PERCONA

TRAINING

Query Optimization 101

---

https://www.percona.com/training

Table of Contents

© 2022 Percona | Confidential

Quick Intro - Percona

- Percona is a leading provider of unbiased open source database solutions
  - Eliminate vendor lock-in
  - Embrace the cloud
  - Optimize Database Performance
  - Reduce Costs and Complexity
- Fully Open-Source Software (MySQL, PGSQL, MongoDB, PMM, etc)
  - It's all on github
  - It's free, as in beer!
- Professional Services (Consulting, Performance Audits, etc)
- Managed Database Services (We do everything for you)
- Support (Traditional ticket-based support, various SLA tiers)

© 2022 Percona | Confidential

Quick Intro - Me

- I'm Matthew, not Matt :)
- Been with Percona for 9.67 years
- Senior Architect / Senior Trainer in Professional Services dept
- Specialize in MySQL and training
- Professional Experience:
  - PayPal - 5,000+ MySQL servers (heavily modulo sharded)
  - Dell Secureworks - SaaS-based application backed by MySQL
  - DBaaS Hosting (Rackspace, Softlayer)
  - VoIP (creator of MySQL DB driver for Asterisk)

© 2022 Percona | Confidential

Query Planning

Query Optimization

PERCONA
TRAINING

What Will You Do?

- The number one goal is to have faster queries.
- The process is:
  - We first ask MySQL "what do you intend to do?"
  - If we don't like it, make a change, and ask again...

© 2022 Percona | Confidential

It All Starts with EXPLAIN

- Bookmark this manual page:
  - http://dev.mysql.com/doc/refman/8.0/en/explain-output.html
- It is the best source for anyone getting started.

© 2022 Percona | Confidential

Find the Title Bambi

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE title = 'Bambi' ORDER BY production_year\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 3331824
     filtered: 10.00
        Extra: Using where; Using filesort
1 row in set, 1 warning (0.00 sec)
```


© 2022 Percona | Confidential

Aha! Now Add an Index

```
mysql> ALTER TABLE movies ADD INDEX (title);
```

```
ERROR 1170 (42000): BLOB/TEXT column 'title' used in key
specification without a key length
```

```
mysql> ALTER TABLE movies ADD INDEX (title(30));
Query OK, 0 rows affected (8.09 sec)
Records: 0 Duplicates: 0 Warnings: 0
```


© 2022 Percona | Confidential

Let's Revisit

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE title = 'Bambi' ORDER by production_year\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: ref
possible_keys: title
          key: title
      key_len: 52
          ref: const
         rows: 4
     filtered: 100.00
        Extra: Using where; Using filesort
1 row in set, 1 warning (0.00 sec)
```

- `ref` is equality for comparison, but not PK lookup.
- Identified 'title' as a candidate index and chose it.
- Size of the index used.
- Anticipated number of rows.

© 2022 Percona | Confidential

Other Ways of Accessing

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE id = 55327\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: const
possible_keys: PRIMARY
          key: PRIMARY
      key_len: 4
          ref: const
         rows: 1
     filtered: 100.00
        Extra: NULL
1 row in set, 1 warning (0.00 sec)
```

- const: at most, one matching row.
- Primary Key in InnoDB is always faster than secondary keys.

© 2022 Percona | Confidential

LIKE

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE title LIKE 'Bamb%'\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: range
possible_keys: title
          key: title
      key_len: 52
          ref: NULL
         rows: 176
     filtered: 100.00
        Extra: Using where
1 row in set, 1 warning (0.00 sec)
```

- Type is Range. BETWEEN, IN() and < > are also ranges.
- Number of rows to examine has increased; we are not specific enough.

