ColumnStore - First class citizen in MariaDB

Vicențiu Ciorbaru
Software Developer Team Lead
@ MariaDB Foundation

Percona Live 2021
2021 May - MariaDB Community Track
## Conclusion

<table>
<thead>
<tr>
<th></th>
<th>InnoDB</th>
<th>ColumnStore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert 100 mil rows</td>
<td>3min 12s</td>
<td>10s</td>
</tr>
<tr>
<td>select count(*)</td>
<td>16.902s</td>
<td>0.45s</td>
</tr>
<tr>
<td>select max(unindexed_col)</td>
<td>20.435s</td>
<td>1.201s</td>
</tr>
<tr>
<td>select avg(unindexed_col)</td>
<td>25.101s</td>
<td>1.487s</td>
</tr>
</tbody>
</table>
## Row based storage

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>City</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>Berlin</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>London</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>New York</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>Beijing</td>
<td>51</td>
</tr>
</tbody>
</table>

One file per table
Row based storage

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>City</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>Berlin</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>London</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>New York</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>Beijing</td>
<td>51</td>
</tr>
</tbody>
</table>

To select one column, requires accessing all table rows.
# Row based storage

Modifying all values in a column, updates all rows.

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>City</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>Berlin</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>London</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>New York</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>Beijing</td>
<td>51</td>
</tr>
</tbody>
</table>
Row based storage

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>City</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>Berlin</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>London</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>New York</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>Beijing</td>
<td>51</td>
</tr>
</tbody>
</table>

Deleting a row, single seek and delete
# Row based storage

## Clients

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>City</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>Berlin</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>London</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>New York</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>Beijing</td>
<td>51</td>
</tr>
<tr>
<td>5</td>
<td>Jane</td>
<td>2021-04-01</td>
<td>IE</td>
<td>Dublin</td>
<td>31</td>
</tr>
</tbody>
</table>

Inserting a row just appends
### Column based storage

#### Clients

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>51</td>
</tr>
</tbody>
</table>

One file per column
## Column based storage

Reading a column accesses only that column

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>51</td>
</tr>
</tbody>
</table>
### Column based storage

Modifying a column accesses only that column

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>51</td>
</tr>
</tbody>
</table>
## Column based storage

### Clients

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>51</td>
</tr>
</tbody>
</table>

Deleting a row accesses **all** files
### Clients

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Enrollment Date</th>
<th>Country</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>2021-04-01</td>
<td>DE</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Bill</td>
<td>2020-03-17</td>
<td>GB</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>2021-01-07</td>
<td>US</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Liu</td>
<td>2021-02-06</td>
<td>CN</td>
<td>51</td>
</tr>
<tr>
<td>5</td>
<td>Jane</td>
<td>2021-04-01</td>
<td>IE</td>
<td>31</td>
</tr>
</tbody>
</table>

Inserting a row accesses **all** files
## Differences between OLTP and OLAP

<table>
<thead>
<tr>
<th>OLTP</th>
<th>OLAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Short transactions</td>
<td>● Fewer, but complex queries.</td>
</tr>
<tr>
<td>● Many concurrent connections.</td>
<td>● Few concurrent connections</td>
</tr>
<tr>
<td>● Data inserted in small increments</td>
<td>● Bulk loading of data</td>
</tr>
</tbody>
</table>
Architectures

**MariaDB & InnoDB**
- Threadpool to manage incoming connections.
- Optimized for parallel transactions, each query runs in a single thread.

**MariaDB & ColumnStore**
- Query is passed down to ColumnStore engine
- ColumnStore executes queries in parallel
Architectures

MariaDB ColumnStore
Installing MariaDB ColumnStore

- Setup a MariaDB repository
  https://mariadb.org/download/#mariadb-repositories

Choose a distribution

- Debian 10 "buster"

Choose a MariaDB Server version

- 10.5

Mirror

- Chroot Network - Bucharest

Here are the commands to run to add MariaDB to your system:

```
sudo apt-get install software-properties-common dirmngr apt-transport-https
sudo apt-key adv --fetch-keys 'https://mariadb.org/mariadb_releaseSigning_key.asc'
sudo add-apt-repository 'deb [arch=amd64] https://mirrors.chroot.ro/mariadb/repo/10.5/debian buster main'
```

Once the key is imported and the repository added you can install MariaDB with:

```
sudo apt-get update
sudo apt-get install mariadb-server
```
Installing MariaDB ColumnStore

- # apt-get install mariadb-plugin-columnstore
Installing MariaDB ColumnStore

- `# apt-get install mariadb-plugin-columnstore`
- `# systemctl start mariadb-columnstore`
Installing MariaDB ColumnStore

- `# apt-get install mariadb-plugin-columnstore`
- `# systemctl start mariadb-columnstore`

DONE!
1. Insert via regular INSERT statements:

