Consistent Order

Overview

Consistent ordering in ClustrixDB guarantees that rows returned from repeated runs of the same query are always in the same order. If a query does not specify an explicit ORDER BY and the consistent_order global variable is set to false, the order in which rows are returned is not guaranteed to be the same for repeated executions.

The internal storage implementation for ClustrixDB is different from MySQL. Since ClustrixDB is a distributed database, the ordering of result sets will not be the same as those from MySQL. If an application's behavior depends on how the rows are ordered, specify an explicit ORDER BY clause. Consistent ordering will also apply when non-unique results are returned from ORDER BY.

Consistent ordering only applies to SELECT statements. It does not apply to INSERT, UPDATE, or DELETE statements. Please see the list of Caveats for Consistent Order below.

Enabling Consistent Ordering

Consistent ordering can be specified per session or globally. To enable consistent ordering, set the value for consistent_order to true.

```
sql> INSERT INTO foo VALUES (0),(1),
(2),(3);
sql> SELECT * FROM foo;
+----+
| id |
+----+
|  2 |
|  1 |
|  4 |
|  0 |
+----+
4 rows in set (0.00 sec)
sql> SET session consistent_order =
true;
sql> SELECT * FROM foo;
+----+
| id |
+----+
|  0 |
|  1 |
|  2 |
|  3 |
+----+
4 rows in set (0.00 sec)
```

Consistent Order may also be set globally.

```
sql> SET global consistent_order = true;
```

Overhead of Consistent Ordering

Enabling consistent ordering increases query performance overhead because ClustrixDB needs to:

1. Determine the appropriate ordering.
2. Perform a secondary compilation after the appropriate ordering is determined.
3. Return the query results in a specific order versus in-parallel, and thus very quickly.

The cost of enabling consistent ordering is that each query will take approximately twice as long to compile and will incur a performance penalty, as well. Each cluster's workload differs, so the best way to determine the specific overhead on your cluster is by inspection.

A Note on Joins

Consistent ordering requires the plan for a query to include steps that ensure the order. Such queries take longer to plan and compile. Queries that do a large number of joins have a higher compilation time, and should use explicit ordering instead.

One notable exception are aggregate queries for which the ORDER BY and GROUP BY specify the same ascending columns. Such queries do not incur the same planner impact.

Caveats for Consistent Order

Consistent ordering does not apply to SELECT statements with the following:

- unions
- sub-selects

Consistent ordering does not apply to INSERT, UPDATE, or DELETE statements.