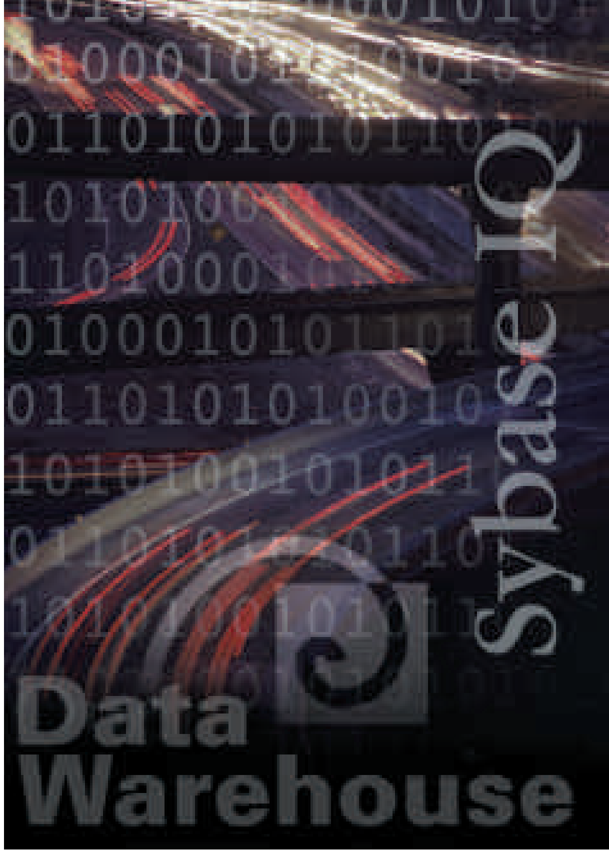
Using Sybase IQ to Implement the Data Warehouse

By John Fotiadis



This is the second half of an article on implementing data warehousing and datamart projects using Sybase IQ. In this segment, the author discusses troubleshooting and incremental refresh strategies, as well as presenting a case study and an overview of the future of data warehousing.

John Fotiadis is group leader for professional services for a Sybase distributor in Greece, where he leads data warehousing and distributed computing projects. He can be reached at johnf@sybase.com.gr.

In the first half of this article (2nd quarter, 1998), we discussed Sybase IQ indexes, query processing, join algorithms, parameters that affect joins, and capacity planning. As users begin to implement data warehouse projects using Sybase IQ, situations will also inevitably arise when IQ does not operate well due to problems in configuration or settings. This article covers a number of common problems, as well as incremental refresh techniques. We also discuss an actual data warehouse implemented recently in Greece, which demonstrates some of the issues which must be explored when creating a large information processing environment.

Troubleshooting Sybase IQ

Following are some hints for troubleshooting Sybase IQ environments relative to the underlying operating system. Finding and resolving problems in IQ has to do basically with the following questions:

Does the catalog server start?

- Check that shared memory is enabled.
- Check that all necessary patches have been installed.
- Check the catalog server log for errors.
- Check $SYBASE has been set.
- On NT, check that %SYBASE%\dll is in the path.
- On NT, make sure there are no dlls in the bin directory.
- For more help, check the Troubleshooting Guide.

Is the catalog server able to accept connections?

- Check that the version of Open Client installed has not been corrupted and check the version of the communications library: % isql –v.
- On NT, try pinging the server using sybping or dsedit.
- Test the open client connection:

```
% isql -U<user> -P<password> -
   S<CatalogServer>
```

- Common problems connecting to the Catalog Server are:
  1. Same Port Number appears multiple times in the $SYBASE/interfaces file or (on NT) the sql.ini file or the interfaces file has been corrupted.
  2. $SYBASE directory has been overwritten or compromised.
  3. Check that the environmental variable $LANG either has not been set or has been set to a correct language (set to 'c' by default on Digital UNIX).

Can IQ connect to catalog server?

