Data Replication Using Change Event Streams or Replication API

Knowledge Article Number 000332219

Description

With data replication, you can store and maintain an external, separate copy of your organization’s pertinent Salesforce data for specialized uses, such as data warehousing, data mining, custom reporting, analytics, and integration with other applications. Data replication provides you with local control and the ability to run large or ad hoc analytical queries across the entire data set without transmitting all that data across the network.

To keep your external data store up-to-date with Salesforce, use Change Data Capture to get real-time notifications of Salesforce record changes. By subscribing to a Change Data Capture channel, you receive a stream of change event messages for record changes, including insertion, update, deletion, and un deletion. With Change Data Capture, you get broad access to data and can perform updates in your target store using transaction boundaries. Change Data Capture provides a versioned event schema and retains change events temporarily for later retrieval. For more information, see the **Change Data Capture Basics Trailhead module** , or for a complete reference, see the **Change Data Capture Developer Guide**.

Alternatively, use the **Replication API** for data replication. The Replication API is not real-time and requires that you make repeated API calls to poll for changes over time. It includes two calls, getUpdated and getDeleted. The getUpdated call returns the IDs of newly created and updated records. The getDeleted call returns the IDs of records that have been deleted.

Resolution

You can use this API to accurately replicate data in your remote system without worrying about determining specific queries. Once you have a list of ID values, you can use retrieve to quickly export the data in batches of up to 2000 records.

The replication API has a specific advantage that it is designed to make sure you can't miss records and won't retrieve duplicates, both of which are possible when directly querying based on CreatedDate or LastModifiedDate fields (because of in-flight transactions that won't appear in queries until they are fully committed).