MariaDB [statistics]>
-> INSERT into sat_scores (id, score) values (100, 1060);
Query OK, 1 row affected (0.240 sec)

- For 100mil rows -> 277 Days!
2. Insert via INSERT ... SELECT

MariaDB [statistics]>
    insert into sat_scores_cs (id, score) 
    (select * from sat_scores);
Query OK, 100000000 rows affected (2 min 2.314 sec)

- As fast as it takes InnoDB to scan the whole table
  - Single threaded because of MariaDB's single thread select.
Inserting data into MariaDB ColumnStore

3. Convert InnoDB table to ColumnStore

   MariaDB [statistics]> alter table sat_scores engine=ColumnStore;
   ERROR 1069 (42000): Too many keys specified; max 0 keys allowed

   ● ColumnStore doesn't support keys

   ● Performance is as slow as single insert
4. LOAD DATA INFILE

MariaDB [statistics]>
   load data infile './data.csv' into table sat_scores_cs
   fields terminated by ',' optionally enclosed by '\'';
Query OK, 100000000 rows affected (1 min 13.297 sec) done
Records: 100000000  Deleted: 0  Skipped: 0  Warnings: 0

● Uses ColumnStore's **cpimport** tool behind the scenes.
Inserting data into MariaDB ColumnStore

5. Use cpimport directly

```
sudo cpimport statistics sat_scores_cs -s , < data.csv
```

... 
2021-04-30 20:35:32 (29561) INFO : Bulk load completed, total run time : 10.0944 seconds

- Best tool to use for fast inserts!
- Configurable number of worker threads.
## Conclusion

<table>
<thead>
<tr>
<th>Operation</th>
<th>InnoDB</th>
<th>ColumnStore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert 100 mil rows</td>
<td>3min 12s</td>
<td>10s</td>
</tr>
<tr>
<td>select count(*)</td>
<td>16.902s</td>
<td>0.82s</td>
</tr>
<tr>
<td>select max(unindexed_col)</td>
<td>20.435s</td>
<td>1.201s</td>
</tr>
<tr>
<td>select avg(unindexed_col)</td>
<td>25.101s</td>
<td>1.487s</td>
</tr>
</tbody>
</table>
Selecting data from MariaDB ColumnStore

- Uses regular SELECT interface

MariaDB [statistics]> select count(*) from sat_scores_cs where id > 2000000;
+----------+
| count(*) |
+----------+
| 97999999 |
+----------+
1 row in set (0.806 sec)

MariaDB [statistics]> explain select count(*) from sat_scores_cs where id > 2000000;
+--------------+--------------------------+-------+...
| id           | select_type              | table |...
+--------------+--------------------------+-------+...
| 1            | PUSHED SELECT            | NULL  |...
+--------------+--------------------------+-------+...
Selecting data from MariaDB ColumnStore

- Uses regular SELECT interface

MariaDB [statistics]> select count(*) from sat_scores_cs where id > 2000000;
+----------+
| count(*)  |
+----------+
| 97999999  |
+----------+
1 row in set (0.806 sec)

MariaDB [statistics]> explain select count(*) from sat_scores_cs where id > 2000000;
+---------------+-----------------+---------+...+-
| id            | select_type     | table   |...+--
| 1             | PUSHED SELECT   | NULL    |...+-

Query Plan not that useful
Selecting data from MariaDB ColumnStore

- ColumnStore QueryPlan has a different format

MariaDB [statistics]> select colSetTrace(1);
Selecting data from MariaDB ColumnStore

- ColumnStore QueryPlan has a different format

MariaDB [statistics]> select calSetTrace(1);

```
MariaDB [statistics]> select row_number() over (),
    max(t1.score)
from sat_scores_cs t1
group by id mod 100000;
```

```
100000 rows in set, 1 warning (7.569 sec)
```
Selecting data from MariaDB ColumnStore

- ColumnStore QueryPlan has a different format

```sql
MariaDB [statistics]> select calSetTrace(1);

MariaDB [statistics]> select row_number() over (),
    max(t1.score)
from sat_scores_cs t1
group by id mod 100000;

...  
100000 rows in set, 1 warning (7.569 sec)
```

```sql
MariaDB [statistics]> select calGetTrace()\G
*************************** 1. row ***************************
calGetTrace():
Desc Mode Table TableOID ReferencedColumns PIO   LIO    PBE Elapsed Rows
BPS  PM   t1    3002     (id,score)        97660 695329 0   7.440   19200000
TAS  UM   -     -        -                 -     -      -   7.385   100000
WFS  UM   -     -        -                 -     -      -   0.018   100000
TNS  UM   -     -        -                 -     -      -   0.004   100000
```
Selecting data from MariaDB ColumnStore