- Sybase IQ contains a primitive unsupported user interface, similar to isql, that a user can use to test the connection. Semicolons are used to execute a line (not go):

```
% sybase_iq -U<users> -P<password> -CS<CatalogServer>
```

- Sometimes this will require setting an environmental variable:
  On Sun and Digital:

```
% LD_LIBRARY_PATH=$SYBASE/lib/iq
```

  On HP-UX:

```
% SHLIB_PATH=$SYBASE/lib/iq
```

  On AIX:

```
% LIBPATH=$SYBASE/lib/iq
```

- Current directory does not have write permissions.
- Disk is full.
- $TMPDIR is set to an invalid directory.
- The catalog server does not have the Sybase IQ objects loaded.
- The catalog server has an old version of the Sybase IQ objects loaded. Use IQinit to (re)load the catalog server.
- Must have set the environment variables before running.

Can open server start ?

- Add the '-V no' (verbose) option to the list of parameters for more information from the Open Server.
- startopen_iq is a shell script whose progress can be viewed by running the script with the '-x' option.

```
% cd $SYBASE/bin
% sh -x startopen_iq <OpenServer> -CS<CatalogServer> [<etc>]
```

- For NT, there is a startopen_iq.bat that will set up and display the environment before starting open_iq. This runs in a DOS environment.

```
C:\> startopen_iq <OpenServer> -CS<CatalogServer> [<etc>]
```

- In running startopen_iq, names of the catalog server and Open Server have been reversed.
- The system has not been configured with enough shared memory.
- Too much shared memory has been allocated for an indexspace.
- On HP-UX, memory is allocated when the Open Server is started.

Can I connect to the open server?

- Test open client connection:

```
% isql -U<user> -P<password> -S<OpenServer>
```

- isql_trace is an undocumented primitive tool that allows the end user to see more of the communication between the Open Server and isql_trace:

```
% isql_trace -U<user> -P<password> -S<OpenServer>
```

- If the catalog server has a different version of the Sybase IQ objects than the engine expected, use IQinit to resolve this.
- The interfaces file (sql.ini) file is corrupted or has a port number assigned to multiple servers.
- The catalog server is not running.

Can I create/use an indexspace?

- The utility check_iq is useful for verifying indexspaces. It scans IQmaster for all segments, then performs a test read on each segment.

```
% check_iq -U<User> -P<Password> -S<CatalogServer> [-D
        <Database>]
```

- $TMPDIR is set to an invalid directory.
- Partition/segment is full.
- Not all segments of the indexspace have been mounted.
- The indexspace files *.DB, *.FL, or *.TI have been deleted.
- Database/Indexspace does not exist.

Sybase IQ Incremental Refreshing Strategies

Currently, there are two ways to load Sybase IQ from operational data sources: by using attached Sybase databases (meaning an Adaptive Server database or any non-Sybase data source that can be accessed through the Adaptive Server CIS layer or OmniCONNECT), or from flat files that have been extracted previously from operational data sources. In either case, we have to implement a batch process that loads the whole or specific parts of the data. The cost of loading a data warehouse is high due to the expense of periodic "refreshing" of Sybase IQ. There is an upper threshold, based on the size of the database, that impacts the volume and frequency with which a data warehouse can be rebuilt or refreshed.

To meet the demands of scalability and information currency, other approaches can extend these scalability limits. Most information stored in a data warehouse is changed infrequently; therefore, one methodology is to capture only the data that has changed. This will generally result in fewer updates, enabling larger warehouses to be supported with more current information. We'll see how, by using Distribution Director, a Sybase IQ database can be loaded incrementally from a staging database populated with data from operational databases via Replication Server.

Replication Server provides the ability to capture only the incremental changes in OLTP databases, continuously feeding target databases with standard SQL statements for the appropriate database transactions and statements (insert, update, delete) performed at the database being replicated. On the other hand, proving the ability to update Sybase IQ with only changed data should significantly reduce maintenance costs. As the next figure shows, Replication Server sends changes to a staging database containing tables that represent all the changes that have occurred at the primary site since the last IQ load cycle.

