# Amazon S3 Export Integration

You can write job results directly to AWS S3 from Treasure Data.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers scalability, data availability, security, and performance. You can use it to store and protect any amount of data for things such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides features for data organization and configuration of access controls for your business, organization, and compliance requirements.



## Prerequisites

- Basic knowledge of Treasure Data, including the TD Toolbelt.
- For AWS: the IAM User with s3:PutObject and s3:AbortMultipartUpload permissions. We recommend that you set no other permissions for the IAM User used for this connection.

## Limitations and Supported

- The query result limit for export to S3 is 100GB. If the query result exceeds the limit, you see the following message in the log:
  The number of chunks for multipart upload is exceeded.
  Try to split data by query or use the table:export command.
- The default export format is CSV RFC 4180.
- Output in TSV format is also supported.

## About S3 Server-Side Encryption

You can encrypt upload data with AWS S3 Server-Side Encryption. You don't need to prepare an encryption key. Data will be encrypted at server side with 256-bit Advanced Encryption Standard (AES-256).

Use the Server-Side Encryption bucket policy if you require server-side encryption for all objects that are stored in your bucket. When you have server-side encryption enabled, you don't have to turn on **use_sse** option. However, job results might fail if you have bucket policies to reject HTTP requests without encryption information.

```
$ td query \
--result 's3://accesskey:secretkey@/bucketname/path/to/file.csv?use_sse=true&sse_algorithm=AES256' \
-w -d testdb \
"SELECT code, COUNT(1) AS cnt FROM www_access GROUP BY code"
```

## About File Formats for S3

The default export format is CSV RFC 4180. Output in TSV format is also supported.

For both CSV and TSV formats, the following table lists options you can use to customize the final format of the files written into the destination:

| Name | Description | Restrictions | CSV default | TSV default | JSONL |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| format | Main setting to specify the file format | | csv | csv (Use ‘tsv’ to select the TSV format) | Use JSONL to select JSONL format |
| delimiter | Use to specify the delimiter character | | , (comma) | \t (tab) | parameter ignored |
| quote | Use to specify the quote character | not available for TSV format | “ (double quote) | (no character) | parameter ignored |
| escape | Specifies the character used to escape other special characters | not available for TSV format | “ (double quote) | (no character) | parameter ignored |
| null | Use to specify how a ‘null’ value is displayed | | (empty string) | \N (backslash capital n) | parameter ignored |
| newline | Use to specify the EOL (End-Of-Line) representation | | \r\n (CRLF) | \r\n (CRLF) | |
| header | Can be used to suppress the column header | | column header printed. Use ‘false’ to suppress | the column header printed. Use ‘false’ to suppress | parameter ignored |

The following example shows a default sample output in CSV format when no customization is requested:

```
code,cnt
"200",4981
"302",
"404",17
"500",2
```

When the format=tsv, delimiter=“, and null=NULL options are specified:

```
$ td query \
--result 's3://accesskey:secretkey@/bucket_name/path/to/file.tsv?format=tsv&delimiter=%22&null=empty' \
-w -d testdb \
"SELECT code, COUNT(1) AS cnt FROM www_access GROUP BY code"
```

The access key and secret key must be URL encoded.
the output changes to:

```
"code" "cnt"
"200" 4981
"302" NULL
"404" 17
"500" 2
```

