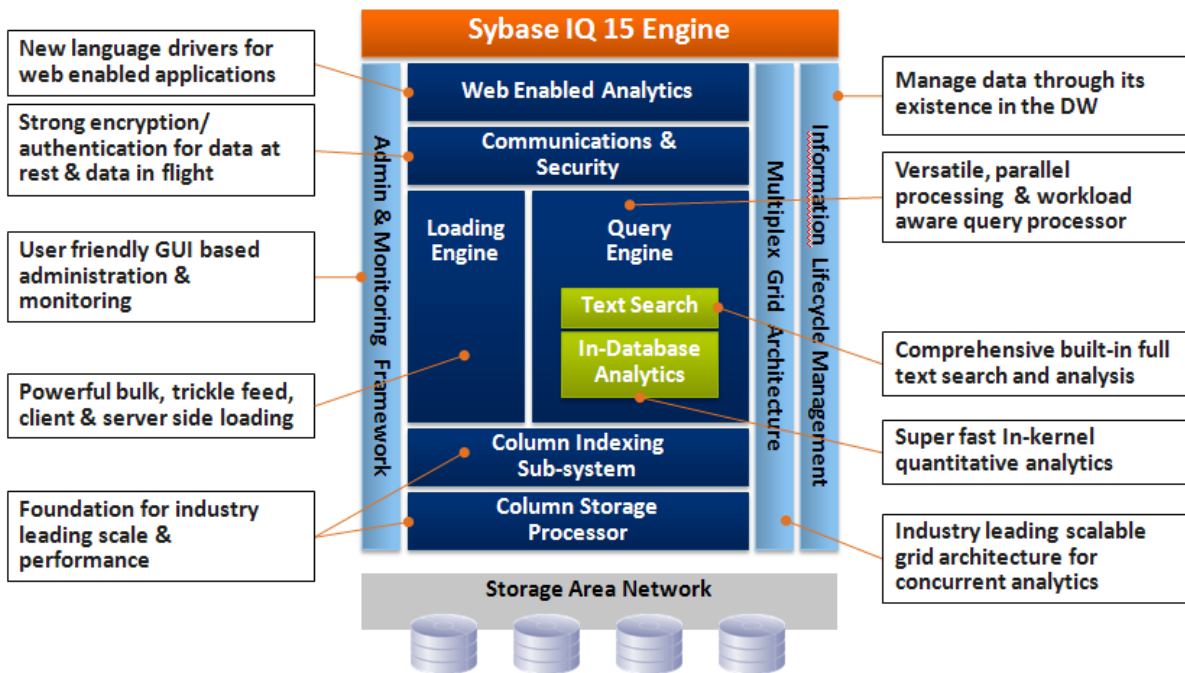
EXECUTIVE SUMMARY

Sybase IQ 15.3 is the culmination of a sequence of releases to offer an analytics platform that supports sophisticated analytics, and scales in multiple dimensions. IQ 15.0 was the first step in March, 2009 to improve the parallelism of IQ running on SMP architectures. IQ 15.1 (July, 2009) brought in-database analytics – statistical algorithms running in the database – near the data – through partner libraries. IQ 15.2 (June, 2010) provided an infrastructure for supporting text analytics – large object storage and retrieval capabilities, indexes designed to quickly locate and score terms, and SQL operators to construct search expressions. IQ 15.3 introduces a new “shared everything” MPP (massively parallel processing) architecture to distribute queries across multiple machines.

This document discusses Best Practices for IQ 15. It is intended to provide a starting point for optimizing your implementation of IQ 15. It includes an introduction to new features, suggested methods for using and configuring important features of IQ, and some precautions for avoiding common problems. As a general purpose document, it is not intended to be a comprehensive guide for every environment. Rather, it is a set of guidelines, suggestions and observations on how to better use IQ in most situations.

WHAT'S NEW IN IQ 15

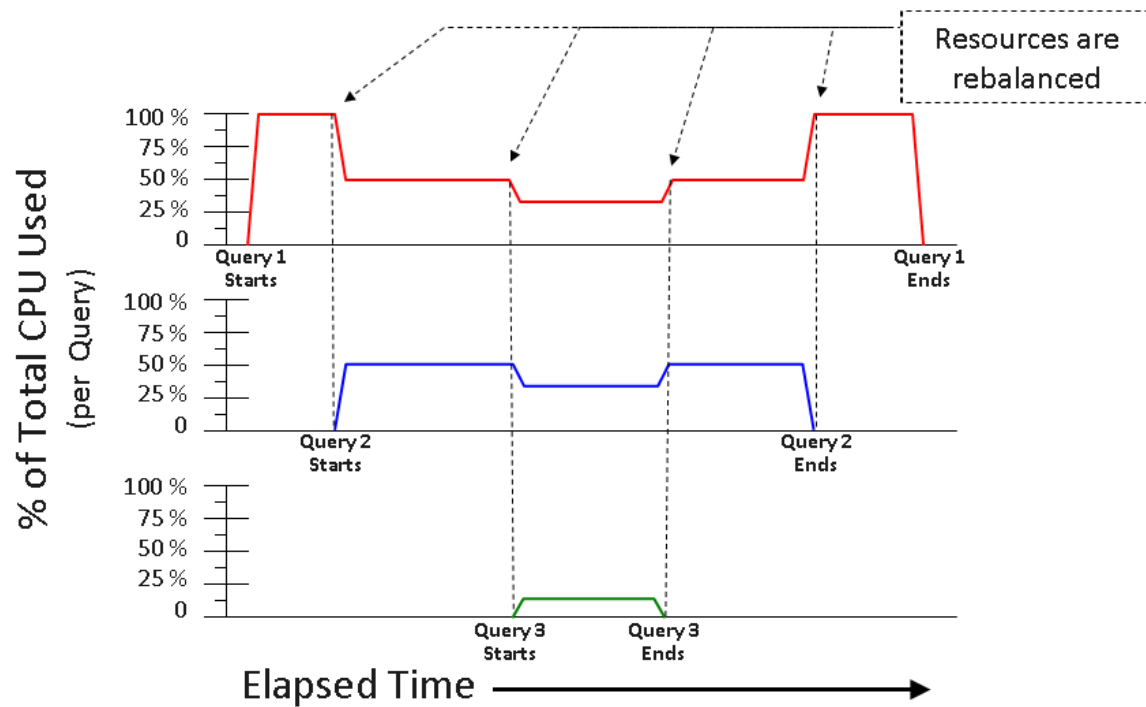
IQ 15 has been one of the biggest releases in Sybase IQ's history. Prior to developing the release, Sybase conducted months of market research involving customer and field interviews, analyst and practitioner viewpoints, competitive studies, partner inputs, and internal ideas. The R&D and QA teams were increased by more than 50% to build the software. Formal beta programs validated and honed the quality of the product. There has been strong passion and commitment behind the release, and aggressive and targeted marketing is ongoing to bring the message to the marketplace. The following diagram illustrates the wide array of capabilities that IQ 15 has to offer:



This document highlights the features that are brand new in the product, and it describes changes and improvements to existing features. Throughout the document, you will be given guidance on how to take full advantage of IQ 15's capabilities.

ENGINE PARALLELISM

Version 15 of Sybase IQ brought about a fundamental change in both load and query operations. The engine has evolved to perform significantly more operations in parallel than before. The parallelism in IQ 15 attempts to strike a balance between single user performance and overall system concurrency. This is achieved through a dynamic reallocation of resources at runtime. To illustrate, the first user in will get all cores needed for the load or query operation. The second user in will not wait for the first user to complete, but rather the engine and optimizer will pull resources away from the first user so that both users can equally consume the CPU resources. The third user in would experience the same behavior; with all three operations consuming roughly one-third of the system each. As an operation comes to completion the resources are freed up and given back to any other currently executing tasks that can benefit from more CPU resources. The following diagram illustrates this process:



UNDERSTANDING QUERY OPTIMIZER CHANGES

IQ 15 performs more query processing in parallel, which means there are more threads allocated to a particular task. More threads require more memory - not only from main and temp cache, but from the operating system.

There are various server and database options that you can set to both allocate sufficient memory and to throttle parallelism to match the resources of your particular machine. Other than -iqmt and -iqtc settings that can be set at server startup to size the main and temporary caches, other options to tune parallelism are:

1. `max_query_parallelism` – This option sets an upper bound which limits how parallel the optimizer will permit query operators, such as joins, GROUP BY, and ORDER BY to be. The value should be \geq of cores.
2. `max_iq_threads_per_team` - This option controls the number of threads allocated to perform a single operation. It should be at least 2X larger than the number of cores.
3. `max_iq_threads_per_connection` - This option controls the number of threads for each connection. It should be larger than `max_iq_threads_per_team` (2 x `max_threads_per_team` or more)

If you have older versions of IQ in your environment, you will want to keep old and new IQ 15 query performance timings and plans available for comparison purposes.

IQ DATABASE OPTIONS AND THE OPTIMIZER

In general, turn on the `QUERY_PLAN` option only when needed to collect query plans for analysis purposes. Keeping this option ON will grow the `.iqmsg` file size at a rapid rate.

Make sure the option "`FORCE_NO_SCROLL_CURSORS`" is turned ON. In rare situations where you need backwards scrolling cursors, you might need to turn it off.

When loading data, turning ON "`APPEND_LOAD`" can improve load performance. With the `APPEND_LOAD` option OFF, IQ will fill in the holes left when rows are deleted. When ON, new rows will go to the end of the table on new pages. When `APPEND_LOAD` is OFF, versioning is higher because more existing pages will get modified. When `APPEND_LOAD` is ON, new pages from the free list will be filled with new data. The holes left by deletes will be compressed out when stored on disk. However, when they are read into the IQ main cache, they are uncompressed and will take up more space compared to full pages for the same number of rows.

Having too many pages that are partially full could be viewed as a form of fragmentation, and using `APPEND_LOAD` OFF will help reuse the holes left by the deletes. There are always tradeoffs: the load with `APPEND_LOAD` ON will run faster because it will use less resources during the load:

- Fewer pages are read in from IQ main store
- Fewer pages are written out to the IQ main store (fewer pages modified = less versioning)

Turn on "`Disk_Striping`" to allow IQ to write to all available disk stripes during a write operation

Avoid usage of "`xp_cmdshell`" calls, if possible within IQ, because some of these calls can be expensive.

While performing an INSERT...LOCATION operation, increase the communication packet size to 4K, if the data volume is large (the default is 512 bytes on most platforms): `INSERT LOCATION PACKETSIZE 4096`. Verify if the remote ASE (`sp_configure default package size/ max package size`) and/or IQ (`-p sever option`) can handle this package size.

Consider using the "MINIMIZE_STORAGE" option for most of IQ data storage. If you do this, FP(1), FP(2) and FP(3) indexes that use lookup tables will be created instead of flat FP indexes. These take up less space and decrease I/O, improving the performance of queries.

SUMMARY OF OPTIMIZER CHANGES

IQ 15.x offers many optimizer configuration parameters which can help improve queries. Please refer to the IQ manuals (Reference: Statements and Options), for details on these various options. Make sure you understand the configuration parameters before making any major change, because server wide configurations can cause undesirable impact on other queries. Instead, you might want to set options temporarily for particular queries.

There are also various optimizer hints available in the query plan. Make note of them, and implement the advised changes if query performance is unsatisfactory.

ADDITIONAL QUERY ENGINE IMPROVEMENTS

IMPROVED TOKENIZATION

FP index tokenization has been implemented for columns with more than 64K distinct values. This is called an FP(3) index, and is structurally similar to the FP(1) and FP(2) indexes. The maximum size of the FP(3) lookup table is 16777216, and each lookup key requires 3 bytes of storage. The creation of an FP(3) index is permitted only if the space used by the lookup table is less than the current value of the `FP_LOOKUP_SIZE` database option, and less than the portion of the main cache specified by the current setting of the `FP_LOOKUP_SIZE_PPM` database option.

The FP(3) index can offer significant performance improvement for queries using extremely large tables with high unique cardinality.

MULTICOLUMN HG INDEXES

An additional feature that boosts query performance is a new multicolumn HG index for ORDER BY queries that reference multiple columns.

COMPONENT INTEGRATION SERVICES

Sybase IQ uses Component Integration Services (CIS) to query tables on remote servers. Changes in Sybase IQ 15.2 allow queries with proxy tables or IN SYSTEM tables to execute significantly faster than earlier versions when the amount of IN SYSTEM or proxy table data is small compared to the IQ data.

Queries that benefit from these changes include SELECT, SELECT INTO, or INSERT...SELECT statements only.

Queries that do not benefit from these changes have the following characteristics:

- References to system catalog tables
- A subquery in a CASE expression
- LOBs (columns of LONGBINARY, LONGVARCHAR, LONGNVARCHAR, LONGBITSTRING, or XML) on remote tables
- A function not supported in the remote node (for example, “bit_or” or user defined functions)
- Queries with global variables

TRANSACTION OUT OF SPACE BEHAVIOR

Main and temp store space management has improved such that when a transaction runs out of space, IQ will terminate it and roll it back, rather than suspend it. Then the DBA can add more space.

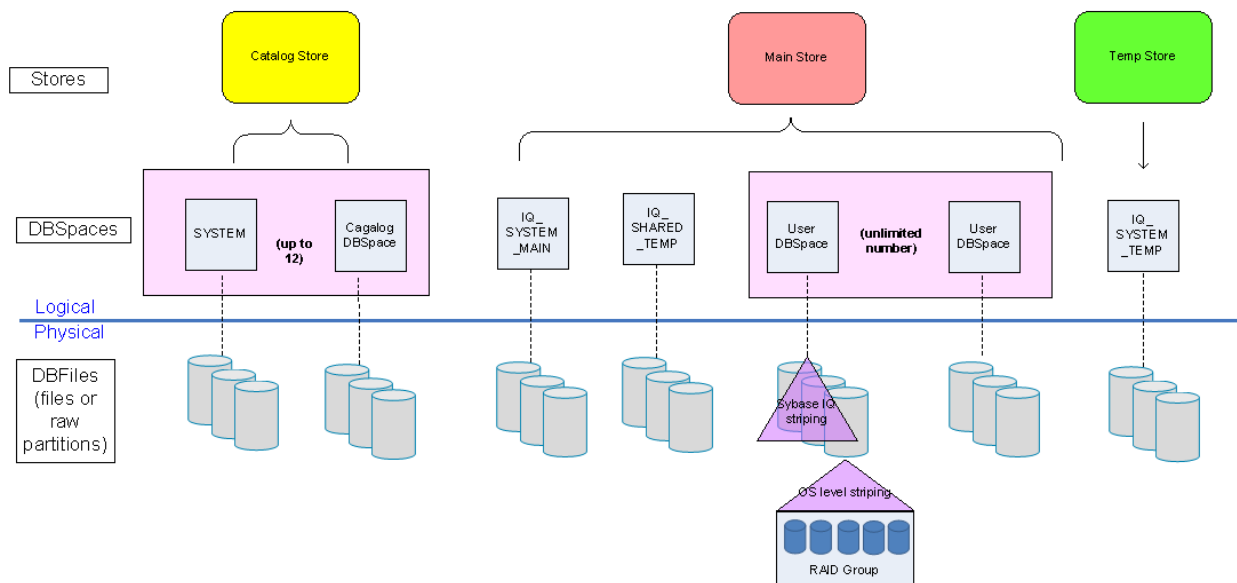
STORAGE CHANGES

In IQ 12.7, a DBSpace was a flat file or raw device. In IQ 15, that has changed. A DBSpace is now a logical storage container for data.