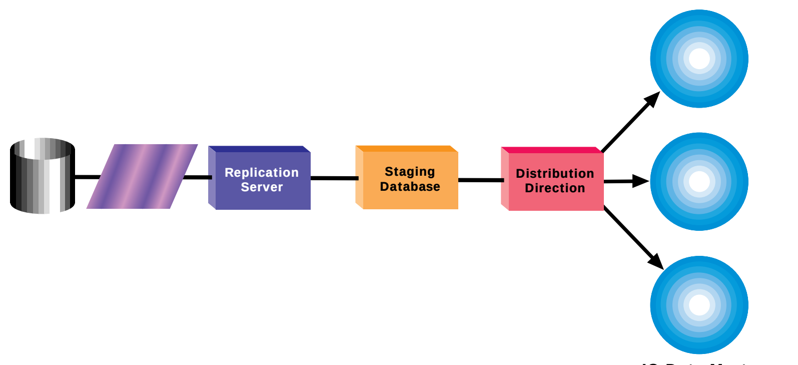


Figure 1 – Distribution Director

In order to maintain replicate staging tables, it is necessary to create a custom Function String Class specifically for this purpose. These function strings allow Replication Server to maintain an "Insert" and "Delete" staging table for each primary table from which changes are being replicated. From a GUI front end, the user can initiate a load of IQ via Distribution Director. At this point, the staging tables are read and the changes stored there are applied to the IQ database. Before this process starts, Replication Server's connection to the staging database must be suspended.

After all the changes in the staging tables are applied to the IQ database, they must be truncated and Replication Server must be reconnected to the staging database. This process ensures that all rows captured and applied to IQ are transactionally consistent. At this point, we have to mention that it is desirable to use Distribution Director to apply changes in staging tables not created by Replication Server. These staging tables would have the same format as the ones for replication, but would be maintained by third-party transformation programs.

Sybase IQ has two technical requirements that involve enhancing the standard methodology used by Replication Server to update target databases:

1. A Sybase IQ database is unavailable for read access while updates are being made to the database. This means that the database is off-line while any update activity is occurring. The final set-up needs to support on-demand selection of update processing, with incremental table changes to maximize availability of the database to readers. In this case, it is desirable that Replication Server send replicated data to staging tables in an intermediate Sybase database, rather than directly to Sybase IQ.
2. Sybase IQ does not support ANSI-compliant DML statements. For example, Sybase IQ does not support the update statement, while the insert/delete statements of IQ are non-standard SQL. Therefore, support must be provided to correctly format Replication Server output into an appropriate structure for updating a Sybase IQ database. In other words, each update sent from the Replication Server to the staging database needs to be decomposed into a Delete and an Insert.

A strategy used to load/delete rows in Sybase IQ using Distribution Director provides better performance and minimizes administrative needs. Database maintenance of IQ should be done on sets of data. For example, loading a set of rows is more efficient than loading one row at a time, and deleting a set of rows is more efficient than deleting a row at a time. As updates are only applied periodically, it would also be more efficient to filter redundant commands so that only the last action for a row is applied to the database. Also, for each staging table there should be only one delete entry and one insert entry for each row in indexes. This must be the case no matter how many times the row is updated, inserted, or deleted in the primary table since the previous load of IQ.

As Sybase SQL Server is provided for catalog services with Sybase IQ, we can use the Catalog Server as the temporary directory for meta-data or "staging" data as required.

Implementing a Warehouse from C-ISAM and Oracle Data Sources

Following we present a case study of a data warehouse system implemented last year in Athens, Greece, using Sybase IQ 11.2 as the data warehouse server. The system has been in production since mid-1997, and the customer is the largest electronic and home appliance equipment company in Greece.

