Starting to Work with Jethro

This topic specifies how to create and use the Jethro database, which imports data from Hadoop and converts it to indexed tables, thereby significantly accelerating all searches within the imported data.
The creation and operation of the Jethro database are carried out by using the steps described below.

Creating Tables

The Jethro loader utility ***JethroLoader*** allows for fast and efficient data loading from text files into a Jethro table. It parses its input into rows and loads them into an existing Jethro table.

In production environments where large data loads are running during user activity, it is recommend to run the *JethroLoader* service from a separate host, to minimize the impact on running queries.

This following distinction between an input file and a table are used in this topic:

- The input file is made of **lines**. Each line has one or more fields.
- A Jethro table is made of **rows**. Each row has one or more columns.

To create a regular table, Run a *CREATE TABLE* statement and provide the table name and list of columns; for example:

```
CREATE TABLE my_table (a INT, b STRING, c TIMESTAMP).
```

The supported data types are: **INT | BIGINT | FLOAT | DOUBLE | STRING | TIMESTAMP.**

Jethro indexes all columns automatically – there is no need to run CREATE INDEX commands.

Jethro also supports working with views and partitioned tables. Check the reference guide for more details, as well as information on each data type and more table functions, such as renaming tables, dropping tables, and dropping columns.

The *CREATE TABLE* command can also be issued using a external script file, using the syntax:

```
JethroClient {instance_name} {instance_address:port} –p {user} –i {scriptfile}
```

For example, assuming we have a file named ***sales_demo.ddl*** with this script:

```
CREATE TABLE sales_demo
(
customer_sk INTEGER,
customer_salutation STRING,
customer_first_name STRING,
customer_last_name STRING,
customer_preferred_flag STRING,
customer_birth_year INTEGER,
customer_birth_country STRING,
customer_gender STRING,
customer_marital_status STRING,
customer_education_status STRING,
store_sk INTEGER,
store_name STRING,
store_city STRING,
store_county STRING,
store_state STRING,
store_country STRING,
item_product_name STRING,
item_class STRING,
item_category STRING,
item_size STRING,
item_color STRING,
sale_date TIMESTAMP,
sale_is_holiday STRING,
sale_is_weekend STRING,
quantity DOUBLE,
list_price DOUBLE,
net_paid DOUBLE,
net_profit DOUBLE
);
```

The *CREATE TABLE* command for a table called sales_demo to be created from such file is simply:

```
JethroClient demo localhost:9111 –p jethro –i sales_demo.ddl
```

Creating the Description File

A description (.desc) file describes the structure of an input file, the way to map it to the schema, and special formatting rules. This file is composed of the following sections:

- **Table-level section**: Describes the format of the input file and several processing options.
- **Column-level section**: A mapping between the fields in the file and columns in the table.
- **Record description section***:* (Optional, a special clause for handling variable format files, namely files with different format per line (see also Reference Guide, under *Input Files with Variable Format*).

Description file example:

```
TABLE sales_demo
row format delimited
fields terminated by '|'
null defined as 'NULL'
(
customer_sk,
customer_salutation,
customer_first_name,
customer_last_name,
customer_preferred_flag,
customer_birth_year,
customer_birth_country,
customer_gender,
customer_marital_status,
customer_education_status,
store_sk,
store_name,
store_city,
store_county,
store_state,
store_country,
item_product_name,
item_class,
item_category,
item_size,
item_color,
sale_date,
sale_is_holiday,
sale_is_weekend,
quantity,
list_price,
net_paid, net_profit )
```

Loading Data

Data loading is performed using the JethroLoader command line tool, which expects three parameters:
JethroLoader {instance-name} {description-file} {stdin} | {input-location [input-location]}...

- **Instance name**: The name of the local instance to connect
- **Description file**: The name of the .desc (description) file
- **Input location**: Either stdin (to load from a pipe) or a list of one or one locations

Each location is a path of a file or directory to be loaded.
Files can be a mix of uncompressed and compressed files (.gz or .gzip extension); compressed files are automatically uncompressed in-memory when they are read.

Examples:

```
JethroLoader demo t1.desc t1_input_dir
JethroLoader demo t1.desc t1.part1.csv t1.part2.csv
JethroLoader demo t1.desc HDFS://home/jethro/files/t1.gz
sed ... | JethroLoader demo t1.desc stdin
```


When loading from a pipe, the program that sends data to JethroLoader may fail during the process. In such a case the loader is not notified of an error; it gets a standard end-of-file (EOF) notification for its input, and therefore will commit and exit without error.
It is your responsibility to monitor for such cases and respond accordingly.

Demo Data

Jethro offers demo data files for the sales_demo table described above that can be freely downloaded and then loaded into Jethro for evaluation purposes. It includes 100 million rows (about 3GB). A lighter 10M rows version (about 300MB) is also available.

- 100m Download: https://jethro.io/demodata-100m
- 10m Download: https://jethro.io/demodata-10m

To open the downloaded archive, run:

```
tar -xvf {filename}
```

Along with the compressed .csv, the archive contains the CREATE TABLE script (.ddl file), the .desc file, and various SQL queries.

Creating the demo data table

Use JethroClient to run the CREATE TABLE script. You need to specify the instance name, the host and port of the JethroServer, the password (default is jethro) and in our case, also the script file:

```
JethroClient demo localhost:{port_number} -p jethro -i sales_demo.ddl
```

Loading the demo data

Use JethroLoader to load the CSV file. Because loading would take a while, it is advisable to run the loading operation in the background:

```
JethroLoader demo sales_demo.desc sales_demo.100m.csv.gz
or
JethroLoader demo sales_demo.desc sales_demo.10m.csv.gz
```

Check the loader results by looking at its report file. The report file resides under **/var/log/jethro/{instance-name}/loader**, and its name follows the convention **loader_report_{timestamp}_{pid}.log**; for example:
more loader_report_20140630_124723_9359.