Sybase IQ 15 lays out a database as a set of logical stores:

- Catalog store: stores database metadata
- Main store: stores system structures and user data
- Temp store: stores temporary data generated during loads and queries (sorting results, etc.) and user defined temporary tables

Stores are composed of logical containers called DBSpaces. A DBSpace is composed of multiple DBFiles which map to individual files or raw storage space. Within the main store, a pre-defined DBSpace, called IQ_SYSTEM_MAIN, holds system structures. IQ_SYSTEM_MAIN is no longer designed to contain user data. With the licensed VLDB option, the user may define an unlimited number of DBSpaces for user data. Without the VLDB option, a user may define a single user data DBSpace with multiple DBFiles within it. Here is a depiction of the storage layout for Sybase IQ 15:



It is highly recommended that IQ_SYSTEM_MAIN not be used for user data. The DBSpace and file free list are maintained in IQ_SYSTEM_MAIN. Versioning space, some node to node communication, the TLV replay, and other system housekeeping is also done in this DBSpace. By default, 20% of IQ_SYSTEM_MAIN is reserved for these purposes. When adding space to IQ_SYSTEM_MAIN, an entire Multiplex must be shut down and the nodes synchronized. For these reasons, it is best to leave IQ_SYSTEM_MAIN as a system area with little or no user data.

When a database object is written to a DBSpace and the DEFAULT_DISK_STRIPING option is 'ON', Sybase IQ automatically distributes data across all available DBFiles in the DBSpace. The user may also set up disk striping across RAID groups at the operating system level.

With IQ 15.3, you will have an additional DBSpace to be used as the shared temporary store for distributed query processing (see [Distributed Query Processing](#) section).

A user DBSpace can be *read-only* or *read-write*. A DBFile can also be marked as read-only or read-write. With this separation, the database administrator's job is more efficient:

- Backup read-only data just once
- Backup/restore a read-only DBSpace independently of read-write DBSpaces
- Restore of read-only data can be performed while the system is operational
- Perform data validation on just the read-write portions of the database

A DBSpace can also be *online* or *offline*. The DBA can mark non-openable and non-usable DBSpaces as offline, and IQ will ignore them until they are brought online again.

The Sybase IQ 15 VLDB option provides partitioning and placement features. A table can be partitioned by specifying a column partition key, and specifying ranges of values of that key. Data within each range of values is relegated to a partition. At this time, range partitioning is the only partitioning scheme that is supported. A data partition includes the base FP (Fast Projection index in IQ) data only, and not the other indexes that exist along with that data.

Partitioning allows data that should be grouped together to be grouped together: e.g., data that shares a common attribute. Partitioning also separates data that should be separated, such as historical data from current data. A partitioned table can be altered to:

- Add/drop partitions
- Split a partition
- Merge adjacent partitions
- Un-partition the table

After partitioning, an administrator can place a database object in a particular DBSpace, or move a database object into a different DBSpace. Database objects are tables, table partitions, columns and indexes. With flexibility of data positioning, frequently accessed data can be assigned to faster storage, and less frequently accessed data can be segregated to cheaper, slower storage. This strategy controls storage costs, while still delivering performance.

HARDWARE CONSIDERATIONS

The increased parallelism and storage management capabilities in IQ 15 allow it to take advantage of more CPU and memory resources, and to utilize tiered storage effectively. Best practices for hardware sizing are well documented in [“A Practical Hardware Sizing Guide for Sybase IQ 15”](#).

CLIENT SIDE LOADING

The IQ bulk loader now handles client side loads. With LOAD TABLE, the input data source file may now exist on either the client or the server. If the file is on the client, issue the command with the USING CLIENT FILE option. Client side loading is supported by clients that use the IQ/SQL Anywhere database drivers: ODBC, JDBC using the iAnywhere driver, OLEDB, ADO.NET, Perl, PHP, and Python. It is not supported by the TDS protocol used by Open Client and JDBC for jConnect.

LOAD TABLE USING CLIENT FILE supports all load options, including LOB support. It is a true bulk loader and performance approximates a server side LOAD TABLE plus network latency to transfer the source data over the network.

In a client side load, the client application opens the file and then sends data packets across the network. Each packet contains a portion of the file. The packets are consumed by the server in memory without recreating the file on the server side.

Client side loading avoids cluttering up the file system on which the server is running. Furthermore, the average user may not have any privileges to write a file on the server file system.

To improve security of client side loading, use Transport Layer Security (TLS) to encrypt data across the network. Additionally, the database administrator can control or disable client side loading with various layered security mechanisms that are available. A client side load shouldn't be any slower than a server side load with FTP time taken into account. With a server side load, you have to wait for the file to be fully written to the remote server before loading can commence. With a client side load, data loading happens immediately. You can also increase the packet size to help with network transmission.

Note: that the iq_bcp utility has been deprecated, and is replaced by the LOAD TABLE USING FILE statement.

OTHER LOADING CHANGES

Load performance on a single machine has improved due to the parallelization of High_Group (HG) and Word (WD) index creation.

In a Multiplex environment, loads can be executed in parallel against different tables by multiple writer nodes. Read more about that in this section: [MULTIPLEX CHANGES](#).

USER DEFINED FUNCTIONS

A user-defined function (UDF) is a mechanism for extending the functionality of a database server by adding a new function that can be evaluated by the database server within SQL statements. In order to use the new scalar or aggregate functions you need to have the UDF license which comes with "RAP - The trading Edition: Enterprise" or a certified partner library.

SOME TIPS FOR USING UDFS

UDF code should be compiled using C++ and not C. Refer to the Sybase IQ "User-Defined Functions Guide" for various compiler switches that should be used during compilation on different operating systems.

By default, UDFs are disabled in a Multiplex for the coordinator and the writer and only enabled for the readers. To enable it on writer or coordinator you need to explicitly add the switch "-sf -external_procedure_v3" to the params.cfg file and restart the server.

HOW TO DISABLE AND ENABLE UDFS

Administrators can enable version 3 UDFs for any server by specifying

```
-sf -external_procedure_v3
```

in the server startup command or in the configuration file.

Administrators can disable **version 3** UDFs for any server by specifying

```
-sf external_procedure_v3
```

Note: version 3 is a Sybase Internal API version for UDFs.

Sybase strongly recommends that customers only run UDFs on a multiplex reader node to eliminate any risk of UDF-induced corruption of the database.

BIT, and TEXT data types are NOT supported as arguments or return types.

LONG VARCHAR and LONG BINARY are supported ONLY as Input Parameters for UDFs in IQ 15.3.

TIME SERIES

Time series is a sequence of time points, measured at successive times and normally spaced at uniform intervals. Time series forecasting uses a model to forecast future data points based on past data points. Sybase IQ includes SQL functions for time series forecasting and analysis. These functions in turn declare user-defined functions (UDFs) that execute within the IMSL™ C Stat and C Math external libraries.

The time series forecasting and analysis functions include both OLAP-style aggregates and supporting scalar functions. These functions are available only with RAP.

FULL TEXT SEARCH

Sybase IQ 15.2 introduced the Unstructured Data Analytics (UDA) option, this option provides the means to store and retrieve unstructured data objects from the IQ store. IQ's full text search capability enables users to search for both words and phrases, perform Boolean and proximity searches and score results based on relevance. Sybase IQ uses TEXT indices to store positional information for terms in an indexed column. TEXT indexes are created using settings stored in a text configuration object. Here is an example of creating a TEXT index using a SQL command:

```
CREATE TEXT INDEX myTxtIdx ON Customers ( CompanyName ) CONFIGURATION default_char
```

Sybase IQ 15.2 ESD #2 added support for NGRAM TEXT indices. An NGRAM is a sequence of variable characters that stands for a word or string of words. Fuzzy search capability over an NGRAM TEXT index can be used to search for misspellings or variations of a word.

Sybase IQ can use external pre-filter and term-breaker libraries to pre-filter and tokenize documents during index creation or query processing. External pre-filter and term-breaker libraries must be provided by a Sybase-certified partner. Currently the only certified partner is ISYS. The ISYS Document Filters for UDA enable a user to create text indices against unstructured data stored in the IQ store. IQ text indices then facilitate the searching of unstructured data using SQL. ISYS document filters support most common document formats such as Microsoft Word (DOC, DOCX), Excel (XLS, XLSX), PowerPoint (PPT, PPTX), Adobe PDF, WordPerfect, Rich Text Format (RTF), Open Document Format (ODF), HTML and many others.

Full text searches are performed using the CONTAINS clause in the FROM clause of a SELECT statement, or by using the CONTAINS search condition (predicate) in a WHERE clause, e.g.:

```
SELECT ID, ct.score, Description
FROM MarketingInformation
CONTAINS ( MarketingInformation.Description,
          'stretch* | comfort*' )
AS ct ORDER BY ct.score DESC;
```

Here are some additional resources that discuss Sybase IQ full text search:

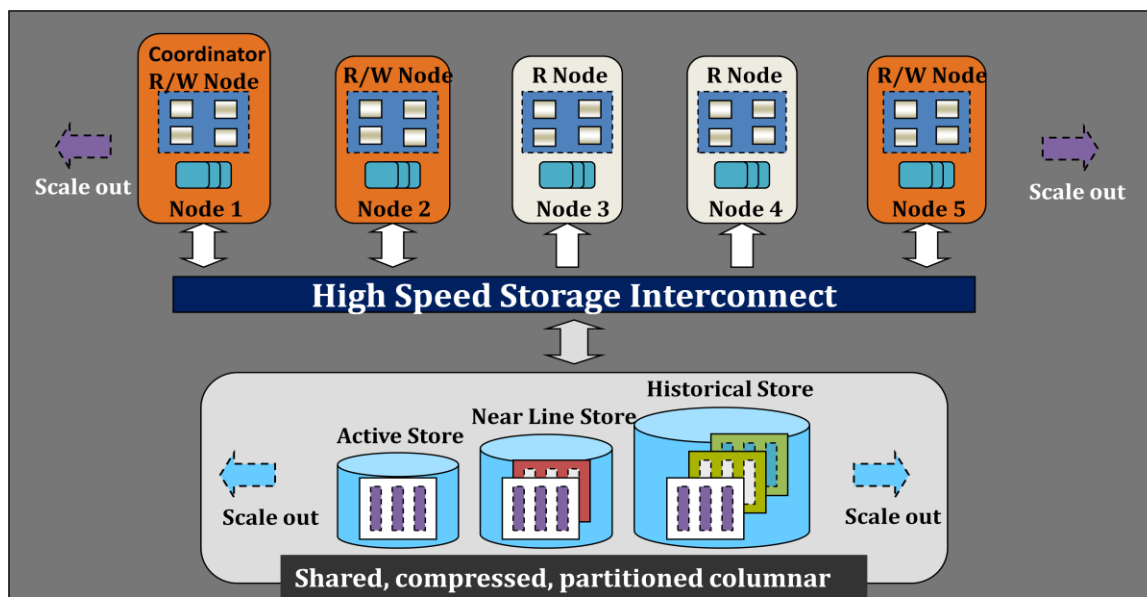
[“Using Sybase IQ as a Text Search Engine for ASE”](#)

[“SYS Prefilter 9.7 plugin running in Sybase IQ 15.2, supporting Full Text Search”](#)

MULTIPLEX CHANGES

A Sybase IQ Multiplex is a group of servers, each running Sybase IQ, that connect to a central store for permanent shared data. Each server maintains its own, local storage for catalog metadata, temporary data, and transaction logs.

In IQ 12, only one node in the Multiplex was allowed to modify shared IQ data. An IQ 15 Multiplex now includes a single coordinator, and permits multiple writers as well as multiple readers. This allows for distribution and parallelization of read-write DML operations (INSERT, LOAD, UPDATE, DELETE) on shared IQ tables. As with IQ 12, though, any object can have at most a single user changing the data or structure. In systems with a lot of tables that are undergoing constant data refreshes, it is possible to architect a system where there are two or more write nodes. The sole responsibility of the write nodes is to handle data changes, while N number of other nodes are defined as query, or read, nodes to handle all read operations in the database.



KEY CHANGES FROM 12.7

- No Local Stores in 15
- No shared IQ store path aliases
- No SQLRemote dependency

COORDINATOR NODE

One node is the heart of the complete multiplex setup - the 'Coordinator'. It is a single designated node in the IQ multiplex, which has the ability to coordinate all RW operations, including schema changes, in the entire multiplex.

The primary server, or coordinator, manages all global read-write transactions and maintains the global catalog and metadata. The table version log (TLV log) is also maintained by the coordinator. It stores information about DDL operations and communicates information about new table versions to the secondary servers (readers or writers).

One thing to note is that all DDL operations are run on the coordinator regardless of which node the SQL was actually initiated on. Typically, this is not an issue. However, when creating indexes on fully populated tables, the actual work is performed by the coordinator, not the write node that where the command was run.

This is an important point to keep in mind when sizing the host for the IQ coordinator. First, place the coordinator on an isolated host that is sized for all DBA maintenance and any real-time DDL operations such as *create index* that may take place. Second, have the coordinator also play the role of writer.

Functions performed by coordinator:

- Coordinate shared IQ table locking between nodes
- Maintain shared IQ storage management
- Maintain global catalog
- Maintain metadata updates for DML and any DDL changes on IQ or SA store objects and sync them across all nodes
- With the implementation of distributed query processing in IQ 15.3, the coordinator is the only node that has the ability to add DBFiles to the IQ_SHARED_TEMP DBSpace, and also administer logical servers.

HOW TO IMPLEMENT LOCAL STORE TYPE FUNCTIONALITY IN 15.X

Another important change in an IQ 15 Multiplex, is that there are no more query server *local stores*. When migrating an IQ 12 database to IQ 15, DBAs need to consolidate data in the local stores into the main shared store. A utility called *iqsunload* is provided to assist you with this. A combination of login policies and DBSpace management in the shared store will provide the node level isolation that local stores used to provide. Here is a process for setting up “local store type” functionality:

- Create a User DBSpace.
- GRANT RESOURCE to certain users.
- Create a login policy that will “LOCK” these users on all servers except from the writer nodes where storage isolation has been implemented.
- Assign this login policy to these users. Only this select group of users will be able to create tables and give select permissions.
- These users will be able to login only into the specified servers.
- Load, inserts, deletes, updates and selects can then be restricted to a specific server.

INTER-NODE COMMUNICATION (INC) AND SQLREMOTE

In IQ 12, communication between secondary nodes and the writer node was handled by SQLRemote processes. In IQ 15 and later, this communication between the secondary nodes (readers and writers) and the coordinator is handled via a new Inter-node Communication (INC) layer. INC provides TCPIP based communication between the nodes with the help of remote procedure calls. The link consists of heartbeat and pooled connections.

The INC communication occurs between the Coordinator and secondary nodes via a dedicated connection. This “dbo” owned connection is established at the time a secondary node starts up and will remain active for the entire duration the node is active. The coordinator monitors this connection to check for secondary nodes’ aliveness. This connection is only used for non-transactional communication. For transactional communication between the coordinator and the secondary nodes, INC will use a connection from the INC connection pool which can be customized via `MPX_CONNECTION_POOL_SIZE` and `MPX_MAX_UNUSED_POOL_SIZE` options. INC reports messages in the `.iqmsg` file.