At the first phase of the project, the operational data of the customer were in C-ISAM (80%) and ORACLE (20%) data sources. The volume of the data in operational systems was about 10GB. In this phase, we used a full-loading data warehouse refresh strategy allowing the whole IQ indexspace to be created every weekend. The data are extracted from operational data sources in three ways:

1. Using OmniCONNECT on HP-UX and attaching to C-ISAM files. In this step, IQ is loaded via attached database from OmniCONNECT. This means that even though C-ISAM is not a Sybase database, IQ can attach to them and consider it as a Sybase database using OmniCONNECT.
2. Flat files are extracted using C-programs from C-ISAM files in parallel while IQ is loading via OmniCONNECT as stated previously. With this technique, there is a type of pipelining in the IQ loading process, since IQ is loaded continuously even in the case where no flat file is created.
3. When loading via OmniCONNECT (for the C-ISAM portion of data) and flat files is finished, IQ is loaded from Oracle data, via DirectCONNECT 10.2 for Oracle. Again, DirectCONNECT for Oracle is attached to Omni-CONNECT, and IQ is loaded from an attached database.

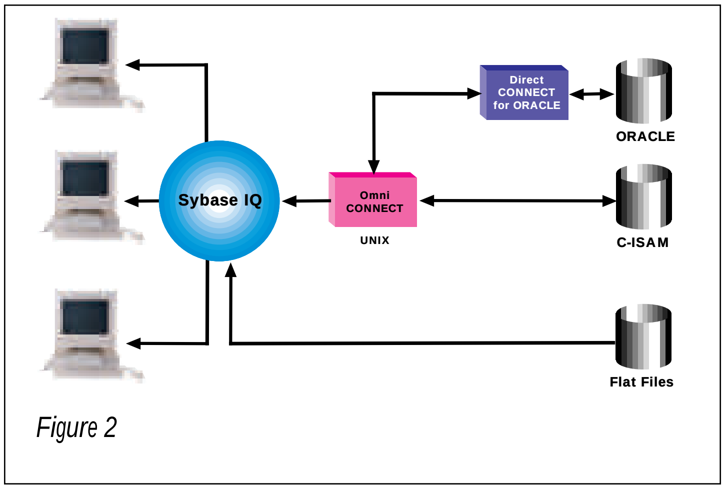


Figure 2

Using this technique, the Sybase IQ indexspace needs five to six hours to be created from scratch in a HP-K370 system with four processors and 1GB of RAM. Figure 2 represents the solution in first phase of the project.

For the next phase of the project, the customer will move all the operational data to an Oracle database and hence has a homogeneous environment for its data in the enterprise. In this case, we intend to change the IQ refresh strategy to an incremental loading scenario. We therefore proposed Distribution Director technology, to be used with Sybase Replication Server on HP-UX and a Replication agent for Oracle to capture changes on operational data in Oracle database. The following figure presents the topology in an incremental loading scenario from an Oracle operational database to Sybase IQ data warehouse server.

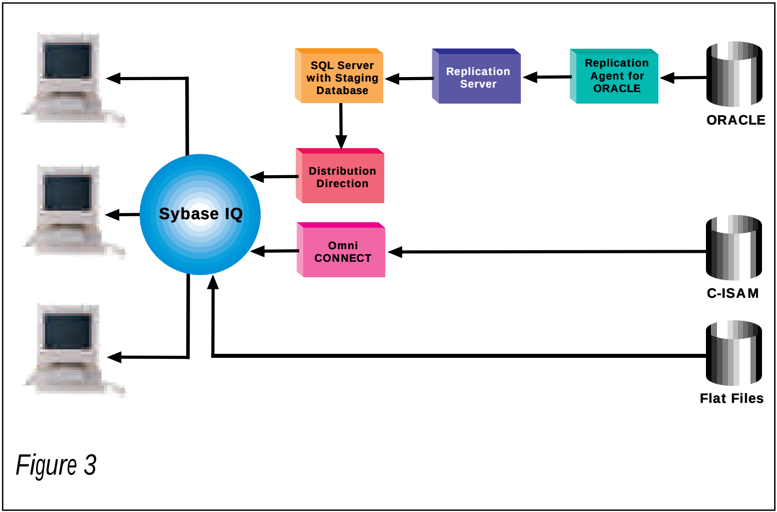


Figure 3

The New Generation of Data Warehouses

In spite of the many benefits of data warehousing, first-generation warehouses have exposed many challenges to effectively delivering the significant benefits that they promise. Some of these issues are present in all data warehouses, although many are aggravated as the data warehouse grows in size and complexity. These issues are manageability, ease-of-use, and enterprise scalability.

