New Features Guide
Sybase® ETL 4.9

Document ID: DC00787-01-0490-01

Last revised: September 2009

This guide describes the new features in Sybase® ETL 4.9.

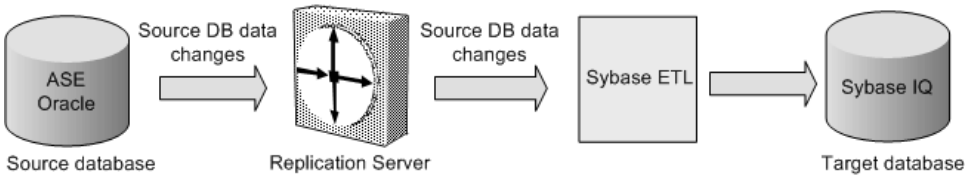
Topic	Page
Using ETL with Sybase Replication Server for enhanced incremental loading	1
ETL Scheduler support	3
Configuring alerts for runtime events	4
Transformation components enhancements	5
SQL Executor component added	5
Copy Splitter component added	6
Data Splitter JavaScript component enhanced	6
Transactionality enhancement	7
Support for connection sharing	9
DB Staging component enhancement	10
Data sink components enhancement	11
Executing multiple projects concurrently	11
Supported repository	11

Using ETL with Sybase Replication Server for enhanced incremental loading

You can now use ETL with Sybase Replication Server® to load data changes that Replication Server captures from Sybase Adaptive Server® Enterprise (ASE) and Oracle databases, into Sybase IQ. This solution supports different schemas between the source and target tables, as well as transformation of data by Sybase ETL.

Copyright 2009 by Sybase, Inc. All rights reserved. Sybase trademarks can be viewed at the Sybase trademarks page at http://www.sybase.com/detail?id=1011207. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. Unicode and the Unicode Logo are registered trademarks of Unicode, Inc. All other company and product names mentioned may be trademarks of the respective companies with which they are associated.

Figure 1: Incremental loading using Sybase Replication Server



Sybase ETL 4.9 includes a new component, CDC Provider Sybase Replication Server, which supports the Sybase Replication Server based incremental load solution, thus enhancing performance and usability, as well as supporting additional use cases.

In versions earlier than 4.9, incremental load was supported by the DB Data Provider Index Load component, which captured the modified data based on ascending index attribute values only.

In addition, you can now capture all data changes, including insert, update and delete activities in the source database, without depending on or modifying the source table schemas, or hampering the performance of the source database. To ensure consistency, you can enable the ETL project-level transactionality feature.

All data changes in the source database are captured by Sybase Replication Server. ETL receives the data changes from Replication Server, transforms it, and loads it into Sybase IQ.

Using the CDC Provider Sybase Replication Server component, you can automate the process of configuring Sybase Replication Server to create and drop replication definitions, replication connections, function strings, and replication subscriptions, or mark table in a source database to replicate, for all ETL tasks.

You can use the ETL incremental load solution to:

- Extract data changes from Oracle and Sybase Adaptive Server Enterprise databases.
- Load data to a target Sybase IQ database or a file.
- Transfer data changes from source tables to target tables that have different table schemas.
- Use different transformation rules on different sets of changed data received from Replication Server.
- Retrieve old values and then apply a transformation rule to compare between old and the changed data.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

ETL Scheduler support

Sybase ETL 4.9 has extended its scheduler functionality to all available platforms by providing its own ETL Scheduler. The ETL Scheduler allows you to create, list, edit, delete, execute, and terminate scheduled tasks using a selected engine from the ETL Development GUI.

Earlier versions of Sybase ETL Development used the Windows Task Scheduler directly. If you are upgrading from Sybase ETL Development 4.8 or earlier and want to migrate scheduled tasks, you must import existing tasks using the Runtime Manager.

Additionally, the ETL Scheduler can simultaneously run multiple instances of a task. For an already running scheduled task, you can specify any one of these options:

- Execute new task concurrently – starts new schedule execution along with the current process.
- Execute new task sequentially – waits to start next schedule execution until the completion of the current process.
- Do not execute new task – continues processing current task and ignores request to start a new schedule execution.
- Cancel the running task before executing new task – stops current process and starts the new schedule execution immediately.

ETL Scheduler follows a file-based configuration. All configurations are stored in a stand alone flat file with each grid engine. A file lock is used to ensure that only one ETL Schedule service is started per host machine. Sybase recommends that you do not manually modify the configuration file. Use the ETL Runtime Manager from the ETL Development GUI for any scheduler configuration.

See Chapter 4, “Advanced Concepts and Tools” in the *Sybase ETL 4.9 Users Guide*.

Configuring alerts for runtime events

In Sybase ETL 4.9, you can set alerts at the occurrence of runtime events such as, project or job start, complete, or error. You can configure ETL to send E-mail alerts when any of these events occur:

Note You can specify multiple email addresses to subscribe to an ETL event notification.