INC connections are used by all secondary nodes for DDL commands and R/W DML operations. For each user connection on the secondary server doing a DDL or R/W operation, an INC connection gets “associated” with the user connection doing that operation until that command commits or rolls back. Then it goes back into the pool. If these transactions are short lived then many user connections can do DDL or R/W operations successfully with the default value of option `MPX_MAX_CONNECTION_POOL_SIZE`. However if there are too many user connections active at the same time doing DDL or R/W operations or the transactions are large (example: many user connections are doing loads or updates at the same time without committing for a long time), then the default setting may not be enough and hence you will have to increase the value of this option. The `-gm` server option specifies how many user connections can connect to this secondary server. Then, depending on the application, one needs to assess how many concurrent RW or DDL operations could occur. Based on that assessment, set the `MPX_MAX_CONNECTION_POOL_SIZE` to a reasonable value. The down side of having a large number is that you can end up having too many INC connections to the coordinator. You may want to execute some of the DDL or R/W operations on the coordinator directly.

INC Connection Pool is separate from the users’ connection pool. If IQ runs out of connections from the INC connection pool, you will see this error: “The number of connections in the connection pool have exceeded the total number of connections allowed in the connection pool. No more connections can be allocated.”.

MIPC

In order to support streamlined communication among nodes participating in the distribution of a query, IQ 15.3 introduces the Multiplex Inter Node Communication (MIPC) framework. The MIPC mesh is a peer-to-peer inter-node communication infrastructure that supplements the Inter Node Communication (INC) protocol added in IQ 15.0. MIPC allows Multiplex nodes to talk directly with each other, and supports the more robust communication requirements of DQP.

There are both public and private configuration options for the MIPC. The private option allows you to specify host-port pairs (TCP/IP protocol only at this time) that Multiplex servers will use exclusively for DQP related communications. If no private interconnection configuration is provided, MIPC uses the host-port pairs specified for other types of communication: external user connections and INC connections.

A private MIPC network has been found during internal testing to provide significant performance benefits over a shared MIPC network – in one particular instance, a distributed query running on 2 nodes over a private MIPC network executed almost as quickly as a 3 node configuration using a shared MIPC network.

DYNAMIC COLLISION

Dynamic collisions only occur on secondary nodes. A collision occurs when a schema change is committed while a secondary node is querying the same object. A dynamic collision results in a termination of the connection to the secondary. If such a scenario occurs, then you will see message similar to the following in the .iqmsg file:

```
“Shared IQ Store update DDL statement: drop table DBA.test Disposition: SQLSTATE:42W21 -- dropped 1 connection(s) for table: DBA.test “
```

In order to have minimal disruption to query operations, it is advised to avoid performing schema changes when the multiplex is heavily used.

MULTIPLEX DO'S AND DON'TS

- Avoid the usage of user table in the catalog. Keep it as small as possible as this will have negative impact on synchronization
- IQ Multiplex does not support DIRECT IO, only raw devices for shared IQ stores
- For optimal performance keep .db, transaction.log, iqmsg, and other logs on local disks.
- IQ Main Store DBSpace files can only be modified from the Coordinator.
- The DBA should shutdown all secondary nodes prior to adding or altering a DBFile in the IQ_SYSTEM_MAIN DBSpace. Once done, then the secondary nodes need to be synchronized.
- Adequate space should be assigned to IQ_SYSTEM_MAIN because all persistent transaction bookkeeping data for an object is kept in this DBSpace.
- When performing a failover procedure to the designated failover node, it is absolutely critical that you make sure the current coordinator server is shut down. If you perform failover when the current coordinator is running, then you could end up corrupting the Multiplex.
- Create/Alter/Drop Multiplex Server can be executed from any server as long as the coordinator is running. When executing any of these commands, IQ will not check that a new secondary server has access to the shared DBSpaces. The user should insure that the secondary server has access before it is added to the Multiplex.
- Backing up a database in a Multiplex setup can only be executed on the coordinator.
- Do not run excessive amounts of Global DDL commands on secondary nodes. Use local temporary tables wherever possible as DDL on local temporary tables does not have the INC overhead.
- It is advisable to run large data loads from secondary nodes (writers) instead of the coordinator as it frees up the Coordinator to do global DDL and other bookkeeping activities.
- A designated failover server cannot be excluded unless it is the last un-excluded server in the multiplex.
- Never start a secondary node in single node mode. If you do that when you already have a good coordinator then the Multiplex will think you are promoting the secondary node to the role of a coordinator. Having two coordinators in a Multiplex setup is not possible and could lead to data corruption.
- IQ 15 does NOT support mixed-versions in a Multiplex. All nodes should be at the same ESD level. Also, all nodes should be running the same operating system.
- The size and amount of storage that needs to be set aside for the shared temporary DBSpace depends on the distributed query processing workload. In a worst case scenario you might end up having the sum equal to all the local temporary DBSpaces. As a starting point it is best to provide shared temporary storage equal to ½ the sum of the peak local temporary storage for each node. In order to calculate the peak local temporary storage usage, you can run the sp_iqstatus system stored procedure during a typical query

workload on a node in a pre-DQP setup. To track shared temporary storage you can execute the `sp_iqspaceused` system stored procedure. (See next section: [“Distributed Query Processing”](#).)

- In 15.3 it is recommended that before dropping a member of a logical server or a logical server itself, ensure that there are no active connections for the logical server on affected servers. Use `sp_iqconnection` to examine the logical server context of active connections. (See next section: [“Distributed Query Processing”](#).)

DISTRIBUTED QUERY PROCESSING

Prior to IQ 15.3, Sybase IQ did not have a built-in load balancer and workload manager. The architecture of Sybase IQ was such that an application would connect to a single node in the Multiplex and run the user’s workload against that one node. This allowed the application and database administrators to have tight controls over the IQ Multiplex and direct certain users or applications to specific hosts for true workload segregation. Should that node become unavailable, though, it was up to the application and/or user to connect to another functioning Sybase IQ node.

There are many situations where such a manual process is unacceptable and workload management coding is a difficult, if not impossible, task.

IQ 15.3 mitigates the problem by introducing the PlexQ™ Distributed Query Platform, a Massively Parallel Processing (MPP) architecture that accelerates highly complex queries by distributing work to many computers in a Multiplex configuration. Unlike shared-nothing MPP architectures, PlexQ utilizes a shared-everything approach that dynamically manages and balances query workloads across all the compute nodes in a Sybase IQ MPP configuration. PlexQ’s automatic workload re-balancer aggressively works to avoid contention among users for system resources, thereby providing predictable high performance and resource efficiency for a broad spectrum of concurrent workloads.

Distributed query processing comes as part of the Multiplex Grid option for Sybase IQ, and is enabled by default. You have flexible options for choosing which servers participate in the execution of a distributed query, or whether to turn it off entirely on a global basis or for particular queries. Refer to the [“Using Sybase IQ Multiplex”](#) user guide, and the Sybase technical white paper [“Scaling Out Query Performance with Sybase IQ 15.3”](#).

SA CHANGES

CATALOG (SA) VERSION CHANGE IQ 15

A SQL Anywhere database is used in IQ as the catalog to store the database object metadata, permissions, DBSpace information, and the database/server options for Sybase IQ. Some operations are also done by the SA engine, such as syntax parsing, connections to a remote server (using the CIS layer), request logging, auditing, etc.

In IQ 12, the SA engine used by IQ was SA 9. IQ 15 uses SA 11.

The new SA version requires the IQ catalog to be recreated. If you are upgrading an IQ 12.7 database to IQ 15, you must run a command line program called “iqunload” to migrate the database. This utility updates the IQ catalog only, and does not touch existing IQ DBSpaces. Refer to the [Installation and Configuration Guide](#) for details about how to perform the migration.

NEW RECOMPILE CLAUSE FOR ALTER PROCEDURE/ALTER FUNCTION

Use the RECOMPILE syntax to recompile a stored procedure. This is useful when the schema of a table referenced in the procedure has changed. The definition stored in the catalog is re-parsed and the syntax is verified.

For more information, please check: [ALTER PROCEDURE statement](#).

CUSTOM COLLATIONS

The creation of custom collations on IQ 15 is not supported.

If you have created a database with a custom collation on a previous version of IQ, you can use the iqunload migration utility to upgrade the database to IQ 15 and preserve the custom collation.

MESSAGE LOG MANAGEMENT

A new message log management feature (-iqmsgsz and -iqmsgnum switches) allows the server to control the size of the message log file, and archive it a configurable number of times. This prevents the message file from growing indefinitely.

MONITOR THE CATALOG’S DISK SPACE

When you create new objects in IQ, their definitions are stored in the catalog, and the .DB and .LOG files will grow. You must ensure that the disk where the catalog (.DB file) and .LOG files are located always has available free space.

The new system procedure **sa_disk_free_space** (<system_DBSpace_name>) will return the system DBSpace name, the free disk size and the total disk size.

The system DBSpaces names are: SYSTEM, TRANSLOG, TRANSLOGMIRROR, and TEMPORARY or TEMP.

Run `sa_disk_free_space` frequently, or schedule its execution in a `CREATE EVENT` statement or in an UNIX script to monitor free space.

Examples:

```
call sa_disk_free_space (system); -- return disk info for the disk where the .DB resides
```

```
call sa_disk_free_space (translog); -- return disk info for the disk where the transaction log .LOG resides.
```

```
call sa_disk_free_space (temp); -- return disk info for the disk where temporary files reside.
```

VIEWS DEPENDENCIES AND SCHEMA-ALTERING CHANGES

View dependencies have been redesigned in IQ 15.2. When you have a view referencing another table/view, the link between both objects is now stored in the `SYSDEPENDENCY` table. The system procedure **`sa_dependent_views`** ('view_name', 'owner') shows the dependencies on all the dependent views for a given table/view.

When you alter the schema of the underlying tables, the server will attempt to recompile the dependent views.

If this "auto recompilation" fails, then manual recompilations of some views might be necessary.

Example:

```
create table T1 (C1 int, C2 bit, C3 varchar(10));
create table T2 (C1 int, C2 bit, C3 varchar(10));
create view V1_T1 as
    select C1, C2, C3 from T1 union all select * from T2;
create view V1_V1 as select C1 from V1_T1 where C2 = 1;
create view V2_V1 as select C1 from V1_T1 where C3 = 'A';
```

-- The 1st "level" view V1_T1 has 2 dependent views, thus `sa_dependent_views` returns 2 rows:

```
select * from systab where table_id in
(select dep_view_id from sa_dependent_views ('V1_T1','DBA')); --> 2 rows returned
```

THE REQUEST LOGGING FILE CONTENT HAS CHANGED ON 15.

The request logging file (also called the `zrlog` file) content has changed in IQ 15. If you have a monitoring process on this file, please make the changes described below to access its content. Also refer to the user documentation to learn about the benefits of the new [Request logging](#) feature.

In IQ 15, you generate the request logging file with `-zr all -o zrlog.txt IQ start` parameters as usual. Now load the `zrlog` file into IQ, with:

```
call sa_get_request_times (zrlog.txt);
```

The function `sa_get_request_times` stores the zrllog's content in the temporary table `satmp_request_time` that can be queried. For example, select all the operations done by connection ID 129:

```
select * from satmp_request_time where conn_id = 129;
```

Select all from all connections executed during a period:

```
select * from satmp_request_time
where start_time between '2010-07-26 10:30:00' and '2010-07-28 21:00:00';
```

Select the statement text of the request id 317:

```
select (stmt) from satmp_request_time where req_id = 317;
```

You will get the statement text on one line which might be difficult to read if it is a long statement. You can use the function `sa_statement_text` to format the statement so that individual items appear on separate lines:

```
call sa_statement_text ('<statement>');
```

Notes:

- When you exit from dbisql, `satmp_request_time` is emptied.
- By default, dbisql displays 5 000 rows, and a max of 100 000 rows,
- If you have a large file, you might need to redirect the result set with the OUTPUT TO command.

AUDITING

With the auditing feature turn on, additional data is saved in the transaction log, including:

- All login attempts (successful and failed), including the terminal ID.
- Accurate timestamps of all events (to a resolution of milliseconds).
- All permissions checks (successful and failed), including the object on which the permission was checked (if applicable).
- All actions that require DBA authority.

To enable auditing, log in as DBA and run:

```
SET OPTION auditing = 'On';
CALL sa_enable_auditing_type( 'all' );
```

This will enable auditing for all connections and all types of operations.

For all the auditing types: [sa_enable_auditing_type system procedure](#)

Tip: Combine this with truncating the transaction log, e.g. each day at 12.00.

This will enable you to collect the audit data on a per day basis.

TEMPORARY FILES

There are now more options to control and monitor temporary files. On Unix/Linux platforms, the locations of the temporary files that are created by the IQ server can be controlled by pointing the IQDIR15 variable to the desired directory with the desired permissions.

Using the `-dt` option at server startup switch, you can now specify the location of temporary files as well. If the option is not specified, then the database server will check the following environment variables, in order:

- IQTMP15 environment variable (or SATMP environment variable for SQLAnywhere temporary files)
- TMP environment variable
- TMPDIR environment variable
- TEMP environment variable

If none of these environment variables are defined, then SQL Anywhere will place the temporary files in the current directory on Windows operating systems, or in the `/tmp` directory on Unix.

CLIENT ACCESS AND NETWORK CONNECTIVITY

CLIENT ACCESS

In the past, client access to Sybase IQ was via JDBC (jConnect is Sybase's implementation of JDBC), ODBC, or Open Client only. There are now new data access APIs in IQ 15:

- .NET
- ADO
- Perl (provided by the SQL Anywhere Perl DBD::SQLAnywhere DBI module)
- Python (provided by the *sqlanydb* interface)
- PHP (provided by the SQL Anywhere PHP module)
- OLE DB
- Native Ruby Driver
- Ruby/DBI Driver

(Sybase IQ support for the Ruby driver is provided only for Windows 32-bit and Linux32 (x86) platforms.)

Source code, sample projects, and OS-specific binaries for Perl, Python, and PHP are installed in the %IQDIR15%\SDK directory on Windows or the \$IQDIR15/sdk directory on UNIX. ADO.NET and OLEDB code samples are in the appropriate%ALLUSERSPROFILE%\SybaseIQ\samples\SQLAnywhere folder.

These new APIs make it easier to build and deploy database applications in multiple programming environments.

Most applications written using OpenClient expect an ASE server and its T-SQL behavior as opposed to ANSI-SQL. When writing stored procedures, make sure the code considers the kind of client connections that will execute it. Some options have a different default value depending on whether it is an Open client or ODBC connection:

- ALLOW_NULLS_BY_DEFAULT
- QUOTED_IDENTIFIER
- STRING_RTUNCATION
- ANSI_BLANKS
- ANSINULL
- CHAINED
- FLOAT_AS_DOUBLE

Note that some older versions (pre-OC 15) of Open Client do not support bigint, unsigned int and unsigned bigint.

For the ODBC driver, it is a good idea to set AutoPreCommit to YES. This way, the user will get the latest version of all the database objects they are going to use in the next query. You can set this at the connection level or as a server option.

