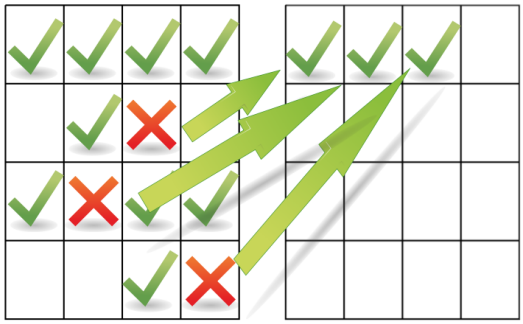
DB2 11 for z/OS new way of archiving data

With DB2 11 for z/OS user can create archive tables to manage historical data for existing tables that is not often referenced. An 'archive table' - a new table type - stores deleted rows from another table. That base table is called an 'archive-enabled table'.

Archive tables provide the following benefits:

- For Archive enabled tables: No application change is required. No DBA intervention to recall data is required.
- DB2 can manage historical data automatically. Users do not have to move data to a separate table.
- Because rows that are infrequently accessed are stored in a separate table, performance of queries against the archive-enabled tables can potentially be improved.
- SQL queries can include or exclude archive table data without having to change the SQL statement and prepare the application again. Instead, new global variables control the scope of a query.
- Data in archive tables can be stored on a lower-cost device to reduce operating costs.

By specifying the new ENABLE ARCHIVE clause on the ALTER TABLE statement, the new version of DB2 allows automatically to insert rows that are deleted from one table into a separate table that is called an 'archive table'. A global DB2 variable (SYSIBMADM.MOVE_TO_ARCHIVE) controls whether DB2 inserts rows that are deleted from this table into the associated archive table. Once this global variable is turned on for archiving, an update the archive-enabled table is not possible. To update rows, set SYSIBMADM.MOVE_TO_ARCHIVE to 'N' first.



If you want to query an archive-enabled table, you have to indicate whether the query considers the rows in the archive table. You also use a global variable SYSIBMADM.GET_ARCHIVE='Y' to indicate this preference. Therefore, you can easily change the query to include or exclude archive table data without having to update the SQL.

Implementing base and archive-enabled tables

Activates archiving:

```
ALTER TABLE base-table-name
      ENABLE ARCHIVE USE archive-table-name;
```

Following criteria must be satisfied to successfully alter the base table:

- The table must not already be defined as an archive-enabled table or an archive table.
- The table must not contain a period.
- The table must be the only table in the table space.
- The table must not have a column mask or row permission defined.
- The table must not be one of the following tables:

  - A materialized query table
  - An incomplete table
  - An auxiliary table
  - A table that is involved in a clone relationship
  - A table that was implicitly created for an XML column
  - A table that contains a security label column
  - A system-period temporal table
  - A history table
- ENABLE ARCHIVE must not be specified with other clauses on the ALTER TABLE statement.
- The privilege set must include the privileges to issue an ALTER TABLE statement for the associated archive table.

The *archive-table-name* must identify a table that exists at the current server and must satisfy the following criteria:

- The table must be the only table in the table space.
- The table must not have an incomplete table definition.
- The table must not be defined as the parent or child in an existing referential constraint.
- The table must not include a period.
- The table must not include a row permission or column mask.
  - The table cannot be one of the following tables: Catalog table, archive-enabled table, an existing archive table, system-period temporal table, history table, declared global temporary table or created global temporary table, materialized query table, a view, auxiliary table, table that was implicitly created for an XML column, clone table, table that has a clone defined on it.
  - The table must not contain any of the following columns: Identity column, row-begin column, row-end column, transaction-start-ID column, security label column.
- The privilege set must include the privileges to issue an ALTER TABLE statement for the associated archive table.
- The archive-enabled table and the associated archive table must have the same number and order of columns. The following attributes for the corresponding columns of the two tables must be the same: Name, data type, length (excluding inline LOB length or XML length in the base table) with precision, and scale, FOR BIT and SBCS or MIXED DATA attribute for character string columns, NULL attribute, HIDDEN attribute, CCSID, field procedure.
- If a column of an archive-enabled table is defined as ROWID, the corresponding column of the archive table must also be defined as ROWID with the GENERATED ALWAYS attribute.
- If a column of an archive-enabled table is defined as row change timestamp, the corresponding column of the archive table must also be defined as row change timestamp with the GENERATED ALWAYS attribute.

If ENABLE ARCHIVE, or DISABLE ARCHIVE is specified, no other clause is allowed on the ALTER TABLE statement, like ADD CLONE, DROP CLONE, RENAME COLUMN, ADD ORGANIZE BY HASH, ALTER ORGANIZATION, DROP ORGANIZATION, ADD VERSIONING, DROP VERSIONING, DROP COLUMN, ACTIVATE, DEACTIVATE.

Removing archive-enablement

The new DISABLE ARCHIVE clause on the ALTER TABLE statement can be used to remove the relationship between the archive-enabled table and the associated archive table. After the ALTER statement is successfully processed, both tables are considered plain tables.

