# Lab 3 Part 4 Report

## Explain

The output of Explain has 12 columns: id, select_type, table, partitions, type, possible_keys, key, key_len, ref, rows, filtered, and Extra. These columns describe how the output of the query is produced.

The **id** describes the SELECT identifier.

The **select_type** describes the SELECT type.

The **table** describes the table for the output rows.

The **partitions** describes the matching partitions.

The **type** describes how it's joined.

The **possible_keys** describes what indexes can be chosen.

The **key** describes which index is actually chosen.

The **key_len** describes the length of this chosen key.

The **ref** describes the columns compared to the index.

The **rows** describes an estimate of the number of rows that will be examined.

The **filtered** describes the percentage of rows that are filtered.

The **Extra** describes any other additional information, such as describing how it's joined.

# Part 4-1

## Before

```
+----+-------------+--------+------------+------+---------------+------+---------+------+---------+----------+-------------+
| id | select_type | table  | partitions | type | possible_keys | key  | key_len | ref  | rows    | filtered | Extra       |
+----+-------------+--------+------------+------+---------------+------+---------+------+---------+----------+-------------+
|  1 | SIMPLE      | Review | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 1120345 |   100.00 | Using where |
+----+-------------+--------+------------+------+---------------+------+---------+------+---------+----------+-------------+
```

| Run | Times (s) | Average | Standard Deviation |
|---|---|---|---|
| 1 | 0.76 | 0.77 | 0.006519202 |
| 2 | 0.773 | | |
| 3 | 0.776 | | |
| 4 | 0.774 | | |
| 5 | 0.767 | | |

## After

```
+----+-------------+--------+------------+-------+---------------+-----------------+---------+------+---------+----------+--------------------------+
| id | select_type | table  | partitions | type  | possible_keys | key             | key_len | ref  | rows    | filtered | Extra                    |
+----+-------------+--------+------------+-------+---------------+-----------------+---------+------+---------+----------+--------------------------+
|  1 | SIMPLE      | Review | NULL       | index | NULL          | review_date_idx | 6       | NULL | 1120345 |   100.00 | Using where; Using index |
+----+-------------+--------+------------+-------+---------------+-----------------+---------+------+---------+----------+--------------------------+
```

| Run | Times (s) | Average | Standard Deviation |
|---|---|---|---|
| 1 | 0.242 | 0.2466 | 0.005683309 |
| 2 | 0.251 | | |
| 3 | 0.25 | | |
| 4 | 0.251 | | |
| 5 | 0.239 | | |

## Improvement

After creating an index in the Review table for the date column, we see the average execution time drop from 0.77 seconds to 0.2466 seconds – an execution time reduction of 68%.

## Alternatives

In this case, the only logical choice of column to index on is the date column, since we are simply returning the number of reviews written in a specific month. For this specific query, no other choice of indexing makes sense.

# Part 4-2

## Before

```
+----+-------------+-------+------------+------+---------------+------+---------+------+---------+----------+---------------------------------------------+
| id | select_type | table | partitions | type | possible_keys | key  | key_len | ref  | rows    | filtered | Extra                                       |
+----+-------------+-------+------------+------+---------------+------+---------+------+---------+----------+---------------------------------------------+
|  1 | SIMPLE      | ud    | NULL       | ALL  | NULL          | NULL | NULL    | NULL |  765240 |    10.00 | Using where                                 |
|  1 | SIMPLE      | r     | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 1120345 |    10.00 | Using where; Using join buffer (hash join) |
|  1 | SIMPLE      | b     | NULL       | ALL  | NULL          | NULL | NULL    | NULL |  183269 |   100.00 | Using where; Using join buffer (hash join) |
+----+-------------+-------+------------+------+---------------+------+---------+------+---------+----------+---------------------------------------------+
```

| Run | Times (s) | Average | Standard Deviation |
|---|---|---|---|
| 1 | 1.416 | 1.4274 | 0.006730527 |
| 2 | 1.43 | | |
| 3 | 1.427 | | |
| 4 | 1.431 | | |
| 5 | 1.433 | | |

## After

```
+----+-------------+-------+------------+------+---------------+---------------+---------+-------+--------+----------+---------------------------------------------+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows   | filtered | Extra                                       |
+----+-------------+-------+------------+------+---------------+---------------+---------+-------+--------+----------+---------------------------------------------+
|  1 | SIMPLE      | r     | NULL       | ref  | review_user_id | review_user_id | 90      | const |     13 |   100.00 | NULL                                        |
|  1 | SIMPLE      | ud    | NULL       | ALL  | NULL          | NULL          | NULL    | NULL  | 765240 |    10.00 | Using where; Using join buffer (hash join) |
|  1 | SIMPLE      | b     | NULL       | ALL  | NULL          | NULL          | NULL    | NULL  | 183269 |   100.00 | Using where; Using join buffer (hash join) |
+----+-------------+-------+------------+------+---------------+---------------+---------+-------+--------+----------+---------------------------------------------+
```

| Run | Times (s) | Average | Standard Deviation |
|---|---|---|---|
| 1 | 0.59 | 0.5982 | 0.00626099 |
| 2 | 0.6 | | |
| 3 | 0.606 | | |
| 4 | 0.601 | | |
| 5 | 0.594 | | |

## Improvement

After creating an index in the Review table for the user_id columns, we see the average execution time drop from 1.4274 seconds to 0.5982 seconds – an execution time reduction of 58%.

## Alternatives

Another option would be to have a single index on business_id in the Review table to be used join the Business table and the Review table on business_id more efficiently.