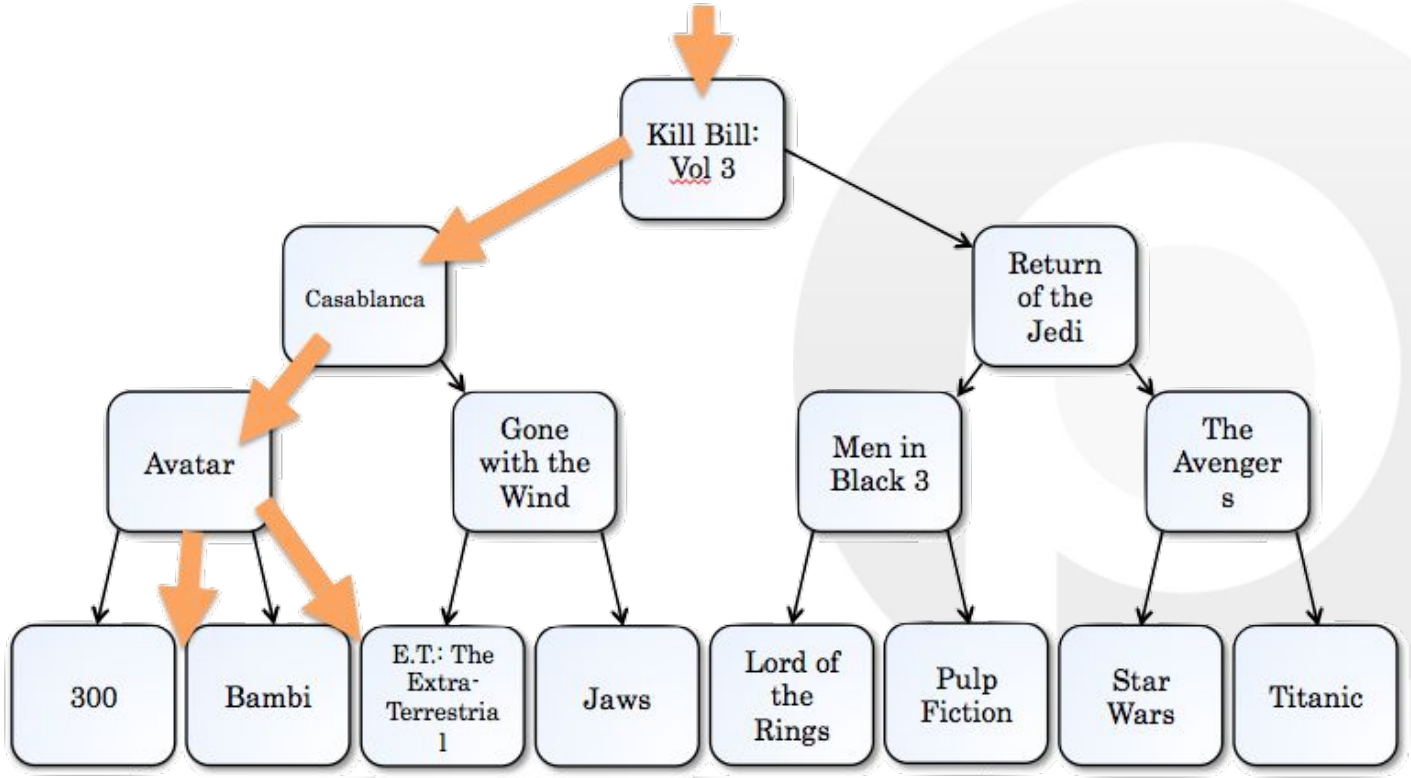
© 2022 Percona | Confidential

Why is that a Range?

- We're looking for titles between BambA and BambZ*
- Going to use an index; made up using a tree-structure
    - InnoDB uses B+Trees for data and indexes

© 2022 Percona | Confidential

A Tree? What's That?



