Automatic Cube Generation

Jethro auto cubes brings the great performance value of pre-aggregated cubes, combined with the ability to automatically generate and transparently maintain those cubes.
Cubes can be useful to accelerate performance for certain sets of queries, mainly queries that are not highly filtered and would normally scan large parts of the data set. Jethro optimizer combines cubes with indexing and caching technology to ensure interactive queries response time on wide range of use cases.

Cubes can be generated in two modes:

- Auto-generation, based on queries sent to the server.
- Manual cube generation for a given query, by using the GENERATE CUBES command.

Auto cube generation process is transparent to the user. Jethro intercepts incoming queries and automatically generates the appropriate cubes. The initial cube generation phase for BI dashboard is called **training phase**. Training phase for a new dashboard usually takes anywhere between few minutes to an hour.

Requirements and Initial Setup

Because cube generation can be a resource-consuming process, it is advisable to allocate a designated Jethro node for this task. If a designated node cannot be allocated, it is recommended turn on cube auto-generation only when business users are not accessing the system.

JethroMaint service is responsible for intercepting incoming queries and sending cube generation request to the designated cube generation execution server. By default, the cube generation host is 'auto', which means 'localhost' (and the default port is identified automatically by the system). To change server address or port, set the following parameter: ***dynamic.aggregation.auto.generate.execution.hosts.***

For example, if the cube generation node DNS is 'cubes.gen', and the JethroServer is running on port 9112, run the following set global command from SQL client to connect to the instance:
set global dynamic.aggregation.auto.generate.execution.hosts=cubes.gen:9112;

It is also possible to assign more than one query node for cubes execution. In that case, the maint service will send each of the cube execution requests to the server which is not busy, according to the queue of requests. To do that, use the following format to describe the hosts: <host:port>;[(<host:port>;...n)]. 'auto' can also be used as one of the hosts assigned. For example:

set global dynamic.aggregation.auto.generate.execution.hosts=cubes.gen:9112;10.1.1.63:9112;auto;

Cubes are persistently stored at the instance storage location (HDFS, NFS, or local file system). When the total cubes' size exceeds the maximum storage size allocated for cubes, unused cubes are automatically dropped. The default storage size is set to 100GB. To set a different size, run the following set global command from SQL client to connect to the instance:

set global adaptive.cache.quota=<Required size in bytes>

Turn on Automatic Cube Generation

By default, auto cube generation is turned off. The turn cube generation state (on/off) is controlled by the following parameter :
dynamic.aggregation.auto.generate.enable.
Use *set global* to enable/disable auto cube generation.

To enable auto cube generation, set the value to 1:
set global dynamic.aggregation.auto.generate.enable=1;

To disable auto cube generation, set the value to 0:
set global dynamic.aggregation.auto.generate.enable=0;

Manual Cube Generation

Cubes can be manually created from a query, by running the **GENERATE CUBES** command, which generates one or more cubes from a given query. The **GENERATE CUBES** command also allows generating cubes with a WHERE statement. For further information, see Generate Cubes under SQL Reference.

Before generating a cube, it is also possible to see all the cubes that can be used by a given query. The command **SHOW CUBES {WITH WHERE <where-clause>} FROM <select-statement>**, can show all the cubes that a query might looking for during its execution, together with a status that show whether this cube can be generated, whether it's already exists, or whether it can't be generated (The result set size of it is too big, or it is not a valid query for a cube).

Cubes Management

To display all generated cubes, use:

SHOW CUBES

To remove all cubes from the cube repository, use:

DROP CUBES

When new data is loaded, cubes are automatically updated in the background. The **maint** service is responsible for updating cubes by sending the following command to the cube generation server. Cubes can also be updated manually by running the command:

UPDATE CUBE BY KEY

Cubes Monitoring

Cubes Queues

Queries with execution time that exeeds the threshold defined (The default threshold is 1 seconed) are sent to a queue of queries that the maint service needs to check. From there, if they are found to match a possible or and existing cube, they are sent to one of 4 possible queues:

- **Generate** - generate a new cube for a new query.
- **Regenerate** - Recreation of a cube from scratch after a schema change (i.e adding/dropping a column relevant to the cube's query, or after loading was done in OVERWRITE mode).
- **Update Incremental** - Incrementaly update cubes due to a new data that was loaded (in APPEND mode).
- **Update Final** - Fully updating a 'final' cube due to a new data that was loaded (in APPEND mode).

There is a priority between them, when the maint service needs to choose which cube will be sent to the cube execution hosts first. The priority is first to send **Update Incremental** queries, then **Update Final,** then **Regenerate**, and then **Generate.** Those queues are active as default, and it is possible to disable some/all of them by setting their proper parameters to 0 from a SQL client:

set global dynamic.aggregation.auto.generate.new.cubes.enable=0;
set global dynamic.aggregation.auto.regenerate.cubes.enable=0;
set global dynamic.aggregation.auto.update.incremental.cubes.enable=0;
set global dynamic.aggregation.auto.update.final.cubes.enable=0;

To monitor the amount of queries accumulated on each queue, run:

tail -f /var/log/jethro/<instance-name>/jethro.log | grep "Queue sizes"

Cubes generation

Jethro.log can also be used to monitor cubes generation:

The following message in the log, notifies about the received query and the beginning of the cube generation process:

GENERATE CUBES autoGen FROM {query}

For example:
GENERATE CUBES autoGen FROM SELECT store_country, store_name, SUM(net_profit) FROM sales_demo WHERE store_country='United States' GROUP BY store_country, store_name

The following message notifies about the actual query which will use for building the cube:
Cube candidate: {query}

Fields which appeared in the WHERE clause of the original query, will be inserted to the GROUP BY clause of the cube. For the example above, the cube candidate will be:

SELECT sales_demo.store_country, sales_demo.store_name, sum(sales_demo.net_profit) FROM sales_demo GROUP BY sales_demo.store_country, sales_demo.store_name

The following message notifies about the end of the cube generation process and its results:

CUBE GENERATED SUCCESSFULLY!