BOOSTING PERFORMANCE OF ORDER BY LIMIT QUERIES

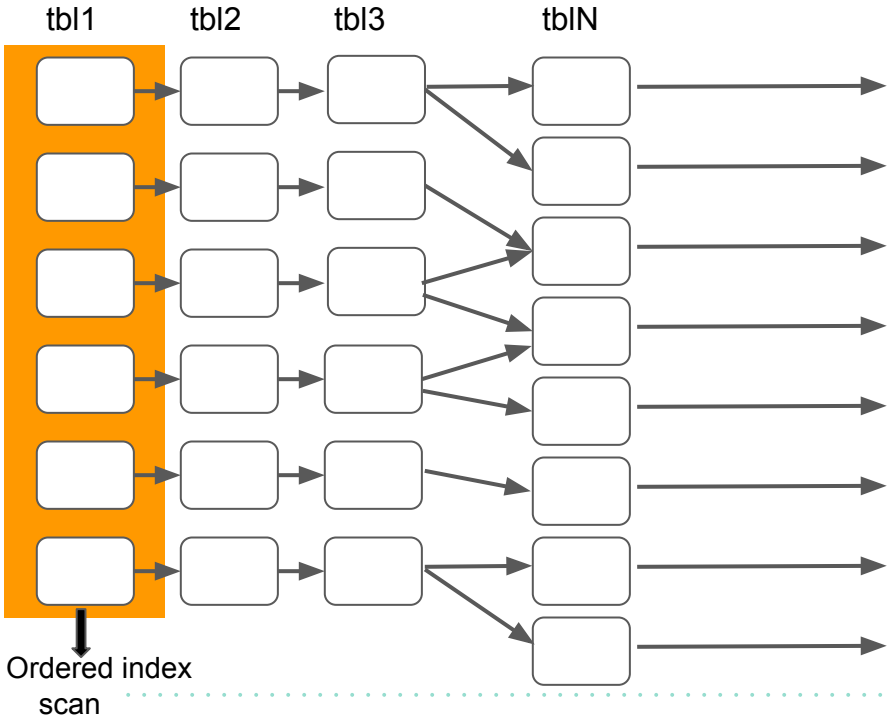
Varun Gupta
Optimizer Developer
MariaDB Corporation

Handling ORDER BY with LIMIT queries

Available means to produce ordered streams:

- Use an ordered index
  - Range access
  - Ref access (but not ref-or-null)
    - Result of ref(tbl.keypart1=const) are ordered by tbl.keypart2,t.keypart3…..
  - Index scan
- Use Filesort