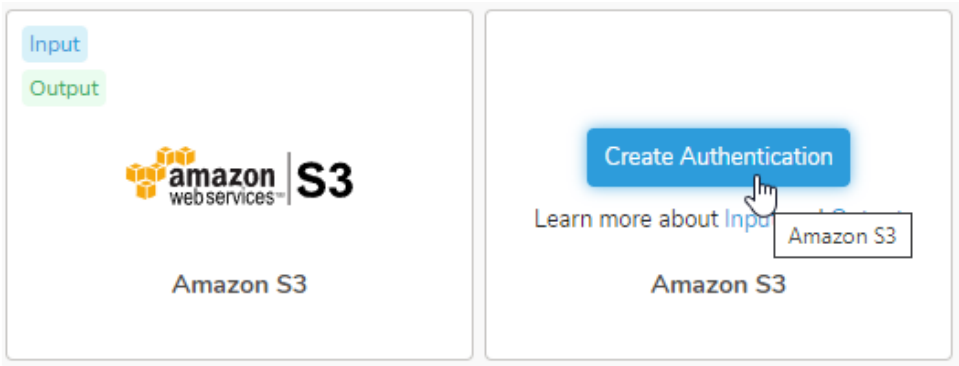
# Configure Results Export to your AWS S3 Instance

Exporting from Treasure Data requires queries. You can create or reuse a query. In the query, you configure the data connection.

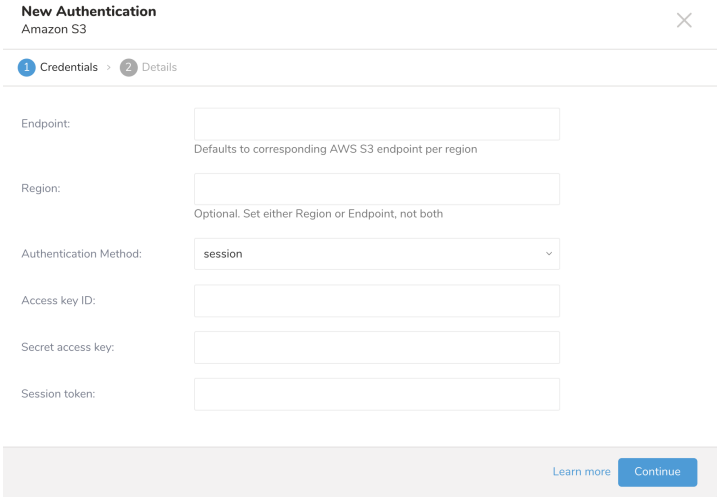
## Create a New Connection

When you configure a data connection, you provide authentication to access the integration. In Treasure Data, you configure the authentication and then specify the source information.

1. Navigate to **Integrations Hub** > **Catalog** and search for AWS S3.
2. Select **Create Authentication.**

3. 


New Authentication dialog opens. You need a client ID and access keys to authenticate using credentials.




4. Set the following parameters. Select **Continue**. Name your new AWS S3 connection. Select **Done**.

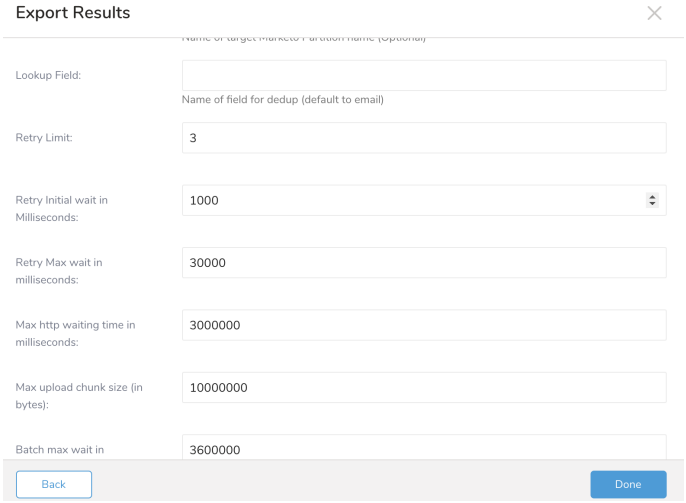
| Parameter | Description |
|---|---|
| **Endpoint** | • S3 endpoint login user name. You can find region and endpoint information from AWS Document. (Ex. *s3-ap-northeast-1.amazonaws.com*) |
| **Authentication Method** | |
| **basic** | • Uses access_key_id and secret_access_key to authenticate. See AWS Programmatic access.<br>  ◦ Access Key ID<br>  ◦ Secret access key |
| **anonymous** | • Uses anonymous access. This auth method can access only public files. |
| **session (Recommended)** | • Uses temporary-generated access_key_id, secret_access_key and session_token. (This authentication method is only available with data import. This can't be used with data export for now.)<br>  ◦ Access Key ID<br>  ◦ Secret access key<br>  ◦ Secret token |
| **Access Key ID** | AWS S3 issued |
| **Secret Access Key** | AWS S3 issued |