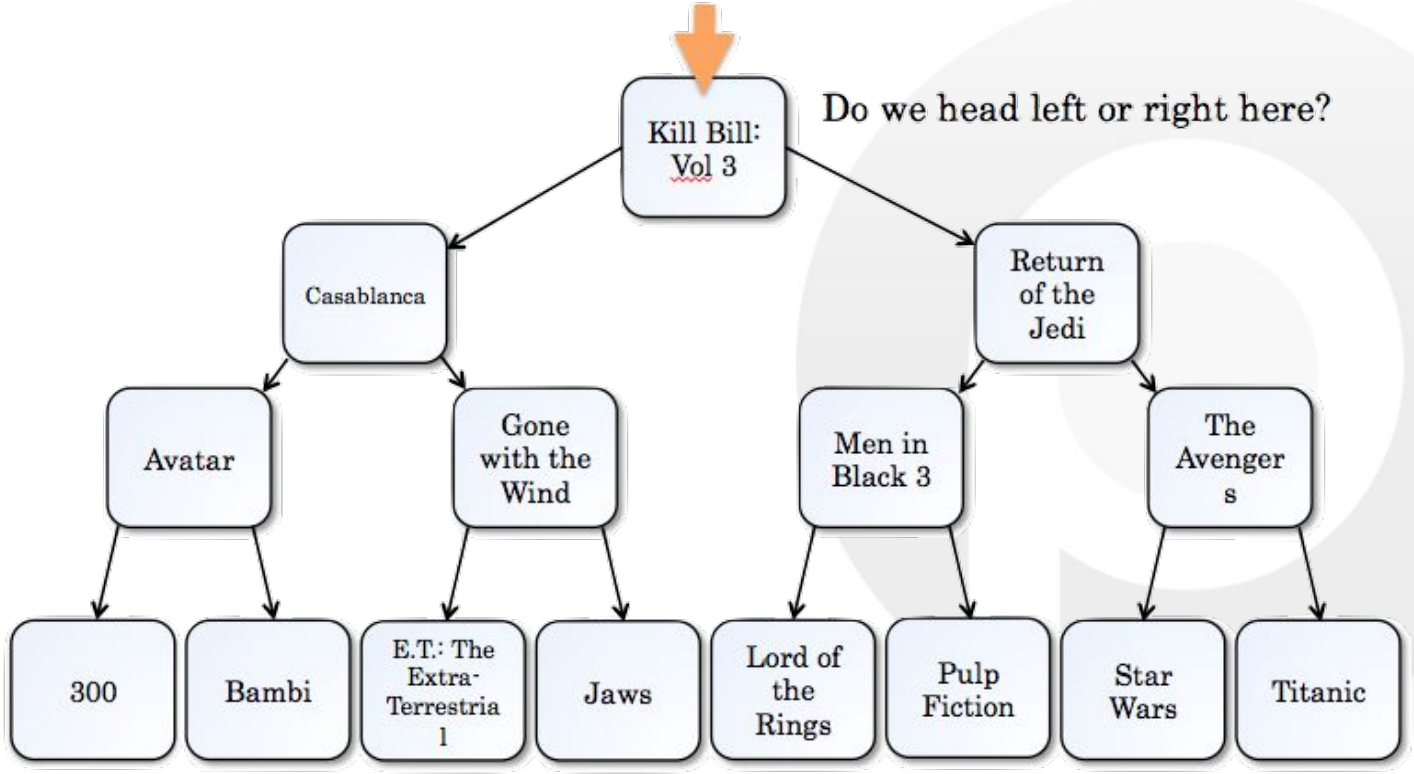
© 2022 Percona | Confidential

Could This Be a Range?

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE title LIKE '%ulp Fiction'\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 3331824
     filtered: 11.11
        Extra: Using where
1 row in set (0.00 sec)
```


© 2022 Percona | Confidential

No, We Can't Traverse



© 2022 Percona | Confidential

LIKE 'Z%'

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE movies LIKE 'Z%'\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: range
possible_keys: title
          key: title
      key_len: 52
          ref: NULL
         rows: 24934
     filtered: 100.00
        Extra: Using where
1 row in set, 1 warning (0.00 sec)
```