Using index to produce ordered stream

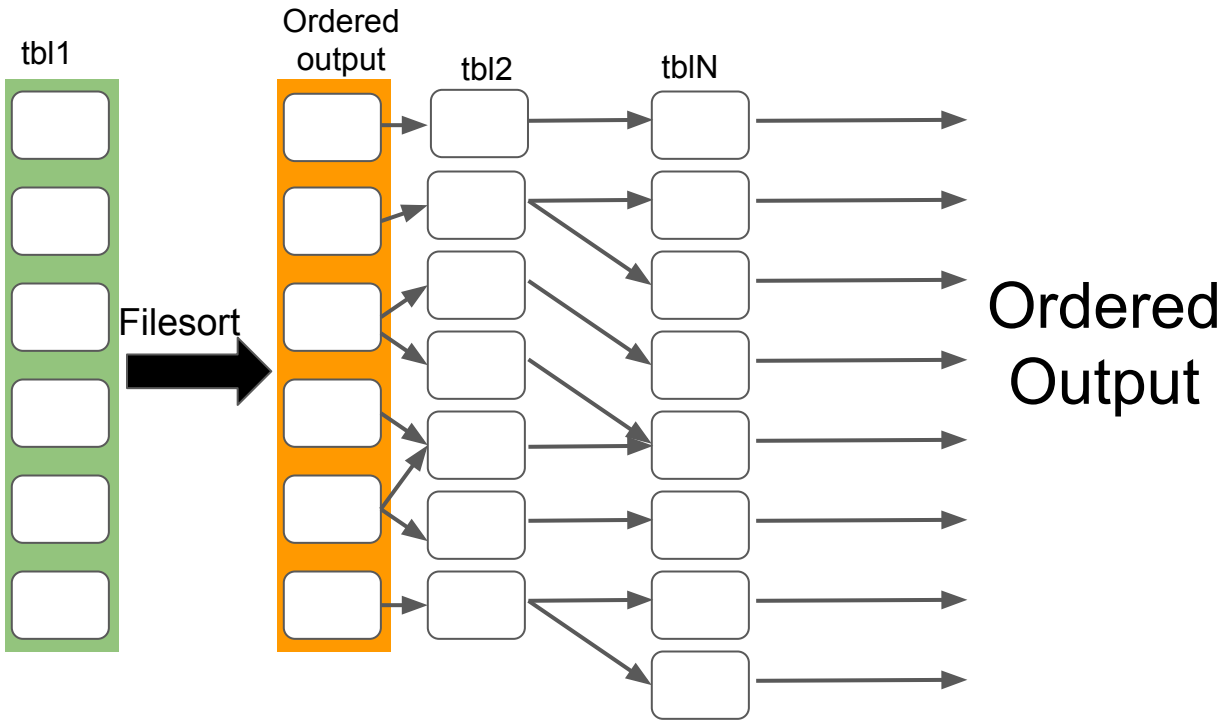


Ordered Output

- ORDER BY must use columns from one index
- DESC is ok if present for all the columns
- Cannot use join buffering as it breaks the ordering
- With LIMIT, the execution stops as soon as LIMIT records are enumerated

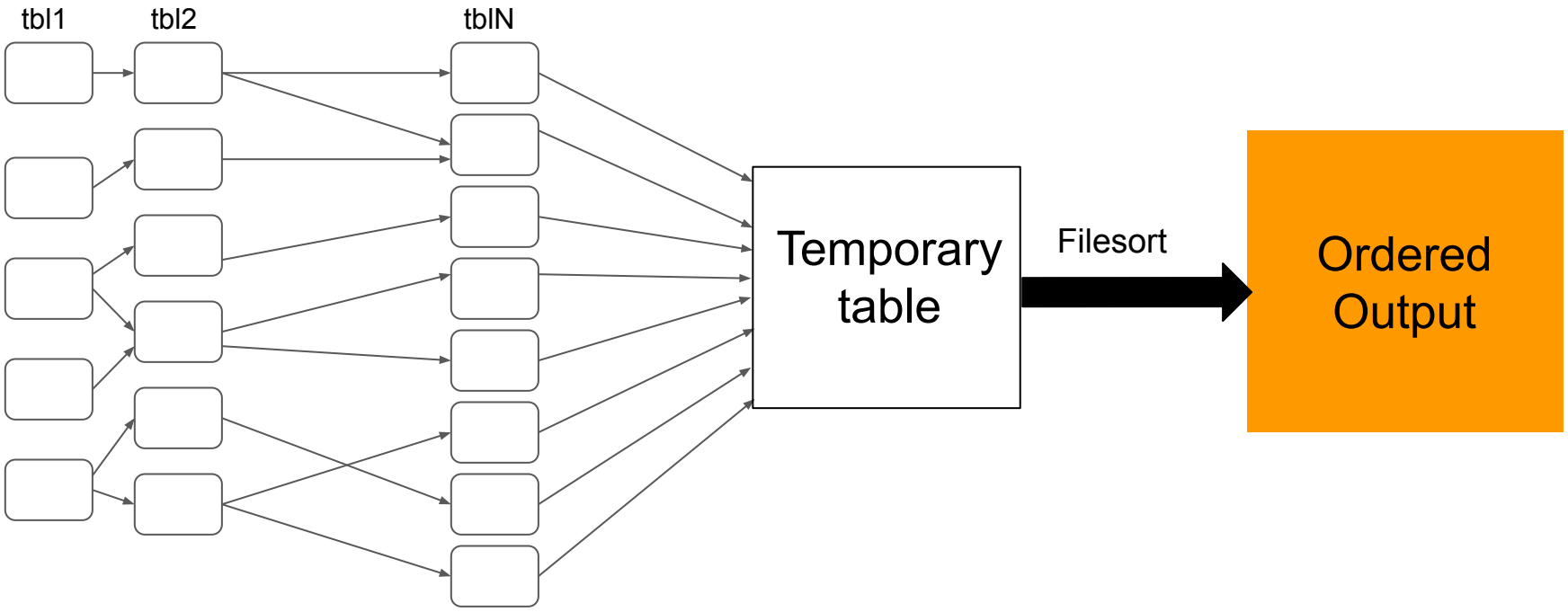
MariaDB

Using filesort on first non-const table



- Filesort is used on the first table instead of an index scan
- Cannot use join buffering as it breaks the ordering
- Condition on first table is checked before filesort
- EXPLAIN shows “Using filesort” in the first row
- With LIMIT, the execution stops as soon as LIMIT records are enumerated