- Jobs – notifies when a job starts, finishes successfully, or finishes abruptly with errors.
- Job Start – occurs when a job is started, and contains the job name, the grid engine on which it is running, and the time it started.
- Job Finish – occurs when a job is finished normally, and contains the job name, the grid engine on which it was running, the start and end time of the job, and the list of all the projects.
- Job Error – occurs when a job finishes with errors and contains details like job name, grid engine name, time start, time failed, and the error message.
- Projects – occurs when a project starts, finishes successfully, or ends abruptly with errors.
- Project Start – occurs when a project starts and contains details like project name, grid engine on which it started, and the time it started.
- Project Finish – occurs when a project ends successfully and contains information such as project name, the grid engine name on which it was running, project start and end time, and the number of rows successfully loaded.
- Project Error – occurs when a project ends abruptly with errors. The mail has details like project name, the grid engine name on which it was running, project start and failed time, number of rows successfully loaded, and the error messages.

Note Multiple grid node processes running on the same grid engine share the same alert service configuration file and alert history log file.

Use the Alert Manager from the ETL Development GUI to create, edit, and delete alert definitions. You can create multiple alerts for any event type for project simulation and job and project execution. However, the alert name must be unique.

You can view event alert history in the File Log Inspector as well as on the ETL Web interface. You can also set filter conditions while mapping an alert. For example, you can define a filter to check for only a specific project name to trigger an alert.

Sybase ETL 4.9 also supports storing all the alert history in a separate log file for help during troubleshooting.

Sybase recommends you to set up email alert notifications on one machine. You can then manually copy alert configuration file across different grid engines running on the network.

See Chapter 4, "Advanced Concepts and Tools" in the *Sybase ETL 4.9 Users Guide*.

Transformation components enhancements

This section describes the enhancements made to the transformation components.

SQL Executor component added

ETL 4.9 includes a new transformation component, SQL Executor, which allows you to execute a custom SQL statement or multiple statements against a database server. SQL Executor is a standalone component without any input or output ports, and can be placed into a project that is separate from other components, or in a single project containing one or more SQL Executor components.

The SQL Executor component also allows you to load data from multiple input data streams into the Sybase IQ database in a single transaction, using multiple SQL statements. This leads to maximum flexibility for all ETL users.

Use the SQL Executor component to:

- Allow other projects to read or write data into the same database.
- Load data from a source table to an IQ supported text file format, using a SQL statement.
- Load data from the text file into the target IQ database, in a single transaction, using the Load Table command.

The SQL Executor component comes with pre- and post-SQL statement parameters. It executes SQL script line-by-line, then checks whether the SQL script is syntactically correct at the target database. You can also define different custom SQL scripts to execute, if the script runs successfully or fails.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

Copy Splitter component added

Sybase ETL 4.9 includes the new Copy Splitter component, which unconditionally copies input data to each output port. The Copy Splitter component lets you write data to each output port without evaluating output port conditions. By default, the component has two output ports.

Using the Copy Splitter component eliminates the need to invoke JavaScript, and reduces the cost of condition evaluation, therefore improving performance.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

Data Splitter JavaScript component enhanced

The Data Splitter JavaScript component has been enhanced to allow you to evaluate and define mutually exclusive output port conditions, thus delivering a significant performance improvement that not only eliminates memory allocation system calls, but also provides an efficient method of memory management.

In versions of Sybase ETL earlier than 4.9, the Data Splitter JavaScript component could only evaluate an input record against the conditions of every output port, causing potential bottlenecks and memory blockages. Furthermore, it did not offer the flexibility to use any other efficient logical constructs.

You can now use the Data Splitter JavaScript component to define mutually exclusive output port conditions, thus significantly reducing condition evaluations. Every input record matches with 0 or 1 output port expressions. The first output port with a matching condition receives the input record. No further port conditions are evaluated. If the expression does not match, it goes to the next port to evaluate the expression until a match occurs to commit the data in the matching port.

If data trends exist that favor some conditions over others, you can specify a custom evaluation sequence, listing the higher probability conditions at the beginning. This is especially useful when the number of output ports increases, as it helps to avoid evaluation of unnecessary conditions.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

Transactionality enhancement

Sybase ETL 4.9 extends data commit and rollback support at the end of the write operation across multiple components within a job and project execution or simulation. Data is committed at the end of the write operation for a successful execution or rolled back for an unsuccessful execution. In addition, the Multi-Project and Synchronizer job components now provide more control over the included projects by allowing you to commit intermediate work as soon as the processing finishes. If a project downstream the Multi-Project components fails, the projects that have already been committed remain unaffected; only downstream projects are rolled back.

In versions earlier than 4.9, you could not use transactions across multiple components or projects. Also, you could not roll back the complete set of changes in a target component. This is because, transactionality was supported only on component and batch levels.

Now, you can use transactionality across multiple projects, thus allowing you to execute or apply an all-or-nothing database update and load strategy. You can also set the job or project property to enable commit and rollback features for all those components that supports transactions. If you do not select the Propagate Rollback option, then the project or job does not enforce a transaction rollback on successful components, if one or more components fail. Each failed component rolls back its own transactions.