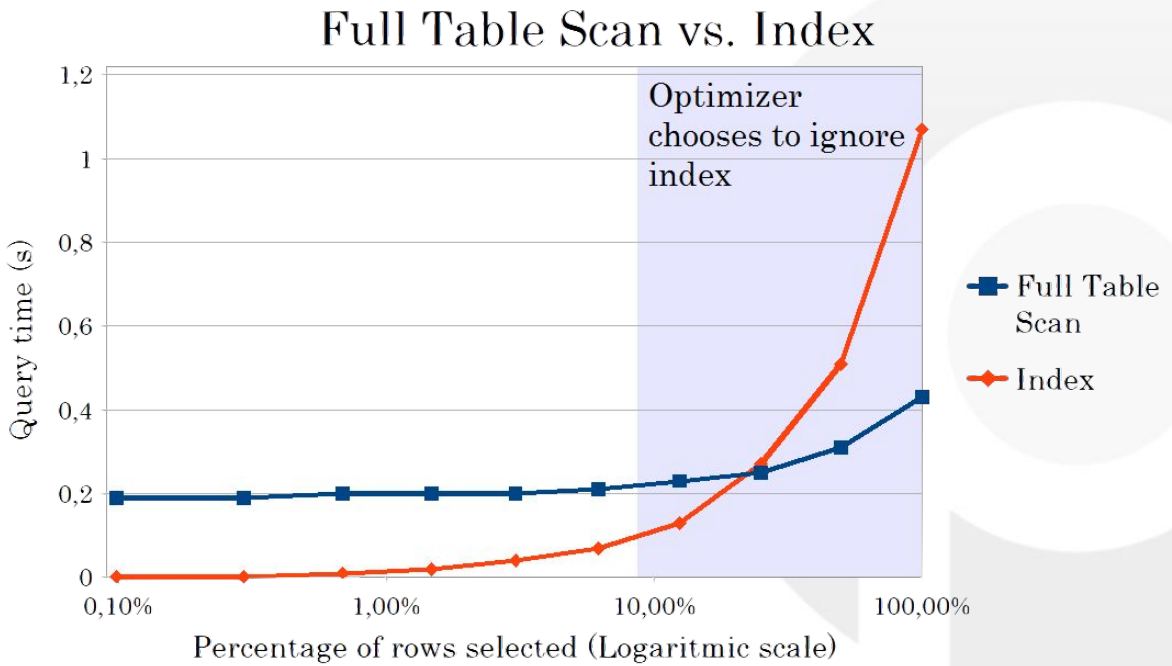

© 2022 Percona | Confidential

LIKE 'T%'

```
mysql> EXPLAIN SELECT id, title, production_year FROM movies
       WHERE movies LIKE 'T%'\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: ALL
possible_keys: title
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 3331824
     filtered: 21.00
        Extra: Using where
1 row in set, 1 warning (0.00 sec)
```


© 2022 Percona | Confidential

Why Avoid Indexes



© 2022 Percona | Confidential

What You Should Take Away

- Data is absolutely critical.
  - Dev environments should contain data exported from production systems.
  - Too few rows may cause optimizer to behave differently

© 2022 Percona | Confidential

What You Should Take Away (cont.)

- Input values are absolutely critical.
  - Between two seemingly identical queries, execution plans may be very different.
  - Just like you test application code functions with several values for input arguments.

© 2022 Percona | Confidential

Common Query Mistakes

Query Optimization

PERCONA
TRAINING

Not So Obvious?

```
mysql> CREATE TABLE orders (
  id int(10) unsigned NOT NULL AUTO_INCREMENT,
  orderId varchar(20) NOT NULL,
  orderType tinyint(3) unsigned NOT NULL,
  orderCountry varchar(40) NOT NULL,
  PRIMARY KEY (id),
  KEY oIdx (orderId));   <-- Index exists!

mysql> EXPLAIN SELECT * FROM orders WHERE orderId = 34239\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: orders
         type: ALL
possible_keys: oIdx
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 6316
     filtered: 10.00
        Extra: Using where
1 row in set, 3 warnings (0.00 sec)
```


© 2022 Percona | Confidential

Type Matters!

- Check those warnings and try again

```
mysql> SHOW WARNINGS\G
...
Message: Cannot use ref access on index 'oIdx' due to type or
         collation conversion on field 'orderId'
Message: Cannot use range access on index 'oIdx' due to type or
         collation conversion on field 'orderId'

mysql> EXPLAIN SELECT * FROM orders WHERE orderId = "34239"\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: orders
         type: ref
possible_keys: oIdx
          key: oIdx
      key_len: 22
          ref: const
         rows: 1
     filtered: 100.00
        Extra: NULL
```


© 2022 Percona | Confidential

Please Use My Index?

```
mysql> ALTER TABLE movies ADD INDEX (production_year);
mysql> EXPLAIN SELECT * FROM movies
       WHERE MAKEDATE(production_year, 1) >= now() - INTERVAL 1 YEAR\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 3244766
     filtered: 100.00
        Extra: Using where
1 row in set, 1 warning (0.00 sec)
```

- An index exists on *production_year*. What's going on?
- What is a better solution?

© 2022 Percona | Confidential

Much Better

```
mysql> EXPLAIN SELECT * from movies
       WHERE production_year >= YEAR(NOW() - INTERVAL 1 YEAR)\G
************* 1. row *************
           id: 1
  select_type: SIMPLE
        table: movies
   partitions: NULL
         type: range
possible_keys: production_year
          key: production_year
      key_len: 5
          ref: NULL
         rows: 205352
     filtered: 100.00
        Extra: Using index condition
1 row in set, 1 warning (0.00 sec)
```

- *Rule of Thumb*: Don't manipulate data already stored

© 2022 Percona | Confidential

Finding Unoptimized Queries

Query Optimization

PERCONA
TRAINING

How do I find slow queries?

