Amazon DynamoDB Connections

Contents:

Feature Availability: This feature is available in the following editions:

- Trifacta® Enterprise Edition
- Trifacta Professional Edition
- Trifacta Premium

Amazon DynamoDB is an easy-to-use online platform for creating and sharing non-relational document based databases. For more information, see *https://aws.amazon.com/dynamodb/*.

Early Preview connection: This connection is in early preview. It is read-only and available only in SaaS product editions. For more information on early previews, see *Early Preview Connection Types*.

Limitations and Requirements

NOTE: During normal selection or import of an entire table, you may encounter an error indicating a problem with a specific column. Since some tables require filtering based on a particular column, data from them can only be ingested using custom SQL statements. In this case, the problematic column can be used as a filter in the WHERE clause of a custom SQL statement to ingest the table.

- For more information, please consult the CData driver documentation for the specific table.
- For more information on using custom SQL, see *Create Dataset with SQL*.

- Supports Access key ID and Secret access key.

Create Connection

via Trifacta application

When you create the connection, please review the following properties and specify them accordingly:

Connection Property	Description
Domain	Your AWS domain name. The default domain value is amazonaws.com.

AWS Region	The hosting region for your Amazon Web Services resources. Default is set to Oregon .
Connect String Options	The following default value sets the connection timeout in seconds: ``` Timeout=0; ``` Setting this value to 0 disables timeouts.
Access key ID	Your AWS account access key. This value is accessible from your AWS security credentials page.
Secret access key	Your AWS account secret key. This value is accessible from your AWS security credentials page.
Default Column Data Type Inference	Leave this value as Enabled.

For more information, see the driver documentation *http://cdn.cdata.com/help/DDF/jdbc/default.htm*.

via API

Depending on your product edition, you can create connections of this type. Key information:

```
"vendor": "amazon_dynamodb",
"vendorName": "Amazon DynamoDB",
"type": "jdbc"
```

For more information, see

Trifacta API Reference docs: *Enterprise | Professional | Premium*

Data Type Conversions

For more information, see the driver documentation *http://cdn.cdata.com/help/DDF/jdbc/default.htm*.

Flattening Documents

Documents can contain other documents, which enables the storage of nested data. You can control the flattening of nested objects and arrays through the CData driver through Connect String Options.

> **NOTE:** Columns that have been flattened can be accessed or referenced using custom SQL queries. Additional information is below.

Flatten Objects:

By default, the CData driver flattens nested Objects. As needed, you can set FlattenObjects to `false` to disable this behavior.

In Amazon DynamoDB, objects are represented as Map datatype.

For more information, see *http://cdn.cdata.com/help/DDF/jdbc/RSBDynamoDB_p_FlattenObjects.htm*

Flatten Arrays:

By default, CData driver does not flatten Arrays.

As needed, you can configure the number of elements that you want to have returned in your flattened arrays.

In Amazon DynamoDB, Arrays are represented as NumberSet, StringSet ,or BinarySet datatypes.

> **NOTE**: List datatype does not support flattening property.

For more information, see *http://cdn.cdata.com/help/DDF/jdbc/RSBDynamoDB_p_FlattenArrays.htm*.

Referencing flattened columns:

If you have flattened Objects or Arrays, you can reference these columns using square brackets in your custom SQL queries.

Example of flattened Object:

```
SELECT [address.city] FROM my_table;
```

Example of flattened Array:

```
SELECT * FROM my_table WHERE [hobbies.0]='cricket';
```