CopyStorm/Restore Tracker Database

Version: 19-May-2016

CopyStorm/Restore uses a local Tracker Database to keep track of the progress of a restore to Salesforce. This document describes the internal structure and types of data stored in a CopyStorm/Restore Tracker Database.

Table of Contents

Executive Summary

CopyStorm/Restore uses a local Tracker Database to keep track of the progress of a restore to Salesforce. This document describes the internal structure and types of data stored in a CopyStorm/Tracker Database.

The intended audience are people who need to understand exactly what records were restored by CopyStorm/Restore.

A Few Basic Facts

There a few basic facts about the CopyStorm/Restore tracker database that will help in understanding.

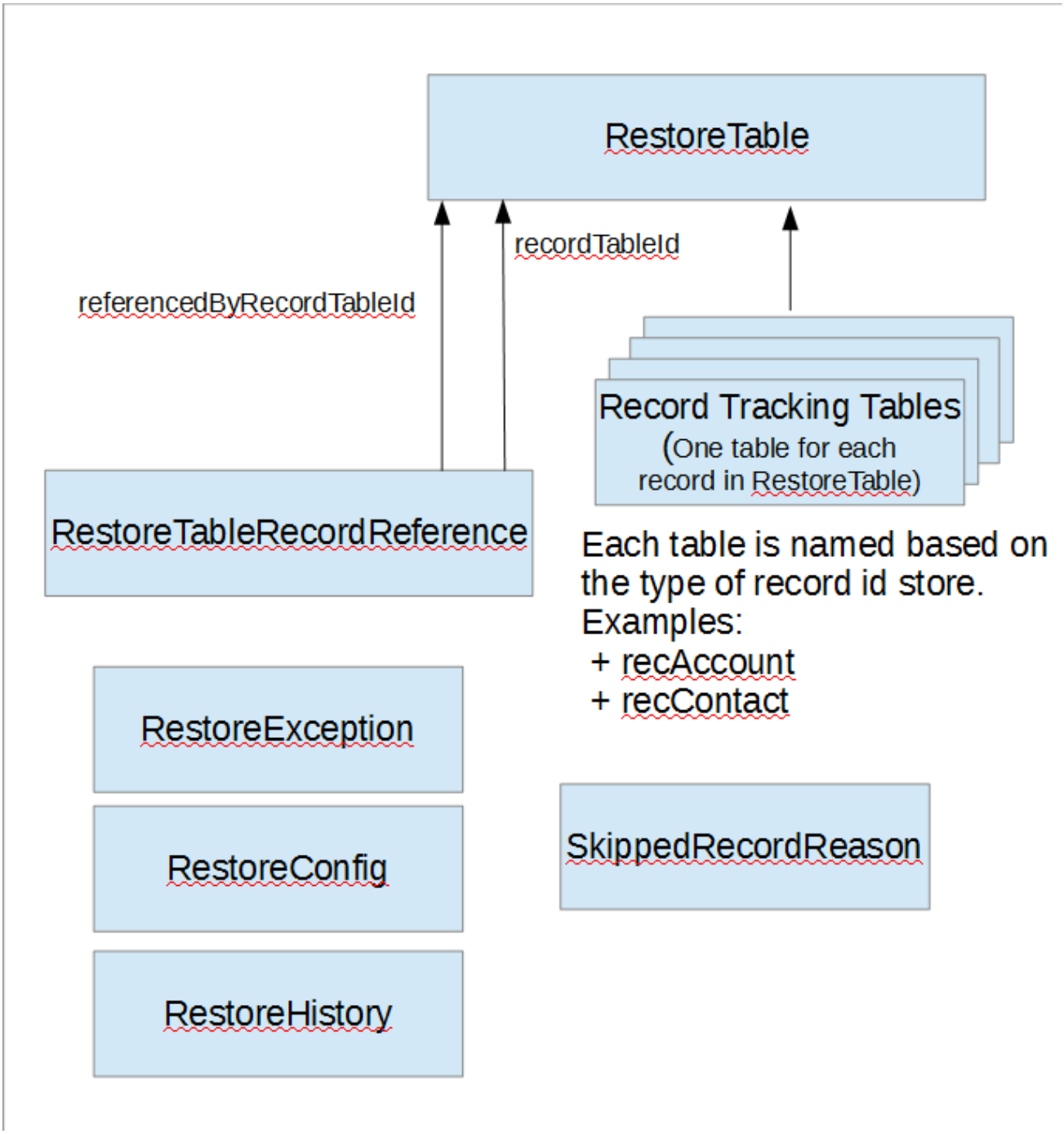
- The name of a tracker database for a restore is found on the *Global Parameters* tab in the *Tracker Database Name* field.
- Tracker databases are stored in the folder $HOME/.capstorm/copyStormRestoreSets.
- You can change the name of the tracker database to any name built from letters and digits.
- Tracker databases are implemented as H2 relational databases. You can learn about H2 http://www.h2database.com/html/main.html. H2 is a small footprint, portable, open source, lightweight database.
- CopyStorm/Restore creates its tracker database automatically on first use.
- The simplest way to view and export data from a tracker database is the Tracker Database tab in the CopyStorm/Restore GUI.
- A browser based tool for opening and viewing H2 databases can be started with the following command line:
    - java -jar [CopyStormRestoreRoot]/lib/h2-1.2.127.jar

Tracker Database ERD

Each tracker database has five key tables and one table for each Salesforce table in the restore set.