Manageability

Key among these issues are manageability and operational issues. New tools from companies have appeared in the past few years to deal with the difficulties of building a data warehouse and creating both technical and business metadata. But no tools have been available for managing the data warehouse once it is built.

Following are some key manageability issues:

- How do you tune the data warehouse for performance without impacting saved queries and decision support applications?
- How do you build the large summary tables necessary for data warehouse performance without impacting the availability of your warehouse?
- How do you effectively manage security in the warehouse environment?

Ease of Use

Other key issues in the data warehouse involve ease-of-use for end users. For example:

- How do you make warehouses easy to use for non-technical business users? Since most knowledgeable users cannot run ad hoc queries, they rely on prescanned reports. In this type of static environment, they do not take full advantage of the warehouse's potential because they cannot ask the questions that they should be asking.
- How do you provide ease of use while allowing the flexibility of ad hoc queries and avoid heavy IT resource expenditures in application development?
- How do users quickly and easily determine what information is available in one or more large data warehouses?

Enterprise Scalability

A final set of issues are directly related to scaling a data warehouse into a large warehouse. Some of these have to do directly with its size, while others relate to support for very different subject areas, data sources, and user groups. For example:

- How do you query across multiple subject areas? For instance, how do you relate both orders and shipments in the same query?
- How do you integrate multiple warehouses in decentralized companies?
- How do you physically manage extremely large data warehouses, ranging up to the terabytes in size?

Proactive Data Warehouses

Most first-generation data warehouses only responded to questions from analysts and other users. Next-generation systems will not wait. They will be proactive, asking questions themselves, all the time. When they find an answer that needs action, they will notify the appropriate company personnel. Two technologies that support detect and alert capabilities include: "database event alerters" which automatically monitor databases for changes that exceed critical thresholds; and integration of data warehouses with groupware and e-mail technologies for intelligent distribution of alerts.

In the next generation, management technologies will be as important to successful data warehousing as data transformation and end-user tools were to the first generation. Replication, especially heterogeneous replication, will ensure problem-free transfer of data and confidence that everyone is looking at the same current information.

Object-Oriented Data Warehouses

One of the clearest cases for object-oriented technologies is in the exploitation of data warehouses. Hundreds of millions of dollars are being invested in creating data warehouse infrastructures. Return on that investment will come only from visible business-critical applications, and those applications will come only from a cascading process of application creation, enhancement, and re-creation. Data warehouses are also being enhanced to support complex data such as video, audio, images, diagrams, and text. Support of those non-traditional datatypes in the same warehouse as traditional datatypes—along with the ability to quickly develop and enhance the infrastructure—is the perfect application of object technology.

Summary

The first wave of data warehousing has been an exciting time. It started with the simultaneous appearance of three enabling technologies: data transformation and meta data tools; symmetric multiprocessing (SMP) hardware and databases that could exploit it; and multidimensional end user tools. Thousands of companies have put these technologies to work and proved that better access to information is not only possible, but also extremely beneficial to their businesses. In proving their value, these pioneers established an enormous demand for data warehousing around the world. But that success has brought with it a new set of demands that some are ill-prepared to meet.

In designing and implementing data warehouse systems using Sybase IQ, we have to consider some critical factors to gain the most out of Sybase IQ capabilities. Careful design and consideration of these factors leads to fast, robust data warehouse systems. Sybase IQ's efficient indexing technology introduces new and different ways of addressing data warehousing problems while preserving all the Sybase technology in its core. ■