An ODBC connection will set the following temporary options on connection:

- SET TEMPORARY OPTION Time_format = 'hh:nn:ss';
- SET TEMPORARY OPTION Timestamp_format = 'yyyy-mm-dd hh:nn:ss.sssss';

- SET TEMPORARY OPTION Date_format = 'yyyy-mm-dd';
- SET TEMPORARY OPTION Date_order = 'ymd';

In order to overwrite these temporary options you should set the correct value in the InitString of the ODBC DSN

See [SQL Anywhere Data Source Utility \(dbdsn\)](#) for details

ODBC DRIVER MANAGER ON UNIX

Sybase IQ now provides the *libdbod11* shared object, which can be used on all supported UNIX platforms as an ODBC driver manager.

See ["Using the SQL Anywhere ODBC Driver Manager on Unix"](#) in *SQL Anywhere Server – Programming*.

See [Using ODBC data sources on Unix](#) to create the ODBC.INI file.

DBISQL CHANGES

The language parser in dbisql was changed considerably in the releases since version 9. When executing batch scripts do NOT mix Watcom-SQL and Transact-SQL dialects, otherwise you could get errors like:

```
Syntax error near 'PROCEDURE' on line 6  
SQLCODE=-131, ODBC 3 State="42000"
```

IMPORTANT INFORMATION: SDK HAS BEEN ADDED TO THE IQ 15.3 INSTALL

The following limited versions of OpenClient utilities have been deprecated. Sybase IQ 15.3 once again includes the complete version of OpenClient :

- iqisql
- iqdsedit
- iqdscp (UNIX only)
- iqocscfg (Windows only)

NETWORK CONNECTIVITY

Here are some suggestions for increasing network throughput when accessing an IQ database:

- Packet Sizes: Use the `-p` server option to set the appropriate packet size. A larger packet size allows a larger data set to be transmitted in each packet.
- Data retrieval depends on the speed of the network. The faster the network cards and LAN, the better concurrency you can achieve.

Note also, that Sybase IQ does not support NAS (Network Attached Storage) at this time. It is under review and will be done on a vendor by vendor basis.

DATABASE AND SERVER OPTIONS

PURPOSE OF DATABASE OPTIONS

Database options provide a method for controlling many aspects of IQ's behavior.

Database options enable you to modify and customize a wide range of IQ behavior. Unlike ASE, IQ provides hundreds of database options. Furthermore, since IQ contains only one database, changing a database option affects the entire server. Depending on how you qualify an option, the scope of these changes ranges from temporarily affecting an individual user to permanently affecting all users on the server.

Most database options are available to all users and are dynamic. However, some options set permanent server-wide behavior and are static. For these options, DBA authority is usually required, and a reboot of the server is needed for these changes to go into effect.

Database options control the following types of behavior:

- **Compatibility:**
 - The extent to which the Sybase IQ behaves similar to Adaptive Server Enterprise.
 - Whether errors are generated when SQL does not conform to SQL92.
- **Error Handling:**
 - Controls what happens when a specified error occurs, such as dividing by zero.
- **Concurrency and Transactions:**
 - Controls the degree of concurrency and the details of COMMIT behavior.
- **Performance and Optimizer Behavior:**
 - Controls performance features, such as index usage and optimizer tips.
- **Query Behavior:**
 - Enables or disables specific features and enables "forcing" specific query behavior.
- **Resource Usage:**
 - Controls resource usage such as cache, and execution time.
- **Diagnostics:**
 - Controls the generation of troubleshooting and performance diagnostics.

HOW TO SET DATABASE OPTIONS

TO SET AN OPTION, USE THE SET OPTION STATEMENT:

Syntax:

```
SET [ EXISTING ] [ TEMPORARY ] OPTION  
... [ userid. | PUBLIC. ]option-name = [ option-value ]
```

Examples:

- SET temporary option Query_Plan_As_HTML = 'ON' ;
- SET option DBA.Query_Plan = 'ON' ;
- SET option PUBLIC. Query_Plan = 'ON' ;

HOW TO CONTROL DATABASE OPTIONS

Given the range and degree of control that database options exercise, the use of these options must be precisely controlled. To support this precision, the SET OPTION statement includes syntax that limits the scope, duration and precedence of its effects.

OPTION SCOPE

There are three levels of scope: **PUBLIC, USER AND TEMPORARY:**

- SET PUBLIC: Is a server-wide setting. It is for DBA use only, and it affects all users.
- SET User-id: Affects the user named in the option. If neither a user-id nor PUBLIC is specified, then by default, the option applies only to the user issuing the command.
- SET TEMPORARY: Remains in effect for the duration of the user connection that set the option.
- SET TEMPORARY PUBLIC: Remains in effect until IQ is restarted (requires the DBA authority).

OPTION DURATION

There are several duration levels: **PUBLIC, PERMANENT, TEMPORARY and TEMPORARY *until the next IQ server restart*:**

- PUBLIC: Is a permanent server-wide option.
- TEMPORARY: Is effective immediately and persists until changed or a user's session ends.
- TEMPORARY PUBLIC: Is for DBA use only. It is effective immediately and *persists* until the IQ server is restarted. Upon restart, the option reverts to its previous permanent setting.
- PERMANENT is set indirectly by specifying the option without the TEMPORARY qualifier. It applies to the user or group that sets it, or to all, if set by the DBA.

PRECEDENCE

- TEMPORARY takes precedence over USER and PUBLIC settings.
- USER takes precedence over Public.

DYNAMIC VERSUS STATIC OPTIONS

- Options that affect the entire server may require the server to be restarted before taking effect.
- Options that affect only the current connection generally take place immediately. In many cases, you can change these option settings in the middle of a transaction.

Tip: Do not change options when a cursor is open since this can lead to unreliable results.

TIPS ON USAGE

• SET USER

- To set an option for a particular user or group, specify the user name or group name.
 - **For example:** SET option USER1.Query_Plan = 'ON' ;
- To set an option for current user, specify the option without specifying a user name or group name.
 - **For example:** SET option Query_Plan = 'ON' ;

• SET PUBLIC

- To set an option for every user, use the PUBLIC keyword:

- **NOTE:** The use of the **PUBLIC** option requires **DBA Authority**.
- **For example:** SET option PUBLIC.mpx_autoexclude_timeout = '0' ;

- **SET PERMANENT**

- The permanent setting is implied. Setting an option without using the TEMPORARY key word makes the option value permanent for the user or group issuing the statement and requires DBA authority.
- **For example:** SET option Query_Plan = 'ON' ;

- **SET TEMPORARY (without the PUBLIC option):**

- To set an for the duration of the current user’s connection only, set the option using the TEMPORARY keyword, *without* specifying PUBLIC or group.
- **For example:** SET temporary option Query_Plan = 'ON' ;

- **SET PUBLIC TEMPORARY:**

- If an option is set to TEMPORARY for the PUBLIC group, it remains in effect for as long as the database is running and reverts back to the permanent value when the server is restarted.
- **For example:** SET PUBLIC. temporary option Query_Plan = 'ON' ;

HOW TO DISPLAY DATABASE OPTIONS USING SP_IQCHECKOPTIONS

TO DISPLAY THE DATABASE OPTIONS FOR YOUR SERVER, USE SP_IQCHECKOPTIONS

The **sp_iqcheckoptions** procedure displays a list of the current and default values of database options that have been **CHANGED** from the default:

- When sp_iqcheckoptions is run as DBA, it lists all options set on a permanent basis for all groups.
- When sp_iqcheckoptions is run as a user, it lists temporary options set for DBA and those temporary options set by the current user.
- All users see non-default server start-up options.

Sample Output for sp_iqcheckoptions:

User_name	Option_name	Current_value	Default_value	Option_type
'DBA'	'Query_Plan_As_HTML'	'On'	'Off'	'Permanent'
'DBA'	'Query_Plan_Text_Access'	'On'	'Off'	'Temporary'
'DBA'	'Query_Plan_Text_Access'	'On'	'Off'	'Permanent'

TIP: RUN SP_IQCHECKOPTIONS BEFORE AND AFTER A MIGRATION

Run sp_iqcheckoptions BEFORE and AFTER an upgrade to verify options settings:

- This will enable you to verify that your post-migration option values are correct.
- This is critical, since in some cases, options may be reset to their default values as a result of the upgrade process. Therefore, you will need to identify these options and return them their desired values as needed.

HOW TO RESET DATABASE OPTIONS

TO RESET AN OPTION, SET THE OPTION WITHOUT A VALUE

To return an option to its original or default value, set the option without a value.

- For example, do the following to reset the “query_plan_as_html” option:

```
SET OPTION query_plan_as_html =
```

THE RESET VALUE

When an option-value is omitted from the set option statement, the option value is changed as follows:

- If the option-value was a personal option setting, the value reverts back to the PUBLIC setting.
- If the option value was TEMPORARY, the option setting reverts back to the permanent setting.

DATABASE OPTIONS THAT AFFECT PERFORMANCE

TIP: FOR INCREASED PERFORMANCE, CONSIDER THE FOLLOWING DATABASE OPTIONS

This section presents database options that DBA's may want to change from IQ's default settings.

SCROLLABLE CURSORS:

- SET OPTION public.Force_No_Scroll_Cursors='on';
 - By default, all query results are buffered in Temp Cache) to permit scrolling back and forth through all result rows. If the result set has millions of rows, this may seriously degrade performance. Therefore, it is typically recommended to disable this option (by setting it “ON”).
 - If specific users require this functionality the option may be set OFF for those users.

QUERY TEMP SPACE:

- SET OPTION public.Query_temp_Space_Limit='0';
 - This option limits the maximum amount of Temporary Cache that can be consumed by a query based on estimates of the query plan. SQL statements that exceed this limit fail with an error message.
 - Setting this option to 0, removes this limit.

QUERY PLAN:

- SET OPTION public.Query_Plan='Off';
 - This option prevents printing of the query plan to the bdata.iqmsg log file.

MINIMIZE STORAGE

- SET OPTION public.Minimize_Storage='On';
 - This option restricts how much space data access methods can consume.
 - It is recommended that this option be set as 'On' before creating any tables in a database.

- **NOTE:** Turn this option off when loading very wide tables (1000 columns or more) and many columns.

COMPATIBILITY

- If writing stored procedures or embedded application code, make sure to explicitly make settings for compatibility as these options will get set to different values for Open Client vs. ODBC connections.

TABLE LOADS AND UNLOADS

- “Disable_RI_Check” = “On”;
 - When loading data from ASE or other sources, you may want to disable referential integrity checking by turning the option “ON”.
- APPEND_LOAD = “On”;
 - This option controls how new rows are inserted into existing tables.
 - The default (OFF) causes all new rows to be inserted into any empty row ids in a table (as a result of earlier deletes) before appending.
 - Table loads tend to be faster if APPEND_LOAD is set ON. If your application rarely deletes rows it is recommended setting this option ON permanently.
- LOAD_MEMORY_MB (deprecated as of IQ 15.2):
 - Beginning in IQ 15.2, all delimited and binary data load operations use IQ temporary cache for load memory. Fixed width loads continue to use load memory as specified by LOAD_MEMORY_MB. See “[A Practical Hardware Sizing Guide for Sybase IQ 15](#)” for sizing guidelines.
- ROW_COUNT =
 - Note: Unlike ASE, this option **ONLY** controls the number of rows returned from a SELECT statement and has no impact on the size of the UPDATE or DELETE commands.

DATABASE OPTIONS FOR COLLECTING QUERY PERFORMANCE DATA

TIP: TO COLLECT QUERY PLAN INFORMATION, SET THE FOLLOWING DATABASE OPTIONS

- SET temporary option TEMP_EXTRACT_NAME1='QueryName.out' ;
- SET temporary option query_plan_as_html_directory = 'directory/path';
- SET temporary option TEMP_EXTRACT_DIRECTORY='directory/path';
- SET temporary option Query_Plan = 'ON' ;
- SET temporary option Query_Plan_As_HTML = 'ON';
- SET temporary option DDL_Information = 'ON' ;
- SET temporary option Query_Detail = 'ON';
- SET temporary option DML_Options10 = 'ON';
- SET temporary option QUERY_PLAN_TEXT_ACCESS = 'ON' ;
- SET temporary option Query_Timing = 'ON';
- SET temporary option Query_Plan_After_Run = 'ON';

NOTE:

- If Query_Plan="ON" is set globally, the IQ MSG file will grow quickly.

- To reduce output size, you can set Query_Plan_As_HTML without setting Query_Plan.

NEW OPTIONS FOR IQ 15.3.

Below, is a listing of the database options that were added in IQ 15.3.

[See Sybase IQ 15.3 - Reference: Statements and Options: Database Options: Alphabetical list of options for more details.](#)

dqp_enabled

Description: This option was introduced in 15.3 to support the new distributed query process (DQP) functionality. It provides a means to disable DQP, since by default, this option is on.

Default: On

SUMMARY OF CHANGED OPTIONS FOR IQ 15.3, 15.2, 15.1 AND 15.0

Below, is a listing of the database options that have different DEFAULT values in version 15 than in version 12.7.

[See Sybase IQ 15.3 - Reference: Statements and Options: Database Options: Alphabetical list of options for more details.](#)

fp_lookup_size

Description: Specifies the maximum number of lookup pages used in Sybase IQ. Controls the amount of cache allocated to the creation of Lookup FP indexes, particularly FP(3) Indexes.

Default: In IQ 15, the default changed from 32767 to 16.

login_procedure

Description: The LOGIN_PROCEDURE option names a stored procedure to run when users connect. You can accept the default or specify a different stored procedure. Do not modify sp_login_environment.

Default: In IQ 15, the stored procedure sp_iq_process_login was taken out of the sequence so the option LOGIN_PROCEDURE calls sp_login_environment which calls sp_tsq_l_environment. IQ 15 also introduced another option (POST_LOGIN_PROCEDURE) which points to a new standard system procedure. IQ 15 further introduces "login policy" as a set of rules applied when a user makes a connection.

main_reserved_DBSpace_mb

Description: Allows control of the amount of space Sybase IQ sets aside in the IQ main store for certain small, but critical data structures used during release savepoint, commit, and checkpoint operations. For a production database, set this value to between 200MB and 1GB. The larger your IQ page size and number of concurrent connections, the more reserved space you need.

Default: In IQ 15, the default changed from 4mb's to 200mb's. However, the default is over written based on a calculation of a minimum of 1% of the last read-write file in IQ_SYSTEM_MAIN or IQ_SYSTEM_TEMP; and a maximum of 50% of the last read-write file in IQ_SYSTEM_MAIN or IQ_SYSTEM_TEMP.

max_iq_threads_per_connection

Description: This option allows you to constrain the number of threads (and thereby the amount of system resources) that the commands executed on a connection can use. For most applications, use the default.

Default: In IQ 15, the default increased from 72 to 144.

max_iq_threads_per_team

Description: Controls the number of threads allocated to perform a single operation (such as a LIKE predicate on a column) executing within a connection. The total for all simultaneously executing teams for this option is limited by the related option, MAX_IQ_THREADS_PER_CONNECTION. For most applications, use the default.

Default: In IQ 15, the default increased from 48 to 144.

max_query_parallelism

Description: Sets an upper bound for parallelism the query optimizer can choose for GROUP BY operations or arms of a UNION, regardless of how many CPUs are available. This option is effective only on GROUP BY operations when PARALLEL_GBH_UNITS is not set. Normally, you would not set this option. However, if you have more than 16 CPUs and you see excessive CPU time spent on system usage, try setting MAX_QUERY_PARALLELISM to a value less than 16. Experiment with this value to determine the right setting for your environment.