- **ColumnStore QueryPlan has a different format**

  ```
  MariaDB [statistics]> select calSetTrace(1);
  ...
  100000 rows in set, 1 warning (7.569 sec)
  ```

  ```
  MariaDB [statistics]> select row_number() over (),
  max(t1.score)
  from sat_scores_cs t1
  group by id mod 100000;
  ...
  100000 rows in set, 1 warning (7.569 sec)
  ```

**Execution Steps**

<table>
<thead>
<tr>
<th>Desc</th>
<th>Mode</th>
<th>Table</th>
<th>TableOID</th>
<th>ReferencedColumns</th>
<th>PIO</th>
<th>LIO</th>
<th>PBE</th>
<th>Elapsed</th>
<th>Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPS</td>
<td>PM</td>
<td>t1</td>
<td>3002</td>
<td>(id,score)</td>
<td>97660</td>
<td>695329</td>
<td>0</td>
<td>7.440</td>
<td>19200000</td>
</tr>
<tr>
<td>TAS</td>
<td>UM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>7.385</td>
<td>100000</td>
</tr>
<tr>
<td>WFS</td>
<td>UM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.018</td>
<td>100000</td>
</tr>
<tr>
<td>TNS</td>
<td>UM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.004</td>
<td>100000</td>
</tr>
</tbody>
</table>
Selecting data from MariaDB ColumnStore

- **ColumnStore QueryPlan has a different format**

MariaDB [statistics]> select calSetTrace(1);

MariaDB [statistics]> select row_number() over (),
  max(t1.score)
  from sat_scores_cs t1
  group by id mod 100000;

...  
100000 rows in set, 1 warning (7.569 sec)

MariaDB [statistics]> select calGetTrace();

```
*************************** 1. row ***************************
calGetTrace():
  Desc  Mode  Table  TableOID  ReferencedColumns  PIO   LIO   PBE  Elapsed  Rows
BPS    PM   t1     3002     (id,score)      97660 695329 0   7.440   19200000
TAS    UM   -      -        -            -     -      -   7.385   100000
WFS    UM   -      -        -            -     -      -   0.018   100000
TNS    UM   -      -        -            -     -      -   0.004   100000
```

Selecting data from MariaDB ColumnStore

- ColumnStore QueryPlan has a different format

```sql
MariaDB [statistics]> select calSetTrace(1);

MariaDB [statistics]> select row_number() over (),
   max(t1.score)
from sat_scores_cs t1
group by id mod 100000;

...  
```

100000 rows in set, 1 warning (7.569 sec)

```sql
MariaDB [statistics]> select calGetTrace()\G
```

<table>
<thead>
<tr>
<th>Desc</th>
<th>Mode</th>
<th>Table</th>
<th>TableOID</th>
<th>ReferencedColumns</th>
<th>PIO</th>
<th>LIO</th>
<th>PBE</th>
<th>Elapsed</th>
<th>Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPS</td>
<td>PM</td>
<td>t1</td>
<td>3002</td>
<td>(id,score)</td>
<td>97660</td>
<td>695329</td>
<td>0</td>
<td>7.440</td>
<td>19200000</td>
</tr>
<tr>
<td>TAS</td>
<td>UM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>7.385</td>
<td>100000</td>
</tr>
<tr>
<td>WFS</td>
<td>UM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.018</td>
<td>100000</td>
</tr>
<tr>
<td>TNS</td>
<td>UM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.004</td>
<td>100000</td>
</tr>
</tbody>
</table>
## Selecting data from MariaDB ColumnStore

```
MariaDB [statistics]> select calGetTrace()\G
*************************** 1. row ***************************
calGetTrace():
Desc  Mode  Table  TableOID  ReferencedColumns  PIO  LIO  PBE  Elapsed  Rows
BPS  PM  t1  3002  (id,score)  97660  695329  0  7.440  19200000
TAS  UM  -  -  -  -  -  -  7.385  100000
WFS  UM  -  -  -  -  -  -  0.018  100000
TNS  UM  -  -  -  -  -  -  0.004  100000
```

CrossEngine Joins

- Requires ColumnStore to connect to MariaDB to fetch external engines data.

- Default values:
  
  ```
  CrossEngineSupport.Host = 127.0.0.1
  CrossEngineSupport.Port = 3306
  CrossEngineSupport.User = root
  CrossEngineSupport.Password =
  CrossEngineSupport.TLSCA =
  CrossEngineSupport.TLSClientCert =
  CrossEngineSupport.TLSClientKey =
  ```

- Use `mcsGetConfig` and `mcsSetConfig` to update the configuration
Conclusions

● ColumnStore is available as a separate Storage Engine in MariaDB since **10.5.4**

● Compared to traditional MariaDB Storage Engines, it has more moving parts.

● Much faster than InnoDB for Analytical Queries

● Inserts must be done in bulk

● Specific configuration steps, separate from mariadb.cnf

● Easy to start up with systemd, different flow to examine queries.
Thank you!

Contact details:

vicentiu@mariadb.org

About:

https://mariadb.org/vicentiu