Using filesort for entire join output



Using filesort for entire join output

- This is a catch-all method
  - Places no limit on join order, use of join buffering etc
- LIMIT is applied only after the entire join is computed. This could be very inefficient for smaller LIMIT.
- EXPLAIN shows “Using temporary;Using filesort” in the first row

ORDER BY with LIMIT and JOIN optimizer

Currently we have:

- Cost of sorting is not taken into account by the join planner
- LIMIT is not taken into account by the join planner
- Once the join order is fixed, we consider changing the access method on the first table (if LIMIT is present) to produce the required ordering. This approach is cost based.

LIMITATIONS (Example 1)

```
SELECT * FROM
t_fact
  JOIN dim1
    ON t_fact.dim1_id= dim1.dim1_id
ORDER BY t_fact.col1
LIMIT 1000;
```

EXECUTION TIME
25.289 sec

```
+-------+------+---------+---------+------------------+------+---------------------------------+
| table | type |  key    | key_len | ref              | rows | Extra                           |
+-------+------+---------+---------+------------------+------+---------------------------------+
| dim1  | ALL  |  NULL   | NULL    | NULL             | 500  | Using temporary; Using filesort |
| t_fact| ref  |  dim1_id| 4       | test.dim1.dim1_id| 2632 |                                 |
+-------+------+---------+---------+------------------+------+---------------------------------+
```

MariaDB

LIMITATIONS (Example 1)

```
SELECT * FROM
  t_fact
    STRAIGHT_JOIN dim1 on t_fact.dim1_id= dim1.dim1_id
ORDER BY t_fact.col1
LIMIT 1000;
```

EXECUTION TIME
0.013 sec

```
+-------+-------+-------+---------+--------------------+------+-------+
| table | type  |key    | key_len | ref                | rows | Extra |
+-------+-------+-------+---------+--------------------+------+-------+
| t_fact | index |col1   | 4       | NULL               | 1900 |       |
| dim1   | eq_ref |PRIMARY | 4       | test.t_fact.dim1_id | 1    |       |
+-------+-------+-------+---------+--------------------+------+-------+
```

MariaDB

LIMITATIONS (Example 2)

```
SELECT t0.ID_t0 , t1.ID
FROM t0
   INNER JOIN t1
      ON t0.ID_t1 = t1.ID
   INNER JOIN z2
      ON t0.ID_z2 = z2.ID AND (z2.ID_LOCALITE = 1)
ORDER BY t0.d
LIMIT 10;
```

EXECUTION TIME
5.151 sec

```
+-------+--------+-------------+---------+---------------+--------+-----------------------------------------------+
| table | type   | key         | key_len | ref           | rows   | Extra                                         |
+-------+--------+-------------+---------+---------------+--------+-----------------------------------------------+
| z2    | ref    | ID_LOCALITE | 4       | const         | 249828 | Using index; Using temporary; Using filesort |
| t0    | ref    | ID_z2       | 4       | test.z2.ID    | 1      | Using where                                   |
| t1    | eq_ref | PRIMARY     | 4       | test.t0.ID_t1 | 1      | Using index                                   |
+-------+--------+-------------+---------+---------------+--------+-----------------------------------------------+
```

MariaDB

LIMITATIONS (Example 2)

```
SELECT t0.ID_t0 , t1.ID
FROM
   t0 STRAIGHT_JOIN t1
     ON t0.ID_t1 = t1.ID
   STRAIGHT_JOIN z2
      ON t0.ID_z2 = z2.ID AND (z2.ID_LOCALITE = 1)
ORDER BY t0.d
LIMIT 10;
```

EXECUTION TIME
0.485 sec

```
+-------+--------+--------+---------+---------------+--------+-----------------------------+
| table | type   | ey     | key_len | ref           | rows   | Extra                       |
+-------+--------+--------+---------+---------------+--------+-----------------------------+
| t0    | ALL    | ULL    | NULL    | NULL          | 500000 | Using where; Using filesort |
| t1    | eq_ref | RIMARY | 4       | test.t0.ID_t1 | 1      | Using index                 |
| z2    | eq_ref | RIMARY | 4       | test.t0.ID_z2 | 1      | Using where                 |
+-------+--------+--------+---------+---------------+--------+-----------------------------+
```