# Define your Query

1. Complete the instructions in Creating a Destination Integration.
2. Navigate to **Data Workbench > Queries**.
3. Select a query for which you would like to export data.

4. Run the query to validate the result set.
5. Select **Export Results**.
6. Select an existing integration authentication.

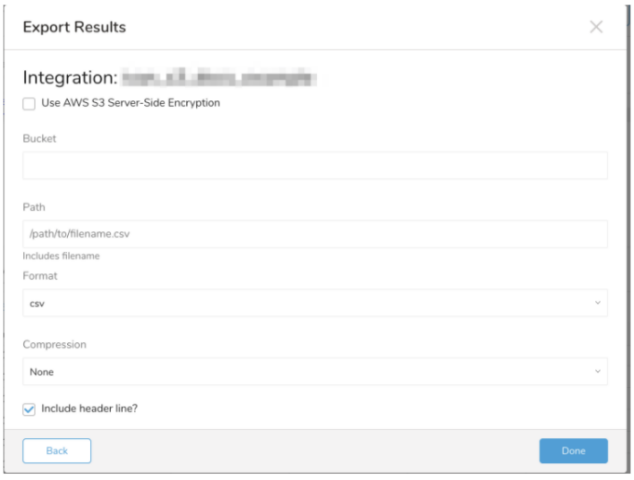

7. Define any additional Export Results details. In your export integration content review the integration parameters. For example, your Export Results screen might be different, or you might not have additional details to fill out:




8. Select **Done**.
9. Run your query.
10. Validate that your data moved to the destination you specified.

## Integration Parameters for S3

Define the following transfer parameters:




- **If `Use AWS S3 Server-Side Encryption` box is checked:**
    - **Server-Side Encryption algorithm.** *(Ex. AES256)*
- **Bucket**: Provide the S3 bucket name (Ex. your_bucket_name)
- **Path**: Specify a prefix for target keys. (Ex. logs/data_)
- **Format**: Format of the exported files (Ex. *csv (comma separated or tab separated*))
- **Compression**: The compression format of the exported files *(Ex. None or gz)*

- **Delimiter**: Use to specify the delimiter character *(Ex, (comma))*
- **String for null cells**: Placed holder to insert for null values *(Ex. Empty String)*
- **End-of-line character**: Specify the EOL(End-Of-Line) representation *(Ex. CRLF)*
- **Quote Character (Optional)**: The character used for quotes in the exported file(Ex. "). Only quote those fields which contain delimiter, quote, or any of the characters in lineterminator.
- **Escape character (Optional)**: The escape character used in the exported file

### Example Query

For example:

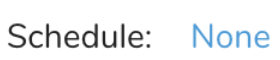
```
SELECT code, COUNT(1) AS cnt FROM www_access GROUP BY code
```

