Using Bulk Export to Amazon S3 Bucket

Treasure Data's bulk-export feature enables you to dump data into your Amazon S3 bucket.

Prerequisites

- Basic knowledge of Treasure Data, including TD Toolbelt.
- Amazon AWS account and Amazon S3 bucket.
    - This feature requires Amazon S3 Permissions for Object Operations
        - s3:PutObject
        - s3:GetBucketLocation

Limitations

- Data from one region cannot be exported to a different region.
- The Bulk Export command no longer supports the partitioning of exported data. This is to optimize the speed of export, which was too slow to meet requirements.

If you do require partitioning, we recommend using this command to export 1-hour segments at a time – automating the process with a script.

- Exporting float type columns is not supported. If you try to run a table export job with float type columns in the schema, you might see the error message:
  `invalid schema: unexpected type: float" A workaround is to manually change the schema of the table to double`
- Bulk export capability is limited to the following regions:

Users of	Code	Region Name
US Region	us-east-1	US East (N. Virginia) S3 bucket
Tokyo	ap-northeast-1	Asia Pacific (Tokyo) region S3 bucket

Exporting Your Data to an Amazon S3 Bucket

We highly recommend that you use jsonl.gz or tsv.gz format, for specific performance optimizations.

The dump is performed through MapReduce jobs. The location of the bucket is expressed as an S3 path with the AWS public and private access keys embedded in it.

The `td table:export` command dumps all the data uploaded to Treasure Data into your Amazon S3 bucket.

1. From a machine where your TD Toolbelt is installed, open a command line terminal.
2. Optionally, use the following syntax to validate the latest usage information for the td table:export command.

```
td table:export -help
```

3. Use the bulk export command to start the bulk export. Specify the database and table from which to dump your data.

```
td table:export <db> <table>
```

4. Optionally, enter values for the options that you want to use. For example, options are:

Option	Description
-w, --wait	wait until the job is completed

-f, --from TIME	export data that is newer than or same with the TIME
-t, --to TIME	export data which is older than the TIME
-b, --s3-bucket NAME	name of the destination S3 bucket (required)
-p, --prefix PATH	path prefix of the file on S3
-k, --aws-key-id KEY_ID	AWS access key ID to export data (required)
-s, --aws-secret-key SECRET_KEY	AWS secret access key to export data (required)
-F, --file-format FILE_FORMAT	file format for exported data. Available formats are tsv.gz (tab-separated values per line) and jsonl.gz (JSON record per line). The json.gz and line-json.gz formats are default and still available, but only for backward compatibility. Use of this option is discouraged because of lower performance.
-O, --pool-name NAME	specify resource pool by name
-e, --encryption ENCRYPT_METHOD	export with server-side encryption with the ENCRYPT_METHOD
-a --assume-role ASSUME_ROLE_ARN	export with assume role with ASSUME_ROLE_ARN as role arn

Amazon Resource Names (ARNs) uniquely identify AWS resources.

Examples

A simple bulk export syntax might look like:

```
td table:export example_db table1 \
 --s3-bucket mybucket \
 -k KEY_ID \
 -s SECRET_KEY
```

Typical bulk export syntax most likely contains the following options:

```
td table:export <database_name> <table_name> \
   --s3-bucket <S3_BUCKET_NAME> \
   --prefix <S3_FILE_PREFIX> \
   --aws-key-id <AWS_KEY> \
   --aws-secret-key <AWS_SECRET_KEY> \
   --file-format jsonl.gz
```

See also:

- TD Table Export Command Reference
- Bulk Export Server-Side Encryption Support