- Slow Query Log
  - Use `pt-query-digest` to parse and aggregate statistics
- *performance_schema*
  - Along with *sys* schema
- Percona Monitoring and Management (PMM)
  - Can use either slow query log or *performance_schema*

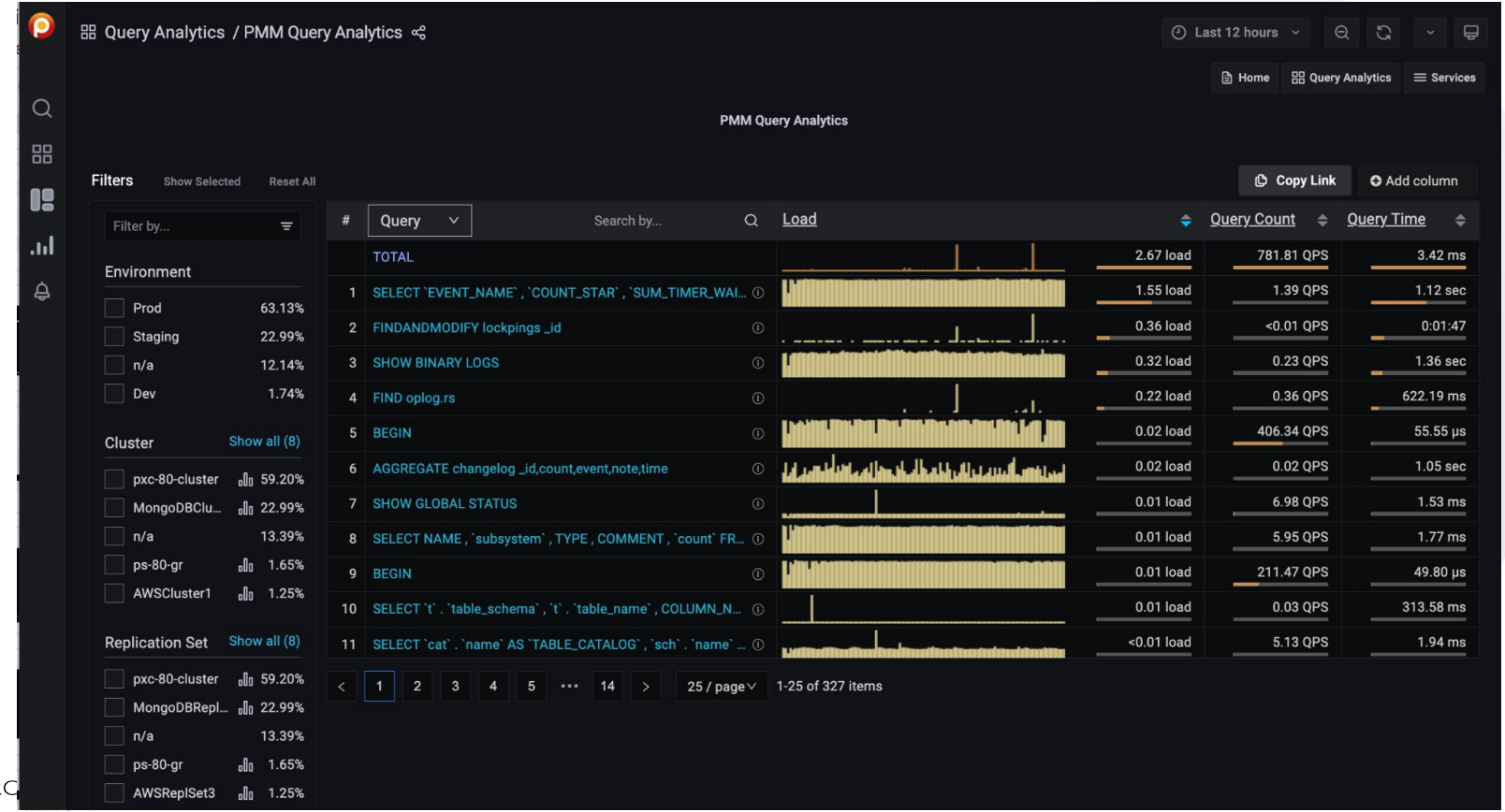
© 2022 Percona | Confidential

Enhanced Slow Log Statistics

```
mysql> SET GLOBAL slow_query_log = ON;
mysql> SET GLOBAL slow_query_log_file = '/var/lib/mysql/slow.log';
mysql> SET GLOBAL long_query_time = 0;
mysql> SET GLOBAL log_slow_verbosity = full;

mysql> SET GLOBAL log_slow_rate_type = [session|query]
mysql> SET GLOBAL log_slow_rate_limit = [1-1000];
       # Depends on above parameter; 100 = 1% for 'query'

mysql> SET GLOBAL log_slow_sp_statements = TRUE;

mysql> SET GLOBAL log_slow_filter = [full_scan|full_join|tmp_table_on_disk|filesort];
```


© 2022 Percona | Confidential

PMM Query Analytics



Other `EXPLAIN` Techniques

Query Optimization

PERCONA
TRAINING

EXPLAIN FORMAT=JSON

```
mysql> EXPLAIN FORMAT=JSON SELECT * FROM users
       WHERE last_name = 'Ascencio' AND first_name = 'Virginia'\G
```

```
****************** 1. row ******************
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "cost_info": {
      "query_cost": "171.60"
    },
    "table": {
      "table_name": "users",
      "access_type": "ref",
      "possible_keys": [
        "first_name"
      ],
      "key": "first_name",
      "used_key_parts": [
        "first_name"
      ],
      "key_length": "102",
      "ref": [
        "const"
      ],
      "rows_examined_per_scan": 143,
      "rows_produced_per_join": 14,
      "filtered": "10.00",
      "cost_info": {
        "read_cost": "143.00",
        "eval_cost": "2.86",
        "prefix_cost": "171.60",
        "data_read_per_join": "6K"
      },
      "used_columns": [
        "id",
        "email_address",
        "last_login_date",
        "first_name",
        "last_name"
      ],
      "attached_condition":
        "(`users`.`last_name` = 'Ascencio')"
    }
  }
}
1 row in set, 1 warning (0.00 sec)
```


© 2022 Percona | Confidential

EXPLAIN FORMAT=TREE

- EXPLAIN FORMAT=TREE provides tree like output about the selected execution plan

```
EXPLAIN FORMAT=TREE SELECT * FROM users WHERE last_name = 'Ascencio' AND first_name = 'Virginia'\G
*************************** 1. row ***************************
EXPLAIN: -> Filter: ((users.first_name = 'Virginia') and (users.last_name = 'Ascencio'))  (cost=5068.85