1. Verify the results in the Amazon S3 bucket that you specified when entering the transfer details.

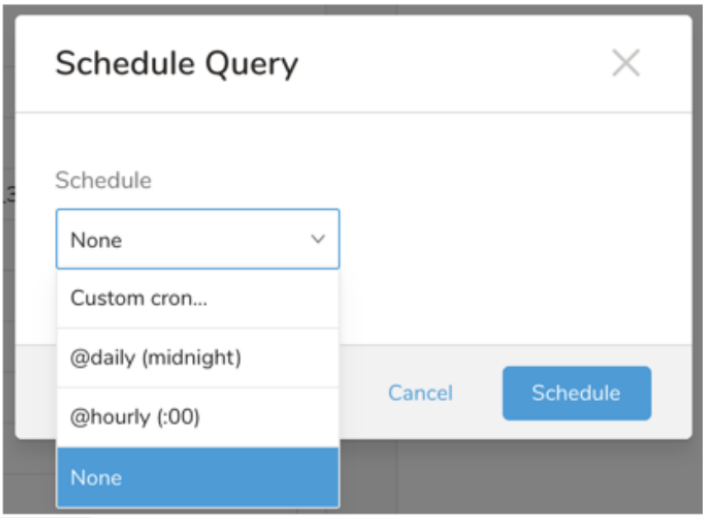
## Optionally Schedule the Query Export Jobs

You can use Scheduled Jobs with Result Export to periodically write the output result to a target destination that you specify.

1. Navigate to **Data Workbench > Queries**.
2. Create a new query or select an existing query.
3. Next to **Schedule**, select None.

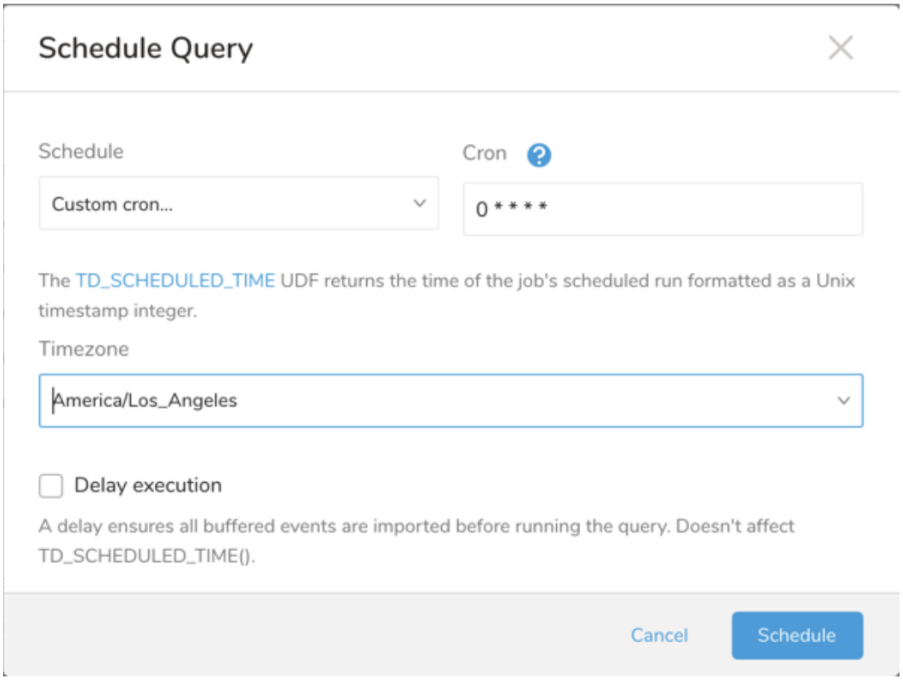



4. In the drop-down, select one of the following schedule options.




| Drop-down Value | Description |
|---|---|
| Custom cron... | Review Custom cron... details. |
| @daily (midnight) | Run once a day at midnight (00:00 am) in the specified time zone. |
| @hourly (:00) | Run every hour at 00 minutes. |
| None | No schedule. |

## Custom cron... Details




| Cron Value | Description |
|---|---|
| `0 * * * *` | Run once an hour |
| `0 0 * * *` | Run once a day at midnight |
| `0 0 1 * *` | Run once a month at midnight on the morning of the first day of the month |
| "" | Create a job that has no scheduled run time. |

```
 *    *    *    *    *
 -    -    -    -    -
 |    |    |    |    |
 |    |    |    |    +----- day of week (0 - 6) (Sunday=0)
 |    |    |    +---------- month (1 - 12)
 |    |    +--------------- day of month (1 - 31)
 |    +-------------------- hour (0 - 23)
 +------------------------- min (0 - 59)
```