Default: In IQ 15, the default increased from 24 to 64.

post_login_procedure

Description: Specifies a login procedure whose result set contains messages that are displayed by the client application immediately after a user successfully logs in. Can be set for a user-id or the PUBLIC group. You can customize the post login actions by creating a new procedure and setting POST_LOGIN_PROCEDURE to call the new procedure. Do not edit dbo.sa_post_login_procedure.

Default: In 15.2, the default procedure changed from DBA.sp_iq_process_post_login to dbo.sa_post_login_procedure.

prefetch_sort_percent

Description: Specifies the percent of prefetch resources designated for sorting objects

increasing this value can improve the single-user performance of inserts and deletes, but may have detrimental effects on multiuser operations. Do not set this option unless advised to do so by Sybase Technical Support.

Default: In 15.2, the default was changed from 50Mb's to 20MB's in 15.2.

query_plan_as_html prefetch_sort_percent

Description: Generates graphical query plans in HTML format for viewing in a Web browser.

Default: In 15.2, the default changed from off to on in 15.3.

query_temp_space_limit

Description: Specifies the maximum estimated amount of temp space before a query is rejected.

Default: In 15.2, the default changed from 2000Mb to 0 (No-limit).

string_truncation

Description: Determines whether an error is raised when a string is truncated.

Default: In 15.2, changed from off to on in 15.0

temp_reserved_DBSpace_mb

Description: Controls the amount of space Sybase IQ reserves in the Temporary IQ Store.

However, Sybase IQ actually reserves the minimum of 200MB and 50% of the size of the last DBSpace.

Default: In 15.2, changed from 2Mb's to 200 Mb's.

temp_space_limit_check

Description: Checks for catalog store temporary space on a per connection basis.

Default: In 15.2, the default changed from Off to On.

SUMMARY OF NEW OPTIONS FOR IQ 15.0, 15.1 AND 15.2.

For a summary of options that were introduced in IQ 15, see [Appendix 1](#).

LISTING OF DEPRECATED OPTIONS FOR IQ 15.0

For a summary of options that were removed from IQ 15, see [Appendix 2](#).

OPERATIONAL MANAGEMENT

Sybase IQ servers can be monitored and administered either using GUI tools or using system stored procedures.

Using Sybase Central and Sybase Control Center GUI tools, you can view the following statistics pertaining to an IQ server :

- CPU usage statistics
- Memory usage statistics
- Cache statistics
- Thread statistics
- Connection statistics
- Request statistics
- Transaction statistics
- Store I/O statistics
- DBSpace Usage
- Network Statistics

Stored procedures like *sp_iqstatus*, *sa_procedure_profile*, *sp_iqsysmon*, *sp_iqworkmon*, etc. can be executed individually or written in a script that can be run on a periodic basis triggered by events. You can also use IQ UTILITIES to monitor buffer cache and other statistics.

Additional OS tools, such as *vmstat*, provided by the OS, also provide useful information on memory and CPU use and disk activity. Some of the OS tools that might assist in debugging issues can be found in the [Diagnostics Checklist](#) section of this guide.

SYBASE CENTRAL

Here are important notes about Sybase Central (SC):

- The Sybase Central GUI version between 12.7 and 15 has changed from v4.3 to v6.0.
- Sybase Central now supports JRE 1.6.0.
- .scRepository is now renamed to .scRepository600.
- The location of .scRepository600 on Windows is now in the directory where the ALLUSERSPROFILE environment variable is pointing to. On Unix, it still is in the directory where SC is installed.
- All the jar files pertaining to Sybase Central have “600” appended to their name. For example, sybasecentral.jar is now sybasecentral600.jar.
- The new topology view for Multiplex servers not only displays the servers graphically, but also gives you the option to add/remove servers, include/exclude servers or change the designated failover node.
- Using the new Performance Monitoring tab for all IQ servers, you can customize the statistics you want to view and also vary the collection rate. In a Multiplex setup, you can view the stats for all, some or one of the servers at the same time.
- You can view ‘Server Messages and Executed SQL’ for the servers you are connected to in SC. This option gives the user the ability to see exactly what SC is doing behind the scenes, thereby assisting with problem debugging.

- The improved Application Profiling wizard allows you to customize what you want to profile. Application profiling allows you to profile stored procedures, functions, triggers and events. It makes recommendations on how to improve performance and captures database activity.
- Connection Profiles now have Import and Export options.

SYBASE CONTROL CENTER

Here are important notes about Sybase Control Center (SCC):

- SCC provides historical and real-time monitoring in a scalable web application. It offers shared, consolidated management of heterogeneous resources from any location, real-time notification of availability and performance, and intelligent tools for spotting performance and usage trends, all via a thin client, rich Internet application delivered through your web browser.
- Sybase IQ server has been added as a supported resource type from SCC v3.1 onwards. This version of SCC is a monitoring tool only. SCC v3.1 supports IQ 15.2 esd 2.1 or later in single node or Multiplex configurations.
- You can manage performance statistics that you would like to collect for the servers you are monitoring. For more details on how to setup SCC and how to configure monitoring of IQ servers, please review the online documentation: [Sybase Control Center for Sybase IQ \(V3.1\)](#)
- Support for administration of IQ servers has been implemented in SCC V3.2. It supports IQ 15.3 or later. In this release you can administer only certain tasks for an IQ server. Over time, everything you can do in Sybase Central will be able to be done in Sybase Control Center.
- Administration and Monitoring tasks you can perform in SCC v3.2 are:
 - Create, start or stop a server
 - Add/Remove/Modify nodes in multiplex
 - Create/Remove/Modify logical servers, add/remove nodes in a logical server.
 - Add/Remove/Modify DBSpaces/DBFiles
 - Add/Remove/Modify users/groups/login policies/authorities
 - Monitor status of servers at the node level or Multiplex level using a Heat Chart
 - Gather/view statistics on as needed basis at the node level or cumulatively at the Multiplex level
 - Setup alerts for SCC to notify you when a resource needs attention
- SCC stores all the information related to all the resources a user has added into SCC, any user preference data, any operational data and/or statistics into a repository (a SqlAnywhere database). This `scc_repository` exists in `<install dir>/SCC-3_2/services/Repository` directory. It is always a good idea to take regular full and incremental backups, especially when one is managing a number of servers.
- What you cannot do in SCC v3.2 is :
 - Add/Remove/Modify objects (tables, views, stored procedures, triggers or events)
 - Add/Remove/Modify Indexes, Join indexes, remote servers
 - Profile / Debug database procedures

BACKUP/RESTORE SAMPLES AND SCENARIOS

In IQ 15, a DBSpace is composed of a set of DBFiles. When creating DBFiles, use logical links to make it easier to move the database storage in case the need arises.

DBSpaces and DBFiles can be Read-Only (RO), Read-Write(RW), and online or offline. You may restrict full, incremental-since-full or incremental backup to just the set of read-write files in the IQ main store. All read-write files will be backed up. An IQ backup may backup a subset of read-only DBSpaces and/or read-only DBFiles in the IQ main store.

A Sybase IQ database can be backed up using one of following methods:

- IQ database backup
- OS-level backup (virtual backup is recommended however)
- Virtual backup and archive backup(for log files)

Sybase IQ provides the following types of backup:

- Full backup : Default, full backup of catalog stores and all used blocks are backup
- Incremental backup: Full backup of catalog store and all blocks changed since the last backup of any kind are backup, Incremental-since-full-backup: full backup catalog store and all blocks changed since last full backup are backup
- Note: the Incremental clause is not allowed for READONLY DBSpaces.
- Virtual backup: Full backup of catalog store and IQ metadata (only information specific to freelist, backup and checkpoint is backed up).
- Virtual decoupled: A virtual backup, where all DBSpaces must be copied after the decoupled backup finishes, followed by a non-virtual incremental backup
- Virtual encapsulated: A virtual encapsulated 'shell command' allows the shell command to be executed as part of the backup. The shell command performs system-level backup of the IQ store. A non-zero status returned from the shell command indicates backup failure and virtual backup returns an error.

In a backup, the IQ temporary store and params.cfg are not backed up, but information needed to recreate the IQ temporary store and metadata is backed up. The DBA should make a copy of params.cfg and save a copy of SYSDBFILE and SYSDBSPACES.

Note: In IQ 15.3 and onwards, a virtual backup for incremental backup and selective backup is disallowed.

In IQ 15, 20% of IQ_SYSTEM_MAIN is reserved for the freelist. The freelist is used to track blocks in use by DBSpaces. It is considered as part of the catalog information. Used blocks for the freelist are backed up, but not reserved blocks, which can be reconstructed easily without a backup of the information.

With a virtual backup, the whole database can be backed up and restored in seconds/minutes, minimizing downtime. SAN technologies provide the capability of creating multiple mirrored copies of IQ DBFile devices. These copies make it possible to offload maintenance tasks, such as consistency checks, on the mirrored copies. The system stored procedure sp_iqfile can be used to create a backup list of files that comprise the database.

The IQ 15 restore command provides a “verify” option to validate database backup archives. The verify option only reads the blocks, and doesn’t do any write operations.

In a restore operation, the IQ store files(.iq), catalog database (.db file) and transaction log (.log file) must not exist in the location to which the database is being restored. If any of these files exist, they should be moved to a different location before starting the full restore.

Backup and restore commands can only be executed on the coordinator in a Multiplex environment.

From 15.2 ESD1 and above, a restore to the disaster recovery (DR) site can be done by starting the utility_db with the DR server name. The DR server no longer has to have the same name as the production server. If the directory structure and DBFile names are exactly the same in the DR environment as in the production environment, then there is no need to drop the secondary servers from the Multiplex.

There are new IQ 15 utilities to provide information about backup and restore activities that have been conducted against the IQ database:

- db_backupheader: reads the backup archive to display the DBSpaces and DBFiles that existed when the backup was done.
- sp_iqbackupsummary: shows all the DBFiles included in a particular backup
- sp_iqbackupdetails: summarizes backup operations performed
- sp_iqrestoreaction : gives information on what restore actions are needed to bring database to a consistent state with a given date.

MAINTENANCE OPERATIONS

This section includes some recommendations for database maintenance operations:

- It is a good idea to have the database schema in a file format in case a situation arises when restoring the database is not an option and you need to rebuild the server. PowerDesigner can be used to generate and back up the database schema for an IQ database.
- The IQ .log and .db files along with .iqmsg and some other files are stored on file system devices. It is important to make sure there is enough space available at any time on these file systems to avoid lack of file system space issues. On Unix platforms, the /tmp file system might be used for temporary work and this directory also needs sufficient space.
- Backup the database when there are major changes in the database as well as at regular intervals. Backups are extremely important for continued operations in case of un foreseen events.
 - Keep copies of the valid database in safe location(s). IQ 15 allows you to verify the database dumps. Verification will inform you of any issues, so you can take corrective action in a timely manner.
 - Always keep a good copy of the catalog .db file. In a Multiplex environment, make sure you have a good copy of the coordinator’s catalog .db file.
 - In order to reduce the time for a backup, it is recommended that you investigate the virtual backup option provided with IQ. This allows you to do more frequent full backups, since they don’t take as long as a regular backup.

- Keep monitoring the .log file size. If it grows too big then take appropriate actions to truncate the .log file. An extremely large .log file can have an adverse impact on database performance.
- IQ 15 has configurable options (-iqmsgsz, -iqmsgnum switches) for keeping the message file size constrained, and making copies of previous versions of the file.
- It is ideal to have a test environment as similar as possible to the production environment. Any modifications should be tested in the test environment for correctness and desired performance before implementing those changes in the production environment.
- For any IQ related errors, contact Sybase technical support as soon as possible, before taking any drastic action. The support staff can guide you properly to avoid major problems.
- Create events in the IQ server that monitor OS space availability. Use the sa_disk_free_space system stored procedure to get this information.

CREATE EVENTS TO MONITOR MAIN AND TEMP DBSPACE USAGE AND TAKE CORRECTIVE ACTION

This section describes stored procedures and events that allow you to monitor main and temporary DBSpace usage, and drop connections that are using too many resources. The stored procedures take the following actions:

- If the **Free** Main / Temp space falls below the threshold value, the stored procedure sp_dropConnOnMainUsed / sp_dropConnOnTempUsed will determine the connection using up the most space in the Main /Temp Store and drop it.
- DBA connections will not be dropped for Main Store by **sp_dropConnOnMainUsed**. **sp_dropConnOnTempUsed** will drop any connection (including DBA connections) using up too much space in Temp store.
- When a connection is **actually** dropped, the stored procedures will write to the log files ('dropConnOnMainUsed.log' and 'dropConnOnTempUsed.log' respectively). The logs will **not** be written to if there are no connections satisfying the above criteria/conditions. These logs will be created where the database (*.db) file is located, unless explicit path names are passed to the stored procedures.

To implement this monitoring capability:

1. Create the stored procedures in your database from a DBA connection using the procedure code in the following sections:

- [sp_dropConnOnMainUsed](#)
- [sp_dropConnOnTempUsed](#)

2. Create events in the database using a DBA connection:

```
CREATE EVENT "dropConnOnMainUsed"
SCHEDULE "dropConnOnMainUsed" START TIME '04:00' EVERY 300 SECONDS
HANDLER
BEGIN
    call sp_dropConnOnMainUsed('dropConnOnMainUsed.log', 90)
END;
```

```
CREATE EVENT "dropConnOnTempUsed"
SCHEDULE "dropConnOnTempUsed" START TIME '04:00' EVERY 300 SECONDS
HANDLER
BEGIN
    call sp_dropConnOnTempUsed('dropConnOnTempUsed.log', 90)
```

END;

For Multiplex environments, you will want to insure that the events are fired only from a writer node. Add the following lines before calling the stored procedures to generalize the events for a Multiplex:

```
declare srvType char(1);
select ServerType into srvType from sp_iqmpxversioninfo();
//Return immediately if Reader Node.
if srvType='Q'
then
    return ;
end if ;
```

NOTE :

Parameters to be configured are :

- * Start time --> 04:00 in the above example . **Set your preferred start time.**
- * Frequency of calling the stored procedure --> 300 seconds (5 min) in the above example. **Set your preferred frequency.**
- * Threshold value ---> Free Space percent (**Not** Used Space). The above example uses 90% . **Set your preferred threshold.**

DATA MODEL RECOMMENDATIONS

This section provides guidance on tuning your database schema for Sybase IQ.