Table Name	Description
RestoreConfig	The *RestoreConfig* table contains simple <name,value> pairs that CopyStorm/Restore uses to determine the current version of the tracker database. This table is of no interest to anyone but CopyStorm/Restore developers.
RestoreTable	The *RestoreTable* table lists all Salesforce tables in the CopyStorm restore set.
Record Tracking Tables	Each record in the *RestoreTable* will also have an associated *Record Tracking Table*. Each Record Tracking Table lists the unique id of each Salesforce record in the restore set and the unique id was assigned to it in Salesforce once restored. In addition these tables contains various status flags that track the progress of the restore. Each Record Tracking Table is named based on the name of Salesforce table it is tracking. Examples: • recAccount • recContact • recMyCustomObject__c
RestoreTableRecordReference	The *RestoreTableRecordReference* table lists references from a field in one Salesforce table to another Salesforce table. It is used to enable full graph traversal when restoring related lists and self-referencing table fields.
SkippedRecordReason	The *SkippedRecordReason* table list the reason why records were skipped. The data is the same as what is display when the *Skipped Reasons* button is click in the CopyStorm/Restore GUI.
RestoreException	The *RestoreException* table contains each exception encountered when a attempting a restore.
RestoreHistory	The *RestoreHistory* contains records related to restore operations that have occurred using the current tracking database. The primary purpose of this table is to help Capstorm support.

The relationships between the various tables is simple.



The complexity of the *Record Tracking Tables* requires an explanation. Early versions of CopyStorm/Restore used a single table to track the Salesforce ids for all records to be restored. In practice this approach experience performance issue when many 1,000,000s of records were in the restore set and contributed by dozens of tables. Putting the pending ids for each Salesforce table into its own tracking database table overcame the performance issue and boosted some key operations (like finding candidate records to restore) by up to 20x for large data sets.

The usage of the tables is equally simple.

- When a table is referenced in the restore set, an entry is made in *RestoreTable*.
- When a record is identified as a candidate for a restore, an entry is made in a table named *recSalesforceTableName* (example: recAccount).
- When a record is a reference to a another table that is being restored, an entry is made in *RestoreTableRecordReference*.

During a restore:

- The restore status of a record is recorded in the appropriate *recSalesforceTableName*.
- The overall restore status of each table is recorded in *RestoreTable*.
- Any exceptions are recorded in *RestoreException*.
- Any skipped records are record in *SkippedReason*.
- High level events are recorded in *RestoreHistory.*

The next few sections describe the columns in each tracking database table.

RestoreConfig

Column	Data Type	Description
name	string	Name of an internal CopyStorm/Restore configuration variable. Often the only variable defined is *Version*. It is unlikely that anyone but Capstorm programmers will use this table.
value	string	Value of a configuration variable.

RestoreTable

Column	Data Type	Description
id	integer	Unique id of a table. This value is the primary key and referenced in other tables.
tableName	string	Name of a Salesforce table that is part of the restore set.
candidateScanComplete	boolean	Flag set to true when all possible restore candidates for the table have been discovered.
recordTrackingTable	string	Name of the H2 table used to track the progress of the restore. If the *tableName* is *Account* then the *recordTrackingTable* will be *recAccount*.

Record Tracking Table (one for each RestoreTable record)

Column	Data Type	Description
originalId	string	Original id of a Salesforce record in the CopyStorm database.
restoredId	string	The unique id of that record as restored in Salesforce. If the restore to Salesforce was a record update, then this value will be the same as *originalId*. If the restore required an insert, then this is the unique id in the target Salesforce.
wasInsert	boolean	Flag set to true if the restore operation on this record was an insert into the target Salesforce.
wasSkipped	boolean	Flag set to true if the restore operations was forced to skip this record.
isPending	boolean	Flag set to true if a restore attempt is pending for this record.
referenceIdScanPending	boolean	Flag set to true if the record needs to be scanned for references to other records – either in the same or other tables.
candidateScanPending	boolean	Flag set to true when the record is still being used to scan for addition restore candidates.

RestoreTableRecordReference

Column	Data Type	Description
recordId	string	Unique Salesforce id in the CopyStorm backup that is referenced by another record.
recordTableId	integer	*RestoreTable.id* of the table containing the *recordId*.
referencedByRecordId	String	Unique Salesforce id in the CopyStorm backup that references the corresponding *recordId* field in this row.
referencedByRecordTableId	integer	*RestoreTable.id* of the table containing the *referencedByRecordId* field.

SkippedRecordReason

Column	Data Type	Description
id	integer	Unique id of a skipped record reason.
tableName	string	Name of the Salesforce table owning the skipped record.
originalId	string	Salesforce id of the record in the CopyStorm backup that was skipped.

Column	Data Type	Description
reason	string	The reason why CopyStorm/Restore skipped the record.
createdDate	timestamp	The date on which the skipped record was created.

RestoreException

Column	Data Type	Description
id	integer	Unique id of an exception.
message	string	The top level message the exception. This is a one line description of the exception.
traceback	string	The full stack trace that caused the exception.
createdDate	timestamp	The date on which the exception record was created.

RestoreHistory

Column	Data Type	Description
id	integer	Unique id of a history event.
event	string	A one line description of the history event.
eventData	string	A longer description of the history event
createdDate	timestamp	The date on which the history record was created.

How to Use Tracker Database Tables

Most users will never look at a tracker database except via the Tracker Database tool in CopyStorm/Restore. What is important is that CopyStorm/Restore uses the tracker database for the following functions:

- A tracker database makes it possible for a restore to be interrupted and restarted.
- A tracker database allows addition items to be added to a restore set after a restore has completed.

The easiest was to examine a tracker database is to use the Tracker Database tool built into CopyStorm/Restore. However, if you want direct access to the database, the easiest wsy is via the web browser tool provided by H2. If you are more ambitious, you can use the H2 JDBC driver to view/manipulate a tracker database using more powerful tools like Squirrel.

Links you may need include:

- H2 - http://www.h2database.com/html/main.html
- Squirrel - http://squirrel-sql.sourceforge.net/