MariaDB

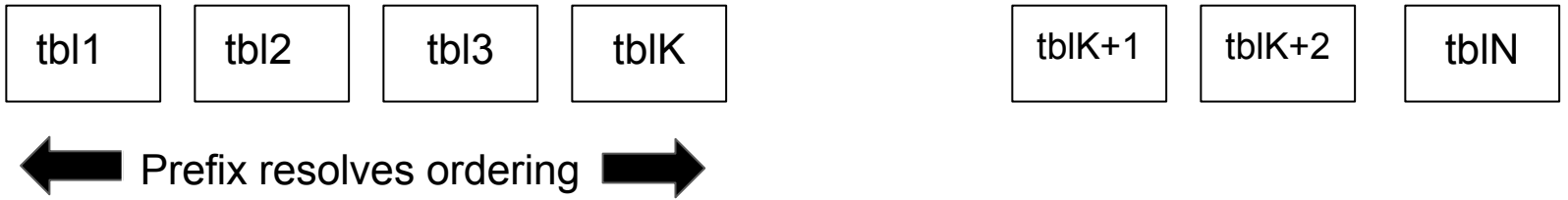
COST BASED OPTIMIZATION

Motivation

- Come up with a cost based optimization that would consider
  - Pushing the LIMIT down to a partial join
  - Cost of sorting
- Shortcut the join execution

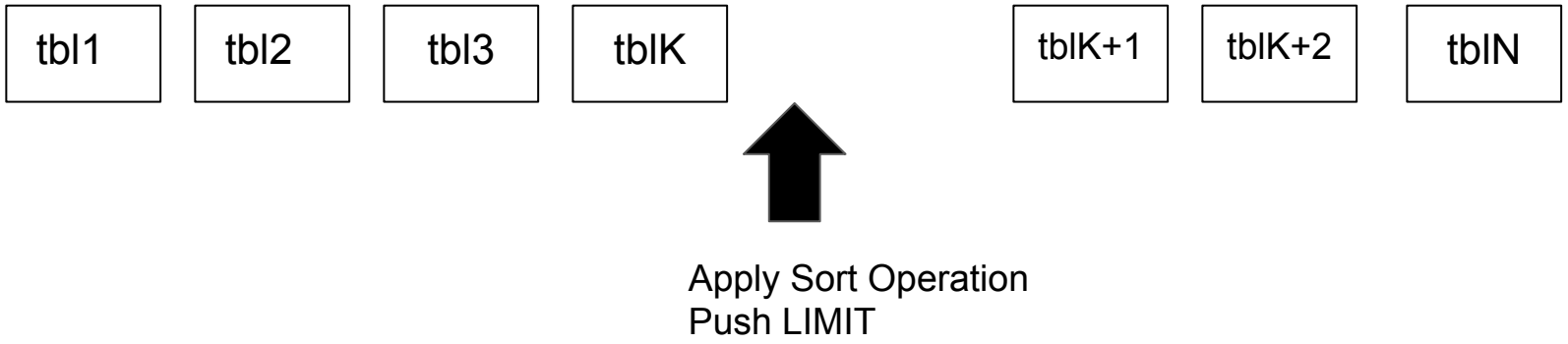
Pushing the LIMIT

Pushing the limit to a partial join means **reading only a fraction of records** of the join prefix that are **sorted in accordance with the ORDER BY clause**.



Pushing the LIMIT

Pushing the limit to a partial join means **reading only a fraction of records** of the join prefix that are **sorted in accordance with the ORDER BY clause**.



Pushing the LIMIT

Pushing the limit to a partial join means **reading only a fraction of records** of the join prefix that are **sorted in accordance with the ORDER BY clause**.

The fraction of records read would be:

```
records= LIMIT * (cardinality(t1,t2....tk) / cardinality(t1,t2....tn))
```

JOIN OPTIMIZATION

- Get an estimate of the join cardinality by running the join planner
- Access methods that ensure pre-existing ordering are also taken into account inside the join planner.