- Proper Data type sizing
 - Use the smallest data types possible for data
 - If time information is not necessary then use DATE instead of DATETIME
 - If data can fit in TINYINT or SMALLINT then use it instead of INTEGER or BIGINT
 - Don't over allocate storage when defining NUMERIC() or DECIMAL()
 - Don't specify CHAR() or VARCHAR() larger than expected maximum length of data
- IQ Unique and Minimize_Storage
 - It is good idea to set Minimize_Storage option to ON before table creation. This allows IQ to optimize the FP indexes for each column.
 - One can use IQ UNIQUE option to force a specific cardinality on a column.
- Null Values
 - It is good idea to specify NULL or NOT NULL for a column, as it allows the optimizer to better guess the join criteria.
 - In IQ, NULL data does not save space on the database page, as it does in ASE.
- Unsigned Data types
 - Use unsigned data types where possible.
 - Comparisons of unsigned data are faster than signed data.
- LONG VARCHAR and LONG BINARY
 - WD and TEXT indexes are the only indexes that are allowed on VARCHAR() data wider than 255 bytes. The TEXT index is the only supported index for LONG BINARY columns.
 - Storage for these data types are allocated in 256 byte chunks.
 - Note that you must have the Unstructured Data Analytics Option license to use data types of LONG VARCHAR or LONG BINARY
 - There are some Sybase IQ functions that return LONG VARCHAR types (REVERSE, SUSER_NAME, UCASE and others). If you use these functions with a SELECT INTO statement, and you don't have the Unstructured Data Analytics Option license, you should use the CAST statement to convert the return value of the function to the correct data type and size.
- Large Object Storage
 - Large objects can be stored in binary or text based objects
 - This option extends long binary data type from a maximum size of 64K to an unlimited size
 - Indexes you can use on LOB are FP, WD and TEXT only.
 - Some functions of value are byte_length64 (returns size of an object) or byte_substr64(returns portions of the object)
- VARCHAR vs. CHAR
 - Use CHAR wherever possible, because storage in IQ is fixed width.
 - VARCHAR types add slight storage overhead for example a VARCHAR(100) will require 101 bytes of storage 100 bytes for the data and 1 byte for the size of data
 - CHAR data is blank padded, VARCHAR is not
- When to use Indexes
 - If you have join columns then you should have HG indexes.

- All searchable columns should either have HG or LF indexes.
- Aggregation columns should have HNG indexes.
- DATE, TIME and DATETIME columns should have DATE, TIME or DTTM indexes.
- If uncertain, place an LF or HG index on the column, depending on cardinality.
- Use Primary Key, Unique Constraint or UNIQUE HG index where appropriate.
- Do NOT have an HNG index on date/time/datetime columns. Replace it with DATE, TIME or DTTM index.
- If data is ONLY returned to the client then indexes are not needed.
- If a column is used for word searching, then place a WD index on it.
- If a column is used for full text searching, then place a TEXT index on it.
- Group By's can take advantage of multi-column indexes, as long as the index completely matches the column and order.
- HG inserts are the most expensive in IQ with respect to the other indexes.
- Keep data types as narrow as possible to improve performance by reducing disk I/O and memory requirements.
- Multi-column primary keys should have an additional LF or HG index placed on each individual column.
- Temporary Tables
 - When you use 'On Commit Preserve Rows' on temporary tables, then rows remain in the table after the transaction has been committed.
 - Temporary tables are available at the current level (parent) and all of its children.
 - A parent cannot see a child's temporary table.
 - Global temporary tables are static across connections and reboots.
- Cursors
 - If you are using cursors, then it generally means row based processing which is not optimal in IQ.
 - IQ is designed for set based processing.
 - If cursors are used then make sure to use NO SCROLL cursors.
 - 'Open with Hold' means the cursor will remain open across transactions. If they are not used, then the cursor is closed when a commit is issued.

SECURITY RECOMMENDATIONS

From its inception, Sybase IQ has offered a secure information environment for its users: authentication, authorization, data encryption, and auditing. The IQ 15 release includes new capabilities for securing the data in your database.

AUTHORITIES

Earlier versions of Sybase IQ supported only two authorities for performing database administrator tasks: DBA and RESOURCE. RESOURCE authority gives a user permission to create and modify database objects. DBA authority enables the user to carry out any activity in the database: create tables, change table structures, create new user IDs, revoke permissions from users, and so on.

DBA authority gives a user the “keys to the kingdom”. A user with the authority to do a backup will also have the power to do all administrative tasks in the database. All power - and potential for failure – is bundled into a single authority.

Sybase IQ rectifies this vulnerability by providing a granular set of authorities, so a user has permission to do only the tasks that he needs to do.

The following table lists the authorities you can assign to users:

Authority	Tasks that can be performed
BACKUP	Backup databases and transaction logs.
DBA	All administrative tasks.
MULTIPLY ADMIN	Multiplex administration tasks – create and drop servers, and change configuration settings.
OPERATOR	Checkpoint databases, drop connections, back up databases and monitor the system.
PERMS_ADMIN	Manage data permissions (except DBA or REMOTE DBA), groups, authorities, and passwords.
PROFILE	Profiling, tracing and diagnostic operations.
READCLIENTFILE	Load data into server from client machine.
READFILE	Use OPENSTRING clause in a SELECT statement to read a file on the server.
REMOTE DBA	Limited set of DBA permissions for a synchronized mobile database user.
RESOURCE	Create and modify database objects.
SPACE_ADMIN	Manage DBSpaces.
USERADMIN	Manage users, external logins and login policies.
VALIDATE	Database, table, index and checksum validation.

WRITECLIENTFILE	Unload data from server onto client machine.
-----------------	--

USER LOGIN POLICIES

In Sybase IQ 15, you can create a login policy that defines the rules to be followed when establishing a user's database connection. The following settings are governed by a login policy:

- Password life time
- Password grace time
- Password expiry on next login
- Locked
- Maximum connections
- Maximum failed login attempts
- Maximum days since login
- Maximum non-DBA connections

When you create a user in the database, you assign a login policy to that user. You can modify login policies, and reassign users to different login policies.

IPV6

Sybase IQ now supports Internet Protocol version 6 (IPv6), which contains addressing and control information to route packets over the internet. IPv6 supports 2¹²⁸ unique IP addresses, which is a substantial increase over the number of addresses supported by its predecessor IPv4.

AUTHENTICATION FOR SYBASE CENTRAL TO COMMUNICATE WITH IQ AGENT

The Sybase IQ agent enables a Sybase Central client to perform remote administrative functions for the Sybase IQ server. Sybase Central communicates with the IQ agent to:

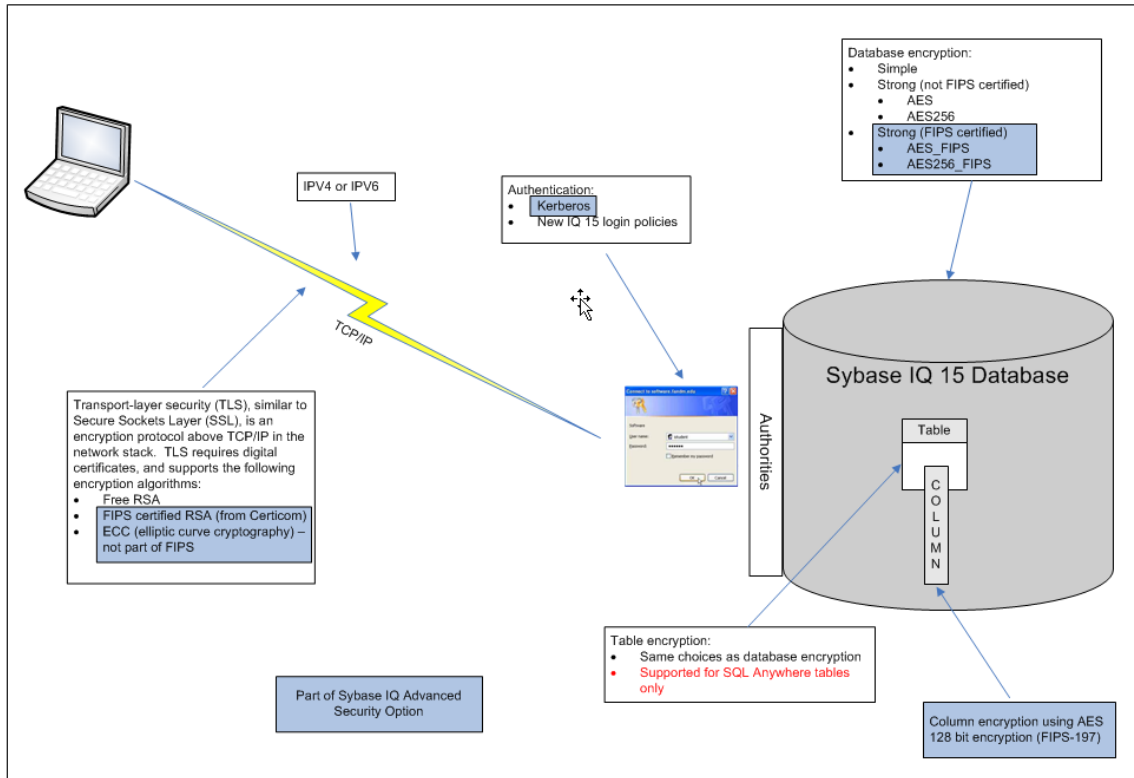
- Start/stop servers
- Access log files
- Perform system functions

An authentication layer has been implemented between Sybase Central and the IQ agent to allow only authenticated clients to access the agent's Remote Method Invocation (RMI) methods to administer the database.

ADVANCED SECURITY OPTION

If you need more secure authentication capabilities, or stronger encryption algorithms, then IQ offers the Advanced Security Option. This licensed option for Sybase IQ includes support for FIPS encryption, database

column encryption, and Kerberos authentication. The following picture shows the security features that are included in the base product, and the features that are included with the Advanced Security Option:



DIAGNOSTICS CHECKLIST FOR TROUBLESHOOTING A PROBLEM IN IQ

This section describes information you should collect from your IQ environment before you call Sybase customer support to help you with a problem.

INFORMATION THAT SHOULD BE COLLECTED FOR ALL PROBLEMS

- 1) Is this problem currently occurring on a Production server or not
- 2) Clear problem description and/or exact error message
 - a. Simplex or Multiplex?
 - b. If Multiplex, which node: coordinator, reader or writer?
 - c. Time of the problem?
 - d. Can the problem be reproduced?
 - e. Is it a non-fatal stacktrace, a hung server or a server crash situation?
- 3) IQ log files (Please collect log files from the last clean boot sequence)
 - a. .iqmsg
 - b. .srvlog
 - c. .stderr
 - d. -zo output file after setting "-zr all" or "-zr sql"
- 4) Has anything changed recently on IQ or the operating system side that might have caused this problem?
- 5) Operating System (OS) information – version, patch level, memory , number of CPUs/cores.
- 6) OS error logs. For example /var/adm/messages or "errpt" output based on the operating system where IQ is installed.
- 7) Complete IQ version output. Execute "start_iq -v2"
- 8) Version of any other tool involved in the problem, e.g. dbisql, OCS, JDBC, Sybase Central Plugin, ODBC, jConnect, etc. Any logs pertaining to these tools.

ADDITIONAL DIAGNOSTICS THAT SHOULD BE COLLECTED FOR SOME SPECIFIC PROBLEMS

Performance

Slow Server Performance

- 1) Diagnostic info from IQ:
 - a. .cfg file used to start server
 - b. exact command used to start server
 - c. output when executing the following using dbisql or iqisql:
 - i. sp_iqstatus
 - ii. select * from sysfile
 - iii. select * from sysiqdbfile
 - iv. select * from sysoptions order by 2,1
 - v. Monitoring output

```
create table iqmontable (c1 int)
go
iq utilities main into iqmontable start monitor '-interval 20'
```

```

go
// run the above for a minute
iq utilities main into iqmontable stop monitor
go
iq utilities private into iqmontable start monitor '-debug -interval
20'
go

// run the above for a minute
iq utilities private into iqmontable stop monitor
go
drop table iqmontable
go

```

IQ will create two output files with "iqmon" in the filename, they will be located in the same directory where the .db file resides.

2) Diagnostics from the Operating System

a. netstat -s

b. netstat -m

c. pstack <process_id>

For simplex server or coordinator and/or secondary node(s) showing problem, gather at least 2-3 "pstack" outputs at least 3-5 minutes apart. The pstack utility may differ based on the operating system that IQ is running on. For example, on AIX the tool is 'procstack'; on Solaris/Linux/HP_UX it is 'pstack'.

d. pmap <PID>

For IQ server(s) showing the problem. On AIX the comparative command is 'svmon -rP'

e. iostat -cdDex 10

f. vmstat -S 5

g. mpstat 180 10

h. Tracing output (**NOTE:** Use these tools with caution and ONLY as a last resort as they are could severely impact performance)

truss (AIX, Solaris etc) = dtrace (Solaris 10+) = strace (Linux) = tusc (HP-UX)

truss -afv all -Dd -o <output_file_name> -p <PID>

Execute the truss command for at least 40 seconds. Use this ONLY as a last resort as it might further slow down the server and in some situations might not provide good results.

strace -f -tt -o <output_file_name> -p <PID>

tusc -o <output_file_name> -c -f -l -p <PID>

tusc -o <output_file_name -ccc -f -l -n -p -v -T "%H:%M:%S" -p <PID>

i. Sometimes 'sar' output might be required.

Slow Query Performance

1) Which client tool is being used to execute the query? For example dbisql, isql, etc.?

2) Which operating system is the client tool running on?

3) What protocol is being used?

4) Was the query scripted, generated or an ad-hoc query?

- 5) What is the SQL for the query?
- 6) What is the schema definition for all the objects involved in the query
 - a. Tables
 - b. Views
 - c. Indexes
 - d. User defined data types
- 7) Generate HTML Query plans in good and slow performance scenarios by setting the following option just prior to executing the query:

```

set temporary option Query_Plan = 'ON';
set temporary option Query_Plan_After_Run = 'ON';
set temporary option Query_Detail = 'ON';
set temporary option Query_Timing = 'ON';
set temporary option Query_Plan_As_HTML = 'ON';
set temporary option Index_Advisor = 'ON';
set temporary option Query_Name = '<Query_name>';
set temporary option Query_Plan_As_HTML_Directory = '<html_plans_directory>';

```

If you do not set Query_Plan_As_HTML_Directory, then by default the html plans will get generated in the same location where the .iqmsg file is created.

Non-Fatal Stacktrace

- 1) The SQL or program that caused the stacktrace
- 2) Did the SQL or program work fine before?
- 3) Can it be reproduced on other servers?
- 4) Can it be reproduced consistently?
- 5) Anything unique on time when it happens?
- 6) Any particular sequence that needs to be followed to repro?

Crash or Fatal Stacktrace

- 1) Can it be reproduced in a simpler repro?
- 2) Can the server be started normally after the crash?
- 3) Collect stktrc*.iq
- 4) Does it appear to be a corruption issue?
- 5) If corruption issue, execute sp_iqcheckdb on the objects involved:

```
sp_iqcheckdb 'verify database/table/index'
```

Hung Server

- 1) Check to see if hung server is due to out of space in some DBSpace?
- 2) Collect at least 2-3 pstack output(s) against the IQ server at least 3-5 minutes apart:


```
pstack <PID>
```

pstack utility may differ based on the operating system that IQ is running on. For example On AIX the tool is 'procstack', on Solaris/Linux/HP_UX it is 'pstack'

Connection Issues

- 1) Are problems specific to a particular client tool?
- 2) Can you ping the server using dbping or tcpip's ping utility?
- 3) Collect client application logs.
 - a. ODBC client - turn on odbc trace
 - b. Open Client – try pinging via dsedit. Setup RIBO and trace
- 4) Ensure server name cache info has correct server information. Locate *sasrv.ini* file. This file contains server information, including server name, protocol, and address. The default location of *sasrv.ini* is *%ALLUSERSPROFILE%\Application Data\SQLAnywhere 11* on Windows and *~/sqlanywhere11* on Unix.
- 5) Try setting up network trace like “tcpdump”

Backup/Restore Issues

- 1) Execute Restore with Verify option to be sure the backup is valid.

