Working with Temporary and Transient Tables

In addition to permanent tables, which is the default table type when creating tables, Snowflake supports defining tables as either temporary or transient. These types of tables are especially useful for storing data that does not need to be maintained for extended periods of time (i.e. transitory data).

Temporary Tables

Snowflake supports creating temporary tables for storing non-permanent, transitory data (e.g. ETL data, session-specific data). Temporary tables only exist within the session in which they were created and persist only for the remainder of the session. As such, they are not visible to other users or sessions. Once the session ends, data stored in the table is purged completely from the system and, therefore, is not recoverable, either by the user who created the table or Snowflake.

Note
In addition to tables, Snowflake supports creating certain other database objects as temporary (e.g. stages). These objects follow the same semantics (i.e. they are session-based, persisting only for the remainder of the session).

Data Storage Usage for Temporary Tables

For the duration of the existence of a temporary table, the data stored in the table contributes to the overall storage charges that Snowflake bills your account. To prevent any unexpected storage changes, particularly if you create large temporary tables in sessions that you maintain for periods longer than 24 hours, Snowflake recommends explicitly dropping these tables once they are no longer needed. You can also explicitly exit the session in which the table was created to ensure no additional charges are accrued.

For more information, see Comparison of Table Types (in this topic).

Potential Naming Conflicts with Other Table Types

Similar to the other table types (transient and permanent), temporary tables belong to a specified database and schema; however, because they are session-based, they aren't bound by the same uniqueness requirements. This means you can create temporary and non-temporary tables with the same name within the same schema.

However, note that the temporary table takes precedence in the session over any other table with the same name in the same schema. This can lead to potential conflicts and unexpected

behavior, particularly when performing DDL on both temporary and non-temporary tables. For example:

- You can create a temporary table that has the same name as an existing table in the same schema, effectively hiding the existing table.
- You can create a table that has the same name as an existing temporary table in the same schema; however, the newly-created table is hidden by the temporary table.

Subsequently, all queries and other operations performed in the session on the table affect only the temporary table.

Important
This behavior is particularly important to note when dropping a table in a session and then using Time Travel to restore the table. It is also important to note this behavior when using CREATE OR REPLACE to create a table because this essentially drops a table (if it exists) and creates a new table with the specified definition.

Creating a Temporary Table

To create a temporary table, simply specify the TEMPORARY keyword (or TEMP abbreviation) in CREATE TABLE.

Note that creating a temporary table does ***not*** require the CREATE TABLE privilege on the schema in which the object is created.

For example:

```
CREATE TEMPORARY TABLE mytemptable (id NUMBER, creation_date DATE);
```

Note
After creation, temporary tables cannot be converted to any other table type.

Transient Tables

Snowflake supports creating transient tables that persist until explicitly dropped and are available to all users with the appropriate privileges. Transient tables are similar to permanent tables with the key difference that they do not have a Fail-safe period. As a result, transient tables are specifically designed for transitory data that needs to be maintained beyond each session (in contrast to temporary tables), but does not need the same level of data protection and recovery provided by permanent tables.

Data Storage Usage for Transient Tables

Similar to permanent tables, transient tables contribute to the overall storage charges that Snowflake bills your account; however, because transient tables do not utilize Fail-safe, there are no Fail-safe costs (i.e. the costs associated with maintaining the data required for Fail-safe disaster recovery).

For more information, see Comparison of Table Types (in this topic).

Transient Tables Created as Clones of Permanent Tables

When you create a transient table as a clone of a permanent table, Snowflake creates a zero-copy clone. This means when the transient table is created, it utilizes no data storage because it shares all of the existing micro-partitions of the original permanent table. When rows are added, deleted, or updated in the clone, it results in new micro-partitions that belong exclusively to the clone (in this case, the transient table).

When a permanent table is deleted, it enters Fail-safe for a 7-day period. Fail-safe bytes incur storage costs. If a transient table is created as a clone of a permanent table, this might delay the time between when the permanent table is deleted and when all of its bytes enter Fail-safe. If the transient table clone shares any micro-partitions with the permanent table when it is deleted, those shared bytes will only enter Fail-safe when the transient table is deleted.

Transient Databases and Schemas

Snowflake also supports creating transient databases and schemas. All tables created in a transient schema, as well as all schemas created in a transient database, are transient by definition.

Creating a Transient Table, Schema, or Database

To create a transient table, schema, database, simply specify the TRANSIENT keyword when creating the object:

- CREATE TABLE
- CREATE SCHEMA
- CREATE DATABASE

For example, to create a transient table:

```
CREATE TRANSIENT TABLE mytranstable (id NUMBER, creation_date DATE);
```

Note
After creation, transient tables cannot be converted to any other table type.

Comparison of Table Types

The following table summarizes the differences between the three table types, particularly with regard to their impact on Time Travel and Fail-safe:

Type	Persistence	Cloning (source type => target type)	Time Travel Retention Period (Days)	Fail-safe Period (Days)
Temporary	Remainder of session	Temporary => Temporary Temporary => Transient	0 or 1 (default is 1)	0
Transient	Until explicitly dropped	Transient => Temporary Transient => Transient	0 or 1 (default is 1)	0
Permanent (Standard Edition)	Until explicitly dropped	Permanent => Temporary Permanent => Transient Permanent => Permanent	0 or 1 (default is 1)	7
Permanent (Enterprise Edition and higher)	Until explicitly dropped	Permanent => Temporary Permanent => Transient Permanent => Permanent	0 to 90 (default is configurable)	7

Time Travel Notes

- The Time Travel retention period for a table can be specified when the table is created or any time afterwards. Within the retention period, all Time Travel operations can be performed on data in the table (e.g. queries) and the table itself (e.g. cloning and restoration).
- If the Time Travel retention period for a permanent table is set to 0, it will immediately enter the Fail-safe period when it is dropped.
- Temporary tables can have a Time Travel retention period of 1 day; however, a temporary table is purged once the session (in which the table was created) ends so the

actual retention period is for 24 hours ***or*** the remainder of the session, whichever is shorter.
- A long-running Time Travel query will delay the purging of temporary and transient tables until the query completes.

Fail-safe Notes

- The Fail-safe period is not configurable for any table type.
- Transient and temporary tables have ***no*** Fail-safe period. As a result, no additional data storage charges are incurred beyond the Time Travel retention period.

Important
Because transient tables do not have a Fail-safe period, they provide a good option for managing the cost of very large tables used to store transitory data; however, the data in these tables cannot be recovered after the Time Travel retention period passes.

For example, if a system failure occurs in which a transient table is dropped or lost, after 1 day, the data is not recoverable by you or Snowflake. As such, we recommend using transient tables ***only*** for data that does not need to be protected against failures or data that can be reconstructed outside of Snowflake.

For more information, see Data Storage Considerations.