rows=500)
    -> Table scan on users  (cost=5068.85 rows=49966)
```

```
EXPLAIN FORMAT=TREE SELECT id FROM title WHERE (title ='Pilot' and production_year > 1977)
 UNION SELECT id FROM title WHERE (episode_nr = 1 and production_year > 1977) \G
*************************** 1. row ***************************
EXPLAIN: -> Table scan on <union temporary>  (cost=2.50 rows=0)
    -> Union materialize with deduplication
        -> Filter: ((title.title = 'Pilot') and (title.production_year > 1977))  (cost=161267.64 rows=51606)
            -> Table scan on title  (cost=161267.64 rows=1548344)
        -> Filter: ((title.episode_nr = 1) and (title.production_year > 1977))  (cost=161267.64 rows=51606)
            -> Table scan on title  (cost=161267.64 rows=1548344)
```


© 2022 Percona | Confidential

EXPLAIN ANALYZE

- EXPLAIN ANALYZE: Shows the EXPLAIN plan (8.0.18)
  - Also executes the query and displays execution time and number of row reads

```
EXPLAIN ANALYZE SELECT * FROM users WHERE last_name = 'Ascencio' AND first_name = 'Virginia'\G
*************************** 1. row ***************************
EXPLAIN: -> Filter: ((users.first_name = 'Virginia') and (users.last_name = 'Ascencio'))  (cost=5068.85 rows=500)
(actual time=5.771..32.774 rows=1 loops=1)
    -> Table scan on users  (cost=5068.85 rows=49966) (actual time=0.054..29.703 rows=49930 loops=1)
```

```
EXPLAIN analyze SELECT id FROM title WHERE (title ='Pilot' and production_year > 1977) UNION SELECT id FROM title
WHERE (episode_nr = 1 and production_year > 1977) \G
*************************** 1. row ***************************
EXPLAIN: -> Table scan on <union temporary>  (cost=2.50 rows=0) (actual time=0.001..0.716 rows=16185 loops=1)
    -> Union materialize with deduplication  (actual time=19669.891..19671.602 rows=16185 loops=1)
        -> Filter: ((title.title = 'Pilot') and (title.production_year > 1977))  (cost=161267.64 rows=51606) (actual
time=8625.515..19215.996 rows=730 loops=1)
            -> Table scan on title  (cost=161267.64 rows=1548344) (actual time=0.023..19061.133 rows=1543719 loops=1)
        -> Filter: ((title.episode_nr = 1) and (title.production_year > 1977))  (cost=161267.64 rows=51606) (actual
time=183.589..446.108 rows=16079 loops=1)
            -> Table scan on title  (cost=161267.64 rows=1548344) (actual time=0.029..361.534 rows=1543719 loops=1)
```