SySam Issues

- 1) Output of sp_iqlmconfig
- 2) .lic file if it is an unserved license. Make sure .iqmsg indicates the directory of this .lic file in its search path. For example:
I. 04/12 12:30:18. 0000000000 Using licenses from: C:\IQ\SYSAM-2_0\licenses\SybaseIQ.lic;C:\Documents and Settings\All Users\SybaseIQ\demo*.lic
- 3) SySam server log if it is served license. Check [SySam documentation on troubleshooting tips](#).
- 4) Contents of <IQ dbname>.Imp file (this is located in the same directory where the .db files resides)

SYSAM AND LICENSING CHANGES

Sybase IQ 15 now uses SYSAM 2 for licensing. This means that instead of running sybinstall with the “–add_license” option, you can now just execute the “sp_iqlmconfig” stored procedure and control license management configurations. The advantage of using SYSAM over sybinstall is that you can add licenses dynamically while the server is running. If using SYSAM server and served licenses then DBAs have to track all the necessary licenses in one location only. About how to license your IQ server, review the “Licensing your Software” chapter in the “Installation and Configuration Guide” for the platform of your choice.

LICENSING OPTIONS IN IQ 15

IQ 15.0:

- Introduction of new licensing options:
 - IQ_VLDBMGMT (allowing more than one user DBSpace, and one license for each 1TB of data).
 - IQ_MPXNODE (each secondary node in the multiplex needs this license).
 - (also introduced IQ_TBSIZE for large databases – now removed)
- IQ_ENC license in IQ 12.6/12.7 is now renamed to IQ_SECURITY

IQ 15.1:

- Introduced “In-database Analytics – Partner Solutions” option. This option enhances the scalar and aggregate user defined functions (UDFs). It is only deployed with partner-certified external libraries.
- Small Business Edition is now licensed based on number of sockets instead of number of cores used by the IQ server. A socket refers to a physical connector on a computer motherboard that accepts a single physical chip (CPU). A chip can have one or more cores.
- Removed IQ_TBSIZE option and combined its features into IQ_VLDBMGMT option. If any of the following conditions are true then one needs to purchase the VLDB license – Need storage greater than 1TB; need additional user DBSpaces (1 user DBSpace in addition to IQ_SYSTEM_MAIN comes with the product); need to use table level partitioning.

IQ 15.2:

- New Unstructured Data Analytics option which combines the LOB and Full-Text Search option

IQ 15.3:

- Sybase InfoPrimer, formerly known as Sybase ETL, is now packaged with IQ 15.3 as a separately licensed option.
- If you have the Multiplex license, then you have the added functionality of using DQP (Distributed Query Processing) by default.

Some licenses are checked only upon access of a feature, some during startup, and some do both checks. Note that some features are only checked on startup after the feature has been added to the database (e.g. passed the threshold for IQ store size, created Multiplex writer, etc.) .

Checks on server startup:

- IQ_CORE

Checks on startup after feature enabled:

- IQ_VLDBMGMT
- IQ_MPXNODE

Checks on feature access:

- IQ_VLDBMGMT
- IQ_SECURITY
- IQ_UDA

Refer to the most current Installation and Configuration Guide for information on the feature set and any changes to the licensing behavior.

LEGACY LICENSING

When IQ 15 was released, a new licensing model was introduced. For customers who already had their current servers on the old licensing model, an upgrade to IQ 15 should allow them to continue to use the old licenses that they had already contracted for. In order to provide a seamless upgrade, and clear differentiation of whether a customer is using licenses from an old or new contract, an interim license called '**Legacy License**' was introduced.

When such customers, log into the Sybase Product Download Center (SPDC), they will see something like:

Version	Description	Date	Available
15.1	Sybase IQ Enterprise Edition (Legacy) 15.1 for Linux x86-64 - 64bit	Jul 9, 2009	Download Log

What does this mean?

This customer already has the pre-IQ 15 Enterprise edition license which includes the Multiplex option:

1. Their current entitlements under pre-IQ 15 for existing number of cores (CPs) will be converted automatically with no changes (e.g., if they have a pre-IQ 15 4-core entitlement, they will get an IQ 15 4-core entitlement).
2. They will be entitled to use the Sybase IQ Multiplex Grid functionality with no additional cost for the number of nodes they currently have. If down the road, they are going to add new secondary nodes in the Multiplex, then they will have to purchase additional IQ_MPXNODE licenses.

In this example, when a customer generates licenses for their upgraded IQ server using the Legacy license, they will get the Multiplex grid option at no cost.

TIPS ON SYSAM IN IQ

- New file <dbname>.Imp (license manager property file) is created when you create a new IQ 15 database. This is IQ's human-readable license file and you have one file for each IQ database you create.
- The information in .Imp file should match the IQ install and license information you have in the served or un-served license. Sybase IQ uses only the license that matches the settings in the configuration parameters (PE= and LT=) of the license manager property file. If a server was started with a grace license then the configuration parameters will be set to blank. If you have a valid license then you can edit the file and set the proper values in the .Imp file which match the info recorded for VENDOR_STRING in your .lic file that you have downloaded from SPDC (Sybase Product Download Center).

- IQ will use iq.default.lmp as a template when creating a new dbname.lmp for a new server. This default file exists in \$IQDIR15\sysam directory.
- License checks are performed periodically by the server. The license status is updated only after the next heartbeat cycle is completed. This may take anywhere between few minutes to few hours.
- Sysam's 'lmdiag' option :
You can use the lmdiag option to generate an output file that will assist in diagnosing license checkout problems. Usage:

```
$SYBASE/SYSAM-2_0/bin/lmutil lmdiag -c <license_file_name>
```

- If IQ is installed on a multi-processor AIX box then you might see the following discrepancy in reporting of processors:

From iqdemo.001.srvlog:

```
I. 05/01 10:31:51.
I. 05/01 10:31:51. 14 physical processor(s) detected.
I. 05/01 10:31:51. Running AIX 5 3 on PPC
I. 05/01 10:31:51. Server built for PPC processor architecture
```

From iqdemo.iqmsg:

```
I. 05/01 10:31:53. 0000000000 Using licenses from: /testhost/iq15/SYSAM-2_0/licenses:/testhost/iq15/IQ-15_0/demo
I. 05/01 10:31:55. 0000000000 Checked out license for 7 IQ_CORE (2010.11150/15-n ov-2010/13E7 0944 F517 63F0) will expire Tue Nov 16 00:00:00 2010.
I. 05/01 10:31:55. 0000000000 WARNING: Sybase IQ functionality that requires the IQ_CORE license will be disabled on Tue Nov 16 00:00:00 2010, unless a suitable IQ_CORE license is obtained before that date.
```

This difference is because at server startup on AIX we are reporting the physical CPUs instead of the logical CPUs. This is just a reporting issue. There is no problem with detecting and checking out the correct number of licenses.

- Some other helpful options for debugging Sysam issues:
 - Determine the status of a specific license server:
sysam status -a
 - Determine whether the license server is serving a license for the given feature or not:
sysam status -f <feature_name>

Example: sysam status -f IQ_CORE

```
lmutil - Copyright (c) 1989-2005 Macrovision Europe Ltd. and/or
Macrovision Corporation. All Rights Reserved.
Flexible License Manager status on Thu 5/5/2011 09:51
Users of IQ_CORE: (Total of 10 licenses issued; Total of 0 licenses in
use
```

- Difference between "**Evaluation version**" of IQ and "**Evaluation license**": these two are NOT the same.

Evaluation version (a.k.a Demo version) of IQ means an installation without a license. This can be selected while installing IQ. Once this is selected, the installer will NOT ask you to chose the PE (Product Edition) or LT (License Type). The iq.default.lmp file created will contain empty PE and LT values, i.e.

PE=

LT=

In the evaluation version, all standard and optional features of IQ are available under 'installation grace'. You will always see the "Checked out graced license..." message for each individual license. There is no limit on the quantity of license available under grace. The only restriction is that the licenses expire after the grace period, which is 30 days from the creation date of the database. In case a server has started multiple databases, then it is 30 days from the creation of the oldest database. At expiry, the IQ server shuts down gracefully.

Evaluation license is a valid IQ license. It gets checked out like a regular license (not in grace). A separate license for each feature in use needs to be downloaded and installed, i.e. IQ_CORE evaluation license will allow only the server to come up and if you need to use security feature, you will need IQ_SECURITY license separately.

Evaluation license is only available on request when the potential customer feels the need to evaluate the product/feature beyond the 30 day grace period. If you create a brand new database then you can install it with the evaluation license and get 30 day grace on this new server. You cannot use the same license to start the old server.

- More SYSAM troubleshooting tips can be found in the "Troubleshooting SYSAM" appendix in the "Installation and Configuration Guide" for your particular platform.

APPENDIX 1: LISTING OF NEW OPTIONS FOR IQ 15.0, 15.1, 15.2 AND 15.3.

Below, is a listing of the options that were introduced in 15.0, 15.1, 15.2.

For more details on the following options, please see the documentation as noted for each option:

- 1: *Sybase IQ 15.2- Reference: Statements and Options : Database Options: Alphabetical list of options*
- 2: *SQL Anywhere 11.0.1 Server - Database Administration : Configuring Your Database: Database options: Introduction to database options: Alphabetical list for additional details.*
- 3: *Sybase IQ 15.3- User-Defined Functions Guide: Testing user-defined functions*
- 4: *Sybase IQ 15.2-Time Series Guide: Overview: IMSL Libraries for Time Series Forecasting and Analysis Functions*
- 5: *Sybase IQ 15.2- Installation and Configuration Guide for Windows: Migrating Data: Avoiding potential migration Problems.*

dqp_enabled (1)

Description: This option was introduced in 15.3 to support the new distributed query process (DQP) functionality. By default, this option is on. Setting it to off disables DQP.

Default: On:

allow_read_client_file (1)

Description: Enables client-side data transfer. This option must be enabled to read from files on a client computer. For example using the READ_CLIENT_FILE function.

Default: Off

allow_snapshot_isolation (2)

Description: Controls whether snapshot isolation is enabled or disabled.

Default: Off

allow_write_client_file (2)

Description: This option must be enabled to write files to a client computer, for example using the WRITE_CLIENT_FILE function. .

Default: Off

ansi_substring (1)

Description: When ON, a negative or zero start offset is treated as if the string was padded on the left with noncharacters. When OFF, the behavior of the **SUBSTRING** function is the same as in earlier versions of IQ. Where possible, use the **LEFT** or **RIGHT** functions instead.

Default: On

btree_page_split_pad_percent (1)

Description: Determines per-page fill factor during page splits for B-Tree structures. .

Default: 50

conn_auditing (2)

Description: Controls whether auditing is enabled or disabled for each connection when auditing option is On.

Default: On

default_DBSpace (1)

Description: Changes the default DBSpace for where tables or join indexes are created. Can be set for a group, or user and takes place immediately.

default: (the empty string)

default_disk_striping (1)

Description: Sets default disk striping value for all DBSpaces. Used only by CREATE DBSPACE if CREATE DBSPACE does not specify striping.

Default: ON for all DBSpaces in the IQ main store.

default_having_selectivity_ppm (1)

Description: Provides default selectivity estimates to the optimizer for most **HAVING**.

Default: 0

default_kb_per_stripe (1)

Description: Sets an upper limit on the amount of data written to a stripe before moving to the next stripe. The default value of 1KB means that each operation writes to a different stripe. To write multiple pages to the same stripe before moving to the next stripe, change the *DEFAULT_KB_PER_STRIPE* setting.

This sets the default size for all DBSpaces in the IQ main store.

Default: 1

default_like_match_selectivity_ppm (1)

Description: Provides default selectivity estimates (in parts per million) to the optimizer for most LIKE predicates. If the column has either an LF index or a 1- or 2- or 3-byte FP index, the optimizer can get exact information and does not need to use this value.

Default: 150000

default_like_range_selectivity_ppm (1)

Description: Provides default selectivity estimates (in parts per million) to the optimizer for leading constant LIKE predicates. If the column has either an LF index or a 1- or 2- or 3-byte FP index, the optimizer can get exact information and does not need to use this value.

Default: 150000

enable_lob_variables (1)

Description: Controls the data type conversion of large object variables. Users must be licensed for the Unstructured Data Analytics Option to use large object variables.

Default: OFF

external_udf_execution_mode (3)

Description: Controls the amount of error checking and call tracing that is performed when statements involving external V3 user-defined functions are evaluated. When set to 0, external UDFs are evaluated in a manner that will optimize the performance of statements using UDFs. When set to 1, external UDFs are evaluated to validate the information passed back and forth to each UDF function. When set to 2, external UDFs are evaluated to not only validate the information passed back and forth to the UDF, but also to log, in the iqmsg file, every call to the functions provided by the UDFs and every callback from those functions back into the server.

Default: 0

fp_lookup_size_ppm (1)

Description: Controls the amount of cache allocated to the creation of Lookup FP indexes, particularly FP(3) Indexes. Must be set public,

Default: 2500

http_session_timeout (1)

Description: Specifies the amount of time, in minutes, that the client waits for an HTTP session to time out before giving up.

Default: 30

java_location (1)

Description: Specifies the path of the Java VM for the database.

Default: Empty string.

java_main_userid (2)

Description: Specifies the database user whose connection can be used for installing classes and other Java-related administrative tasks.

Default: DBA

java_vm_options (1)

Description: Lets you to specify options that the database server uses when launching the Java VM specified by the `JAVA_LOCATION` option.

Default: Empty string

materialized_view_optimization (2)

Description: Lets you specify the circumstances under which the optimizer can use stale materialized views. (data becomes stale when any of the base tables referenced by the materialized view is updated.

- **Disabled:** Do not use materialized views for query optimization.
- **Fresh:** Use a materialized view only if it is fresh.
- **Stale** Use materialized views even if they are stale.

Default: Stale

max_client_statements_cached (1)

Description: Controls the number of statements cached by the client.

Default: 10

max_prefix_per_contains_phrase (1)

Description: Specifies the number of prefix terms allowed in a text search expression.

Default: 1

max_priority (1)

Description: Controls the maximum priority level for connections.: High, Above Normal, Normal, Below normal, Low, Background

Default: normal

max_query_tasks (2)

Description: Specifies the maximum number of server tasks that the database server can use to process a query in parallel:

- 0 allows the database server to use as many tasks as it chooses.
- 1 disables intra-query parallelism
- Any other value sets the maximum number of tasks allowed per query
- The number of tasks the database server can use for all requests is also limited by the
The number of users set using `-gn` option and by the number of logical processors available.

Default: 1

max_temp_space (1)

Description: Limits temporary store space used per connection. Enables management of space for both loads and queries. If the connection exceeds the value, IQ rolls back the current statement and returns an error message: "Max_Temp_Space_Per_Connection exceeded."

Default: 0

max_temp_space_per_connection (1)

Description: Controls the maximum amount of temporary file space a connection can use before the request fails with an error message. The `temp_space_limit_check` option must be also be set to On (the default).

- A value of 0 indicates no fixed.
- Any other value specifies the number of bytes.

Description:

mpx_autoexclude_timeout (1)

Description: Specifies timeout for auto-excluding a secondary node on the coordinator node.

Does not apply to the designated failover node:

- 0 indicates that the nodes will not be auto excluded.