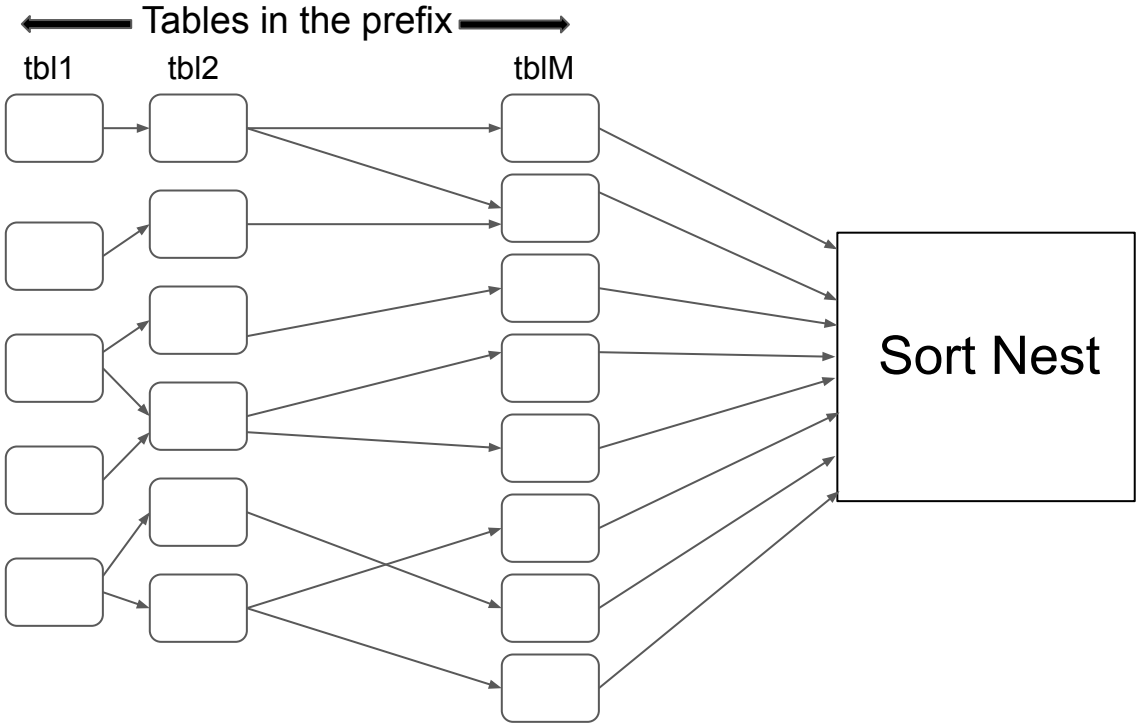
JOIN OPTIMIZATION

- For each partial join prefix that can resolve the ORDER BY clause the prefix is extended with two options:
  - Insert the sort operation immediately and push LIMIT
  - Extend the partial join prefix and add sort operation later
- Equalities are propagated from the WHERE clause so that all join prefixes which can resolve the ordering are taken into account.
  - Example if the ORDER BY clause is t1.a and there is an equality defined t1.a=t3.a
    - Join prefix t2, t3 => limit will be pushed
    - Join prefix t2, t1 => limit will be pushed