© 2022 Percona | Confidential

Beyond Explain

Query Optimization

PERCONA
TRAINING

The Limitations of EXPLAIN

- EXPLAIN shows MySQL's intentions; there is no post-execution analysis.
  - How many rows actually had to be sorted?
  - Was that temporary table created on disk?
  - Did the LIMIT 10 result in a quick match, resulting in fewer rows scanned?
  - ... we don't know.

© 2022 Percona | Confidential

What Did the Engine Do?

```
mysql> SHOW STATUS LIKE 'Ha%';
+----------------------------+----------+
| Variable_name              | Value    |
+----------------------------+----------+
| Handler_commit             | 0        |
| Handler_delete             | 0        |
| Handler_discover           | 0        |
| Handler_prepare            | 0        |
| Handler_read_first         | 1        | <-- First entry in index
| Handler_read_key           | 13890229 | <-- Read a row based on key
| Handler_read_next          | 14286456 | <-- Read the next row
| Handler_read_prev          | 0        |  -- in key order
| Handler_read_rnd           | 0        |
| Handler_read_rnd_next      | 2407004  | <-- Read the next row
| Handler_rollback           | 0        |  -- from disk
| Handler_savepoint          | 0        |
| Handler_savepoint_rollback | 0        |
| Handler_update             | 0        |
| Handler_write              | 2407001  | <-- Insert row to table
+----------------------------+----------+
```


© 2022 Percona | Confidential

SHOW PROFILES

- Enable profiling:

```
mysql> SET profiling = 1;
```

- Run some query(s):

```
mysql> SELECT STRAIGHT_JOIN COUNT(*) AS c, person_id
       FROM cast_info FORCE INDEX(person_id)
       INNER JOIN title ON (cast_info.movie_id=title.id)
       WHERE title.kind_id = 1 GROUP BY cast_info.person_id
       ORDER by c DESC LIMIT 1;
```

- View the report:

```
mysql> SHOW PROFILES;
| Query ID | Duration     | Query
|        1 | 211.21064300 | SELECT STRAIGHT_JOIN ...
```


© 2022 Percona | Confidential

SHOW PROFILES (cont.)

```
mysql> show profile for query 1;
+--------------------------+----------+
| Status                   | Duration |
+--------------------------+----------+
| starting                 |  0.000079 |
| checking permissions     |  0.000005 |
| checking permissions     |  0.000006 |
| Opening tables           |  0.000025 |
| init                     |  0.000034 |
| System lock              |  0.000013 |
| optimizing               |  0.000018 |
| statistics               |  0.000095 |
| preparing                |  0.000029 |
| Creating tmp table       |  0.000022 |
| Sorting for group        |  0.000012 |
| Sorting result           |  0.000004 |
| executing                |  0.000004 |
...
```

```
...
| Sending data             | 28.653018 |
| converting HEAP to MyISAM |  0.100713 |
| Sending data             | 17.346021 |
| Creating sort index      |  0.148820 |
| end                      |  0.000007 |
| removing tmp table       |  0.007086 |
| end                      |  0.000009 |
| query end                |  0.000013 |
| closing tables           |  0.000022 |
| freeing items            |  0.000263 |
| logging slow query       |  0.000007 |
| logging slow query       |  0.000005 |
| cleaning up              |  0.000052 |
+--------------------------+-----------+
26 rows in set (0.00 sec)
```


© 2022 Percona | Confidential

PERCONA

TRAINING

Questions?

https://forums.percona.com/

---

https://www.percona.com/training