Default: 60

mpx_heartbeat_frequency (1)

Description: Specifies interval for the heartbeat thread to wake and clean up the connection pool on the secondary node.
Default: 60

mpx_idle_connection_timeout (1)

Description: Specifies the time after which an unused connection in the connection pool on a secondary node will be closed.
Default: 600

mpx_max_connection_pool_size (1)

Description: Specifies maximum number of connections allowed in the connection pool on a secondary node.
Default: 10

mpx_max_unused_pool_size (1)

Description: Specifies maximum number of unused connections in the connection pool on a secondary node.
Default: 10

oem_string (2)

Description: Stores user-specified information in the header page of the database file.
Default: Empty string

Priority (2)

Description: Sets the execution priority level for requests from a connection. Cannot be higher than max_priority option.
 Settings: Critical, High, Above Normal, Normal, Below Normal, Low, Background
Default: normal

query_mem_timeout (2)

Description: Sets the maximum time, in milliseconds that a request waits for a memory grant:

- -1, the request waits for up to 50 times the estimated execution time for the request.
- 0, the request waits forever.
- Positive number: Sets the maximum time that the request waits.

Default: -1

request_timeout (2)

Description: Controls the maximum time (wall-clock time) that a single request can run.

- 0, requests do not time out.
- Max: 86400

Default: 0

secure_feature_key (2)

Description: Enables features for the connection that were secured using the database server -sf option.

- -sk overrides the -sf option and specifies a key that can be used to re-enable all secured (disabled) features and gives that connection authority to change the features that are secured for all databases running on the database server.
- Can be set as a temporary option only

Default: Null

subquery_caching_preference (1)

Description: Controls which algorithm to use for processing correlated subquery predicates. Normally used for internal testing and does not apply to IN subqueries.
Default: 0 = Let the optimizer choose.

subquery_flattening_percent (1)

Description: Allows the user to change the threshold at which the optimizer decides to transform scalar subqueries into joins. This option only applies to correlated scalar subqueries:

- 0 = The optimizer cost model decides
- 1 to $(2^{32} - 1)$ = The percentage of references at which to flatten.

Default: 100

subquery_flattening_preference (1)

Description: The percentage of references at which to flatten:

Default: 0 = Allow the IQ optimizer to decide to flatten subqueries.

synchronize_mirror_on_commit (2)

Description: Controls when database changes are assured to have been sent to a mirror server when running in asynchronous or asynfullpage mode. Set as temporary or for specific applications by examining the APPINFO string in a login procedure. On = each COMMIT causes changes recorded in the transaction log to be sent to the mirror server and acknowledged.

Default: Off

time_series_error_level (4)

Description: Controls error handling for the time series functions that call the IMSL libraries.

Default: 0

time_series_log_level (4)

Description: Controls error logging behavior for the time series functions that call the IMSL libraries.

Default: 0

tsql_outer_joins (5)

Description: Setting this option to "On" allows use of the deprecated Transact-SQL outer join operators.

Default: On

updatable_statement_isolation (1)

Description: Controls whether connections can send query feedback to the statistics governor.

When set to Off, the statistics governor does not receive feedback about the health of statistics or fix any statistics for the specified connection. However, it does continue to monitor the usage of statistics and create or drop statistics based on their usage. Under normal circumstances, it should not be necessary to turn this option off.

Default: 0

APPENDIX 2: LISTING OF DEPRECATED OPTIONS FOR IQ 15.0, 15.1, 15.2 AND 15.3

Below, is a listing of IQ 12 options that were removed in 15.

#	OPTION NAME	IQ 12.7 ESD#11
1	ansi_integer_overflow	Off
2	auto_commit	Off
3	auto_refetch	On
4	automatic_timestamp	Off
5	bell	On
6	char_oem_translation	Detect
7	command_delimiter	;
8	commit_on_exit	On
9	convert_hg_to_1242	OFF
10	default_having_selectivity	0
11	default_like_match_selectivity	15
12	default_like_range_selectivity	15
13	describe_java_format	Varchar
14	disk_stripping	On
15	echo	On
16	enable_ordered_pushdown_insertion	ON
17	enable_thread_allowance	ON
18	flatten_subqueries	OFF
19	float_as_double	Off
20	headings	On
21	input_format	ASCII
22	iqmsg_length_mb	0
23	isql_command_timing	On
24	isql_escape_character	\
25	isql_field_separator	,
26	isql_log	
27	isql_plan	Graphical
28	isql_plan_cursor_sensitivity	ASENSITIVE
29	isql_plan_cursor_writability	On
30	isql_quote	'
31	java_heap_size	1000000
32	java_input_output	Off
33	java_namespace_size	4000000
34	java_page_buffer_size	50
35	local_kb_per_stripe	1
36	local_reserved_DBSpace_mb	200
37	log_detailed_plans	On
38	log_max_requests	100
39	main_cache_memory_mb	16
40	main_kb_per_stripe	1
41	max_work_table_hash_size	20
42	min_nlpdj_filtered_ppm	2500
43	min_nlpdj_table_size	10000
44	min_smpdj_or_hpdj_table_size	100000
45	mpx_global_table_priv	OFF
46	mpx_local_spec_priv	0
47	mpx_options	0

48	nulls	(NULL)
49	on_error	Prompt
50	optimistic_wait_for_commit	Off
51	optimization_logging	Off
52	os_option_crash	0
53	out_of_disk_message_repeat	120
54	out_of_disk_wait_time	30
55	output_format	ASCII
56	output_length	0
57	output_nulls	
58	parallel_gbh_enabled	ON
59	parallel_gbh_min_rows_per_unit	3000000
60	parallel_gbh_units	0
61	percent_as_comment	On
62	query_plan_on_open	Off
63	quiet	Off
64	return_java_as_string	Off
65	ri_trigger_time	After
66	screen_format	Text
67	sort_phase1_helpers	3
68	sqlconnect	
69	sqlstart	
70	statistics	3
71	temp_cache_memory_mb	12
72	temp_kb_per_stripe	1
73	thread_count	0
74	thread_stack	16384
75	thread_swaps	18
76	truncate_date_values	On
77	truncate_with_auto_commit	On
78	truncation_length	256
79	tsql_hex_constant	On
80	upgrade_database_capability	

APPENDIX 3: SP_DROPCONNONMAINUSED PROCEDURE

```

create procedure
//
// This procedure is to monitor IQ server by checking dbspace usage and to prevent
// the IQ server from running out of main space.
//
// Information written by this procedure into the log is as follows:
//
// 1. Main Store
// 2. Temp Store
// 3. Versioning Space
//
// If usage of Main store reaches the specified threshold,
// IQ server disconnects the connection which holds the most Main space.
// Exceptions: DBA connections never dropped
dba.sp_dropConnOnMainUsed(LogFile varchar(50),FreeMainSpace integer)
begin

```

```

declare maintotal unsigned bigint;
declare mainused unsigned bigint;
declare temptotal unsigned bigint;
declare tempused unsigned bigint;
declare databasename varchar(30);
declare versionsize varchar(255);
declare servername varchar(30);
declare connname varchar(30);
declare TempKB unsigned bigint;
declare connuserid varchar(30);
declare CurrTime varchar(30);
declare MsgText varchar(255);
declare connid integer;
declare blocksizeX2 unsigned bigint;
declare local temporary table m_iq_txn_table(
    TxnID unsigned bigint null,
    CmtID unsigned bigint null,
    VersionID unsigned bigint null,
    State char(12) null,
    TxnCreateTime char(26) null,
    ConnHandle unsigned bigint null,
    IQConnID unsigned bigint null,
    Dbremote bit not null,
    CursorCount unsigned bigint null,
    SpCount unsigned bigint null,
    SpNumber unsigned bigint null,
    MainTableKBCreated unsigned bigint null,
    MainTableKBDropped unsigned bigint null,
    TempTableKBCreated unsigned bigint null,
    TempTableKBDropped unsigned bigint null,
    MainWorkspaceKB unsigned bigint null,
)
in SYSTEM on commit preserve rows;
declare local temporary table iq_status_main(
    Name varchar(40) null,
    Value varchar(128) null,
)
in SYSTEM on commit preserve rows;
set CurrTime="left"(convert(varchar(30),getdate(*),115),16);
execute immediate 'iq utilities main into iq_status_main status';
select substring(Value,cast(locate(Value,'=') as tinyint)+1,length(Value)) into
versionsize
from iq_status_main where name like '%Other%' order by Name asc;
select Value into versionsize from iq_status_main where name = 'Other Versions:.';
call sp_iqspaceused(maintotal,mainused,temptotal,tempused);
set databasename=db_name(*);
set servername=@@servername;
// IQ main store free space > FreeMainSpace then return. If it is < FreeMainSpace
// then drop the connection which is taking maximum main space.
if cast(100-(mainused*100/maintotal) as integer) > FreeMainSpace then
drop table m_iq_txn_table;
drop table iq_status_main;
return

```

```

end if;
select first block_size/512 into blocksizeX2 from SYSIQINFO;
execute immediate 'iq utilities main into m_iq_txn_table command statistics 10000';
//Drop connection only when they could release some amount of space.
select top 1 ConnHandle,
    connection_property('Name',connHandle) as Name,
    connection_property('Userid',connHandle) as Userid,
    max(cast(MainTableKBCreated*blocksizeX2/2 as unsigned
bigint)+cast(MainTableKBDropped*blocksizeX2/2 as unsigned bigint)) as MainWorkSpaceKB
into connid,
    connname,connuserid,
    TempKB from m_iq_txn_table where
MainTableKBCreated > 0 and
Userid <> 'DBA' and
MainTableKBCreated > MainTableKBDropped
group by ConnHandle order by
MainWorkSpaceKB desc;
if connid is not null then
    execute immediate 'drop connection ' || connid;
    set MsgText='echo ' || CurrTime || ' IQ Main free space of: ' ||
        cast(cast(100-(mainused*100/maintotal) as numeric(5,2)) as varchar(10)) || '% '
|| '>> ' || LogFile;
    call xp_cmdshell(MsgText);
    set MsgText='echo ' || CurrTime || ' IQ Temp free space of: ' ||
        cast(cast(100-(tempused*100/temptotal) as numeric(5,2)) as varchar(10)) || '% '
|| '>> ' || LogFile;
    call xp_cmdshell(MsgText);
    set MsgText='echo ' || CurrTime || ' IQ Versioning Size : ' || cast(versionsize as
varchar(30)) || '>> ' || LogFile;
    call xp_cmdshell(MsgText);
    set MsgText='echo ' || CurrTime || ' DBA dropped Connection Handle : ' ||
cast(connid as char(10)) || ', UserID : ' ||
        cast(connuserid as char(10)) || '. Used MainWorkSpaceKB : ' || cast(TempKB/1024
as numeric(10,2)) || ' MB. >> ' || LogFile;
    call xp_cmdshell(MsgText)
end if;
drop table m_iq_txn_table;
drop table iq_status_main
end

```

APPENDIX 4: SP_DROPCONNONTEMPUSED PROCEDURE

```

create procedure dba.sp_dropConnOnTempUsed(LogFile varchar(50),FreeTempSpace integer)
begin
    declare maintotal unsigned bigint;
    declare mainused unsigned bigint;
    declare temptotal unsigned bigint;
    declare tempused unsigned bigint;
    declare databasename varchar(30);
    declare versionsize varchar(255);
    declare servername varchar(30);

```

```

declare connname varchar(30);
declare TempKB unsigned bigint;
declare connuserid varchar(30);
declare CurrTime varchar(30);
declare MsgText varchar(255);
declare connid integer;
declare blocksizeX2 unsigned bigint;
declare local temporary table t_iq_txn_table(
    TxnID unsigned bigint null,
    CmtID unsigned bigint null,
    VersionID unsigned bigint null,
    State char(12) null,
    TxnCreateTime char(26) null,
    ConnHandle unsigned bigint null,
    IQConnID unsigned bigint null,
    Dbremote bit not null,
    CursorCount unsigned bigint null,
    SpCount unsigned bigint null,
    SpNumber unsigned bigint null,
    MainTableKBCreated unsigned bigint null,
    MainTableKBDropped unsigned bigint null,
    TempTableKBCreated unsigned bigint null,
    TempTableKBDropped unsigned bigint null,
    TempWorkSpaceKB unsigned bigint null,
)
in SYSTEM on commit preserve rows;
declare local temporary table iq_status_temp(
    Name varchar(40) null,
    Value varchar(128) null,
)
in SYSTEM on commit preserve rows;
set CurrTime="left"(convert(varchar(30),getdate(*),115),16);
execute immediate 'iq utilities main into iq_status_temp status';
select substring(Value,cast(locate(Value,'=') as tinyint)+1,length(Value)) into
versionsize
    from iq_status_temp where name like '%Other%' order by Name asc;
select Value into versionsize from iq_status_temp where name = 'Other Versions: ';
call sp_iqspaceused(maintotal,mainused,temptotal,tempused);
set databasename=db_name(*);
set servername=@@servername;
// IQ temp store free space > FreeTempSpace then return. If it is < FreeTempSpace
// then drop the connection which is taking maximum temp space.
if cast(100-(tempused*100/temptotal) as integer) > FreeTempSpace then
    return
end if;
set MsgText='echo ' || CurrTime || ' IQ Main free space of: ' ||
    cast(cast(100-(mainused*100/maintotal) as numeric(5,2)) as varchar(10)) || '% ' ||
'>> ' || LogFile;
call xp_cmdshell(MsgText);
set MsgText='echo ' || CurrTime || ' IQ Temp free space of: ' ||
    cast(cast(100-(tempused*100/temptotal) as numeric(5,2)) as varchar(10)) || '% ' ||
'>> ' || LogFile;
call xp_cmdshell(MsgText);

```

```

    set MsgText='echo ' || CurrTime || ' IQ Versioning Size : ' || cast(versionsize as
varchar(30)) || '>> ' || LogFile;
    call xp_cmdshell(MsgText);
    select first block_size/512 into blocksizeX2 from SYSIQINFO;
    execute immediate 'iq utilities main into t_iq_txn_table command statistics 10000';
    select top 1 ConnHandle,
        connection_property('Name',connHandle) as Name,
        connection_property('Userid',connHandle) as Userid,
        max(cast(TempWorkspaceKB*blocksizeX2/2 as unsigned
bigint)+cast(TempTableKBCreated*blocksizeX2/2 as unsigned
bigint)+cast(TempTableKBDropped*blocksizeX2/2 as unsigned bigint)) as TempWorkspaceKB
into connid,
        connname,connuserid,
        TempKB from t_iq_txn_table
    group by ConnHandle order by
        TempWorkspaceKB desc;
    execute immediate 'drop connection ' || connid;
    set MsgText='echo ' || CurrTime || ' DBA dropped Connection Handle : ' ||
cast(connid as char(10)) || ', UserID : ' ||
        cast(connuserid as char(10)) || '. Used TempWorkspaceKB : ' || cast(TempKB/1024 as
numeric(10,2)) || ' MB. >> ' || LogFile;
    call xp_cmdshell(MsgText);
    drop table t_iq_txn_table;
    drop table iq_status_temp
end

```



SYBASE, INC.

WORLDWIDE HEADQUARTERS

ONE SYBASE DRIVE

DUBLIN, CA 94568-7902 USA

Tel: 1800 8 SYBASE

Copyright © 2011 Sybase, Inc. All rights reserved. Unpublished rights reserved under U.S. copyright laws. Sybase, and the Sybase logo are trademarks of Sybase, Inc. or its subsidiaries. All other trademarks are the property of their respective owners. ® indicates registration in the United States. Specifications are subject to change without notice.