JOIN EXECUTION

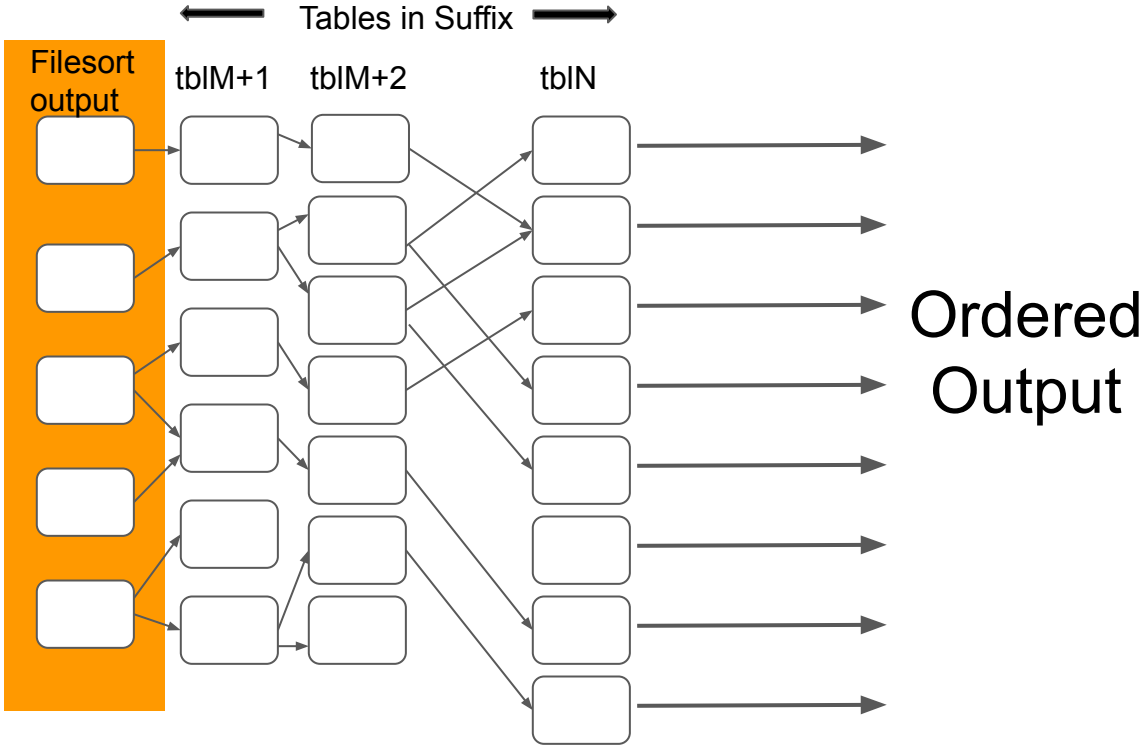
- Materialize the prefix that resolves the ORDER BY clause
- Sort the materialized nest in accordance with the ORDER BY clause
- Read records from the the result of sorting one by one and join with the tables in the suffix with NESTED LOOP JOIN.
- The execution stops as soon as we get LIMIT records in the output.

Execution path using a sort nest



- A materialized nest is a nest whose tables are joined together and result is put inside a temporary table.
- Sort nest is a materialized nest which can be sorted.
- After the sort-nest is filled, this table is passed to filesort()
- Join buffering is allowed for the tables in the prefix
- Conditions that depend only on the tables of the prefix are checked before sorting

Execution path using a sort nest



- Cannot use join buffering after the sort nest is formed
- As soon as the LIMIT records are found the join execution stops

MariaDB

EXAMPLES

```
SELECT * FROM customer, orders, lineitem, nation
WHERE c_custkey = o_custkey AND
            l_orderkey = o_orderkey AND
           o_orderdate >= '1993-10-01' AND
            o_orderdate <  '1994-01-01' AND
           l_returnflag = 'R' AND c_nationkey = n_nationkey
ORDER BY c_acctbal, n_name
LIMIT 10;
```

```
+------------+------+--------------+---------+-------------------------+------+---------+----------------+
| table      | type | key          | key_len | ref                     | rows | r_rows  | Extra          |
+------------+------+--------------+---------+-------------------------+------+---------+----------------+
| nation     | ALL  | NULL         | NULL    | NULL                    | 25   | 25.00   |                |
| customer   | ref  | i_c_nationkey | 5       | dbt3.nation.n_nationkey | 6000 | 6000.00 |                |
| <sort-nest> | ALL  | NULL         | NULL    | NULL                    | 14   | 19.00   | Using filesort |
| orders     | ref  | i_o_custkey  | 5       | sort-nest.c_custkey     | 15   | 8.32    | Using where    |
| lineitem   | ref  | PRIMARY      | 4       | dbt3.orders.o_orderkey  | 4    | 2.67    | Using where    |
+------------+------+--------------+---------+-------------------------+------+---------+----------------+
```

MariaDB

EXAMPLES

```
SELECT * FROM customer, orders, lineitem, nation
WHERE c_custkey = o_custkey AND
            l_orderkey = o_orderkey AND
           o_orderdate >= '1993-10-01' AND
            o_orderdate <  '1994-01-01' AND
           l_returnflag = 'R' AND c_nationkey = n_nationkey
ORDER BY c_acctbal, n_name
LIMIT 10;
```

```
+------------+------+--------------+---------+-------------------------+------+---------+----------------+
| table      | type | key          | key_len | ref                     | rows | r_rows  | Extra          |
+------------+------+--------------+---------+-------------------------+------+---------+----------------+
| nation     | ALL  | NULL         | NULL    | NULL                    | 25   | 25.00   |                |
| customer   | ref  | i_c_nationkey| 5       | dbt3.nation.n_nationkey | 6000 | 6000.00 |                |
| <sort-nest>| ALL  | NULL         | NULL    | NULL                    | 14   | 19.00   | Using filesort |
| orders     | ref  | i_o_custkey  | 5       | sort-nest.c_custkey     | 15   | 8.32    | Using where    |
| lineitem   | ref  | PRIMARY      | 4       | dbt3.orders.o_orderkey  | 4    | 2.67    | Using where    |
+------------+------+--------------+---------+-------------------------+------+---------+----------------+
```