You can also select whether to commit or roll back any operation done by transactional components during project simulation. You can simulate either a success or failure condition on execution termination and reset the project to its initial state. This clears all port buffers, releases temporary tables, and closes database connections and temporary files. You can also disable new transactionality feature for all components within a job or project.

The improved transactionality feature lets you:

- Perform an incremental load of data from a source database to Sybase IQ. On success, new rows are inserted and modified rows are updated into Sybase IQ. On execution failure, all modifications to the target table and staging table in IQ are rolled back as part of a single transaction.
- Load multiple tables into Sybase IQ. On success, all tables in IQ are loaded with the new information from the data in the source database, and if execution fails, the modifications to all target tables are rolled back.
- Delete multiple rows from a target IQ database in a single transaction.
- Load IQ via Load Table as part of a single transaction.
- Rollback source components pre- and post-processing SQL data from the target table at the end of the project execution.
- Simulate a DBMS-style transaction when writing data to a local text file using the Text Data Sink component.
- Commit all the projects of a multiproject job within a single distributed transaction.
- Simulate a transactional project and commit or roll back at any step during the simulation process.
- Stop a transactional job and ensure the data written to transactional resources within the job rolls back.
- Create a job using the Migration wizard, defining transactionality for all projects and target DB components.

All database changes, including pre-SQL and post-SQL statements that are enclosed in a database transaction are rolled back along with all the other transactional components in the project, if project execution fails. You can also manually roll back any uncommitted tasks during project simulation.

However, you must execute the data definition language (DDL) SQL commands (create, drop, and alter) before executing any pre-SQL commands that you may need to rollback. This is because DDL SQL commands causes the database server to do an implicit commit transaction. For example, if you insert data into a table and execute CREATE TABLE command, the inserted data cannot be rolled back because it has already been implicitly committed by the database.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

Support for connection sharing

ETL 4.9 includes support for connection sharing for all these destination components:

- DB Bulk Load Sybase IQ
- DB Data Sink Insert
- DB Data Sink Update
- DB Data Sink Delete

This new functionality allows you to use multiple database components to access the same source or target table without any lock table failure error. It allows components to share a single connection to the database with other target components that have defined identical connection and database parameters. In versions earlier than 4.9, project execution failed when two or more components tried to access the same table at the same time. Now, with the new connection sharing functionality you can extract data from multiple source tables and upload it into a single target table simultaneously. It also allows you to perform bulk-loading into a Sybase IQ database, leading to better performance.

The connection sharing functionality is supported only between components in the same project. Components that exist in different projects but inside the same job cannot share a connection. Also, you cannot use connection sharing functionality with the existing ETL multiple writer functionality.

Note Components that define the same database interface and login information, but have differing database options cannot share a connection, and generate an error when the project is executed or simulated.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

DB Staging component enhancement

With Sybase IQ as the staging platform, Sybase ETL 4.9 lets you bulk-load data into staging tables using the Load Table statement, thus, leading to faster performance. In earlier versions, the DB Staging component could use the INSERT statement to perform only row-based operations for loading the incoming data into the target database.

The DB Staging component has been enhanced to:

- Create tables automatically based on the input port structures at runtime. You can remove a table created at runtime once the project finishes processing.
- Execute additional SQL scripts to modify or update the staging tables before output is created. You can now execute SQL transformation scripts once all the data from the transformation flow has been loaded into the tables associated with the input ports, and before retrieving the query result set.
- Ensure data is committed at the end of the write operation for a successful execution, and rolled back for an unsuccessful execution.

You need not specify the Load Stage Path when client-side loading is enabled for your Sybase IQ 15.0 staging database with ODBC as the interface. The IQ server automatically uses its default LOAD TABLE statement to add records from files located on the remote host machines into the Sybase IQ table.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

Data sink components enhancement

The DB Data Sink Insert, DB Data Sink Update, and DB Data Sink Delete components, have been enhanced to bulk-load data into target tables of Sybase IQ database using the Load Table statement.

See Chapter 5, “Components” in the *Sybase ETL 4.9 Users Guide*.

Executing multiple projects concurrently

Sybase ETL 4.9 supports parallel project execution to improve performance especially when you use a single ETL server to schedule multiple projects. In earlier versions, you could only run one project per job per grid engine, which caused under utilization of available resources. Now, you can execute multiple ETL projects concurrently on a single or multiple grid engines. Multiple ETL users can also execute projects concurrently on the same remote grid engine.

You must edit the *Default.ini* file in the *etc* subdirectory of the installation folder, to specify the maximum number of projects you want to execute at one time. By default, the maximum number of projects that can be executed at a time is 10. However, if you set the maximum project value (`MAXPROJECTS`) to 0 or a negative number, there is no limit to the number of projects that can be concurrently executed on the grid engine.

See Chapter 6, "Sybase ETL Server" in the *Sybase ETL 4.9 Users Guide*.

Supported repository

Sybase ETL 4.9 supports only SQL Anywhere® 11 as its repository database. If you are using any other repository from an earlier version of ETL, you must migrate your existing ETL repository to SQL Anywhere 11.

See "Migrating from an existing repository to SQL Anywhere" in Chapter 4, "Upgrading" in the *Sybase ETL Installation Guide*.