The following named entries can be used:

- Day of Week: sun, mon, tue, wed, thu, fri, sat
- Month: jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec

A single space is required between each field. The values for each field can be composed of:

| Field Value | Example | Example Description |
|---|---|---|
| a single value, within the limits displayed above for each field. | | |
| a wildcard `'*'` to indicate no restriction based on the field. | `'0 0 1 * *'` | configures the schedule to run at midnight (00:00) on the first day of each month. |
| a range `'2-5'`, indicating the range of accepted values for the field. | `'0 0 1-10 * *'` | configures the schedule to run at midnight (00:00) on the first 10 days of each month. |

| | | |
|---|---|---|
| a list of comma-separated values `‘2,3,4,5’`, indicating the list of accepted values for the field. | `0 0 1,11,21 * *’` | configures the schedule to run at midnight (00:00) every 1st, 11th, and 21st day of each month. |
| a periodicity indicator `‘*/5’` to express how often based on the field’s valid range of values a schedule is allowed to run. | `‘30 */2 1 * *’` | configures the schedule to run on the 1st of every month, every 2 hours starting at 00:30. `‘0 0 */5 * *’` configures the schedule to run at midnight (00:00) every 5 days starting on the 5th of each month. |
| a comma-separated list of any of the above except the `‘*’` wildcard is also supported `‘2,*/5,8-10’`. | `‘0 0 5,*/10,25 * *’` | configures the schedule to run at midnight (00:00) every 5th, 10th, 20th, and 25th day of each month. |

5. (Optional) If you enabled the Delay execution, you can delay the start time of a query.

### Execute the Query

Save the query with a name and run, or just run the query. Upon successful completion of the query, the query result is automatically imported to the specified container destination.

> Scheduled jobs that continuously fail due to configuration errors may be disabled on the system side after several notifications.

## Optionally Configure Export Results in Workflow

Within Treasure Workflow, you can specify the use of this data connector to export data.

- About Using Workflows to Export Data with TD Toolbelt for more information on using data connectors in the workflow to export data.
- Treasure Boxes to see an example workflow.
- About Workflow Secrets Management to learn more about how to configure secrets to mask credentials in your workflow.

Learn more at Using Workflows to Export Data with the TD Toolbelt.

```
timezone: UTC

_export:
  td:
    database: sample_datasets

+td-result-into-s3:
  td>: queries/sample.sql
  result_connection: your_connections_name
  result_settings:
    bucket: your_bucket
    path: /path/file_${moment(session_time).format("YYYYMMDD")}.csv.gz
    compression: 'gz'
    header: true
    newline: \r\n
    "null": "hoge"
```

-

## Using the CLI to Export Results to AWS S3

If the TD Console is not available or does not meet your needs, you can use the CLI to issue queries and output results. You format the query output results using the CLI.

## Required

The access key and secret key must be URL encoded.

## Define the Query Export in CLI

To output the result of a single query to an S3 buck add the --result option to the td query command. After the job is finished, the results are written into your database:

For on-demand jobs, just add the --result option to the td query command. After the job is finished, the results are written to the S3 bucket with the given name and path. The access key and secret key must be URL encoded.

```
$ td query \
--result 's3://accesskey:secretkey@/bucketname/path/to/file.csv.gz?compression=gz' \
-w -d testdb \
"SELECT code, COUNT(1) AS cnt FROM www_access GROUP BY code"
```

For security reasons, you may want to use AWS IAM to manage storage write access permissions.

You can specify the compression option (only gz is allowed at this moment) in —result URL to compress the result. Without the compression parameter, it generates uncompressed data. The access key and secret key must be URL encoded.

```
$ td query \
--result 's3://accesskey:secretkey@/bucketname/path/to/file.csv' \
-w -d testdb \
"SELECT code, COUNT(1) AS cnt FROM www_access GROUP BY code"
```