Deactivates archiving:

```
ALTER TABLE base-table-name
      DISABLE ARCHIVE;
```

Deployment of automatic archiving

Two new updatable global variables are introduced to give the user control over whether archived rows for rows deleted from an archive-enabled table are automatically written to an associated archive table, and whether rows in the archive table should be included when an archive-enabled table is referenced in a table-reference.

- If SYSIBMADM.GET_ARCHIVE = 'Y'
  data is retrieved from the archive table when an archive-enabled table is referenced in a table-reference. The access of historical data in the archive table is 'transparent' to the application - DB2 rewrites the query with UNION ALL operator. All subsequent SQL statements including those from invoked function, stored procedure, and trigger. This allows the application to see both active and archive data without modifying the SQL statements in multiple packages.

- If SYSIBMADM.MOVE_TO_ARCHIVE='Y'
  historical data is stored in the associated archive table when a row is deleted in an archive-enabled table. The storing of a row of historical data in the archive table is 'transparent' to the application. UPDATE statements will fail.

The following BIND/routine options are added to control the sensitivity to the settings of the SYSIBMADM.GET_ARCHIVE global variable:

- ARCHIVESENSITIVE (default YES)
  - BIND PACKAGE
  - REBIND PACKAGE
  - REBIND TRIGGER PACKAGE
  - CREATE TRIGGER (implicit trigger package)
- ARCHIVE SENSITIVE (default YES)
  - CREATE FUNCTION (SQL scalar)
  - ALTER FUNCTION (SQL scalar)
  - CREATE PROCEDURE (SQL native)
  - ALTER PROCEDURE (SQL native)
- ARCHIVE SENSITIVE (DB2I panels)
  - Install panel DSNEBP10
  - Install panel DSNEBP11
  - Install panel DSNEBP19

Restrictions

- ALTER TABLE of an archive-enabled table with the ADD COLUMN clause also implicitly adds the new column to the associated archive table.
- You cannot ALTER TABLE DROP COLUMN if table is an archive-enabled table or an archive table.
- Any reference to an archive-enabled table for existing values in an INSERT, UPDATE, DELETE, MERGE will not include rows of the associated archive table.
- CREATE TRIGGER and REBIND TRIGGER PACKAGE will fail if the trigger has archive enabled table reference in the WHEN clause and the trigger package is generated with ARCHIVESENSITIVE YES.
- Restrictions on INSERT and UPDATE statements:
  - You cannot insert rows into an archive-enabled table if the value of the SYSIBMADM.MOVE_TO_ARCHIVE global variable is Y. Otherwise, if this global variable is not set to Y, you can specify an archive-enabled table as the target of the INSERT statement. In this case, the content of the associated archive table is not affected.
  - A reference to an archive-enabled table as the target of the UPDATE statement does not affect rows in the associated archive table. For archive-enables tables to update, following conditions must met:
    - The SYSIBMADM.MOVE_TO_ARCHIVE global variable is set to Y.
    - The SYSIBMADM.GET_ARCHIVE global variable is set to Y, the
    - ARCHIVESENSITIVE bind option is set to YES
    - and the operation is a positioned update
- Restrictions on DROP statements:
  - You can drop a database that contains an archive table only if the database also contains the associated archive-enabled table. You can drop a database that contains an archive-enabled table when the associated archive table is contained in another database. In this case, the action cascades to drop the archive table in the other database.
  - If an archive-enabled table is dropped, the archive table and any indexes that are defined on the archive table are also dropped. To drop an archive-enabled table, the privilege set must also contain the authorization that is required to drop the archive table. An archive table cannot be explicitly dropped by using the DROP statement.
  - The table space cannot be dropped if it contains a history table or an archive table.

Effects of downwards compatibility and subsystem defaults

New options offered in V11 can be used in dynamic SQL statements only when this special register value has the V11R1 value (or implicitly inherited it from the DSNZPARM default). For example, the SYSTIMESENSITIVE, BUSTIMESENSITIVE, and ARCHIVESENSITIVE options cannot be explicitly specified with the value of YES in new-function mode if this special register is set to V10R1.

Users can change the value of this special register by executing the SET CURRENT APPLICATION COMPATIBILITY in dynamic SQL statements:

```
SET CURRENT APPLICATION COMPATIBILITY = 'V11R1';
SET CURRENT APPLICATION COMPATIBILITY = 'V10R1';
```

This special register is applicable only to dynamic SQL and specifies the DB2 release level that the dynamic SQL is compatible with. The data type is VARCHAR(10). A routine environment cannot inherit this special register value from the caller's environment, even if the routine was created with the INHERIT SPECIAL REGISTER option.

The initial value of CURRENT APPLICATION COMPATIBILITY is determined by the value of the APPLCOMPAT bind parameter for the package.

The system parameter APPLCOMPAT is the default value when binding packages without explicitly specifying the APPLCOMPAT bind option. Ensure that all applications are ready for new compatibility behaviour before you change the default value. Before Version 11 new-function mode, the APPLCOMPAT subsystem parameter must be set to V10R1. This setting ensures that existing SQL applications are bound for compatibility with the previous release by default. After DB2 is in new-function mode, users can consider setting the APPLCOMPAT subsystem parameter to V11R.

Sources

IBM® DB2® 11 for z/OS® literature, What's New, Administration Guide, SQL Reference, Introduction