MariaDB

EXAMPLES

```
SELECT * FROM
t_fact
  JOIN dim1
    ON t_fact.dim1_id= dim1.dim1_id
ORDER BY t_fact.col1
LIMIT 1000;
```

```
MariaDB [test]> SHOW CREATE TABLE t_fact \G
*************************** 1. row ***************
       Table: t_fact
Create Table: CREATE TABLE `t_fact` (
  `fact_id` int(11) NOT NULL,
  `dim1_id` int(11) NOT NULL,
  `dim2_id` int(11) NOT NULL,
  `col1` int(11) NOT NULL,
  PRIMARY KEY (`fact_id`),
  KEY `dim1_id` (`dim1_id`),
  KEY `dim2_id` (`dim2_id`),
  KEY `col1` (`col1`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

MariaDB

EXAMPLES

```
SELECT * FROM
t_fact
  JOIN dim1
    ON t_fact.dim1_id= dim1.dim1_id
ORDER BY t_fact.col1
LIMIT 1000;
```

EXECUTION TIME
0.013 sec

Speedup
1900x

```
+------+-------------+--------+--------+---------+---------+----------------------+------+-------+
| id   | select_type | table  | type   | key     | key_len | ref                  | rows | Extra |
+------+-------------+--------+--------+---------+---------+----------------------+------+-------+
|    1 | SIMPLE      | t_fact | index  | col1    | 4       | NULL                 | 1900 |       |
|    1 | SIMPLE      | dim1   | eq_ref | PRIMARY | 4       | test.t_fact.dim1_id | 1    |       |
+------+-------------+--------+--------+---------+---------+----------------------+------+-------+
```

MariaDB

EXAMPLES

```
SELECT *
FROM customer, nation
WHERE c_nationkey=n_nationkey AND
  n_name in ('USA','Germany','FRANCE','Belgium')
ORDER BY c_acctbal
LIMIT 10;
```

```
*************************** 1. row ***************
       Table: customer
Create Table: CREATE TABLE `customer` (
  `c_custkey` int(11) NOT NULL,
  `c_name` varchar(25) DEFAULT NULL,
  `c_address` varchar(40) DEFAULT NULL,
  `c_nationkey` int(11) DEFAULT NULL,
  `c_phone` char(15) DEFAULT NULL,
  `c_acctbal` double DEFAULT NULL,
  `c_mktsegment` char(10) DEFAULT NULL,
  `c_comment` varchar(117) DEFAULT NULL,
  PRIMARY KEY (`c_custkey`),
  KEY `i_c_nationkey` (`c_nationkey`),
  KEY `c_acctbal` (`c_acctbal`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
1 row in set (0.001 sec)
```

MariaDB

EXAMPLES

```
SELECT *
FROM customer, nation
WHERE c_nationkey=n_nationkey AND
   n_name in ('USA','Germany','FRANCE','Belgium')
ORDER BY c_acctbal
LIMIT 10;
```

EXECUTION TIME
0.002 sec

Speedup
43x

```
+------+-------------+----------+--------+------------+---------+---------------------------+------+-------------+
| id   | select_type | table    | type   | key        | key_len | ref                       | rows | Extra       |
+------+-------------+----------+--------+------------+---------+---------------------------+------+-------------+
|    1 | SIMPLE      | customer | index  | c_acctbal  | 9       | NULL                      | 31   | Using where |
|    1 | SIMPLE      | nation   | eq_ref | PRIMARY    | 4       | dbt3.customer.c_nationkey | 1    | Using where |
+------+-------------+----------+--------+------------+---------+---------------------------+------+-------------+
```

MariaDB

Limitations

- Depends heavily on the SELECTIVITY of the conditions
  - Use histograms to provide selectivities
  - Few predicates selectivity is unknown
    - Example: t1.a < t2.b
- Estimate of join cardinality are very pessimistic.

THANK YOU!