

# ASSIGNMENT - 8

1. We optimized the 'UNION' clause query using 'OR' (as shown in Query\_ID 16 & 17).

```
mysql> show profiles;
+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+
| 3 | 0.00139350 | (select * from passenger where pfirst_name like '%ak%') union (select * from passenger where plast_name like '%jd%') |
| 4 | 0.00187150 | select * from passenger where pfirst_name like '%ak%' or plast_name like '%jd%' |
| 5 | 0.00257825 | (select * from passenger where pfirst_name like '%ak%') union (select * from passenger where plast_name like '%jd%') |
| 6 | 0.00190925 | select * from passenger where pfirst_name like '%ak%' or plast_name like '%jd%' |
| 7 | 0.01154900 | select count(*) from passenger |
| 8 | 0.00213875 | select count(*) from passenger |
| 9 | 0.00188550 | select count(*) from passenger |
| 10 | 0.00641600 | select count(*) from passenger |
| 11 | 0.00267425 | select count(*) from passenger |
| 12 | 0.00611850 | select count(*) from passenger |
| 13 | 0.00260625 | select count(*) from passenger |
| 14 | 0.00241125 | select count(*) from passenger |
| 15 | 0.00251875 | select count(*) from passenger |
| 16 | 0.00192200 | select * from passenger where pfirst_name like '%ak%' or plast_name like '%jd%' |
| 17 | 0.00549150 | (select * from passenger where pfirst_name like '%ak%') union (select * from passenger where plast_name like '%jd%') |
+-----+-----+-----+
15 rows in set, 1 warning (0.00 sec)
```

2. As we can observe the number of scans before optimizing were 2066 and after optimizing(applied fulltext index on address column) the number of scans comes out to be 1.

```
mysql> show profiles;
```

Query_ID	Duration	Query
58	0.07137975	drop index idx_airline_airid on airline
59	0.00663550	explain select city_name from city where state = (select a_state from airport where a_state = 'Jjrqxnhh')
60	0.00653025	explain select city.city_name from city natural join airport where airport.a_state = 'Jjrqxnhh'
61	0.00298125	explain select city.city_name from city natural join airport natural join airport_consists_of natural join airline where airport.a_state = 'Jjrqxnhh'
62	0.02123725	show index from passenger
63	0.02569750	select * from passenger
64	0.00316850	select * from passenger where p_address = '3215 Madsen Street, Hayward, CA, 94541'
65	0.00023175	select * from passenger match (p_address) against ('3215 Madsen Street, Hayward, CA, 94541')
66	0.00016800	select * from passenger match (p_address) against ('3215 Madsen Street, Hayward, CA, 94541' in natural language mode)
67	0.05692500	select * from passenger where match(p_address) against ('3215 Madsen Street, Hayward, CA, 94541' in natural language mode)
68	0.00774250	select * from passenger where match(p_address) against ('3215 Madsen Street, Hayward, CA, 94541')
69	0.00104000	select * from passenger where match(p_address) against ('3215')
70	0.00335650	select * from passenger where p_address = '3215'
71	0.00460125	select * from passenger where p_address like '%3215%'
72	0.00071725	select * from passenger where match(p_address) against ('3215')

```
15 rows in set, 1 warning (0.00 sec)
```

```
mysql> explain select * from passenger where p_address like '%3215%';
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	passenger	NULL	ALL	NULL	NULL	NULL	NULL	2066	11.11	Using where

```
1 row in set, 1 warning (0.00 sec)
```

```
mysql> explain select * from passenger where match(p_address) against ('3215');
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	passenger	NULL	fulltext	idx_passenger_padd	idx_passenger_padd	0	const	1	100.00	Using where; Ft_hints: sorted

```
1 row in set, 1 warning (0.00 sec)
```

3. a) Modified the notice\_board id (nb\_id) datatype from 'INT' to 'TINYINT'. As a result, it decreased the execution time, which can be seen in Query\_ID 46 & 48 respectively.

**Explanation:** Smaller data types are usually faster, as they typically use less space on the disk, in memory, and in the CPU cache. They also generally require fewer CPU cycles to process.

```
mysql> show profiles;
+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+
| 34 | 0.09332125 | alter table flight_info modify nb_id int |
| 35 | 0.00517225 | alter table notice_board modify nb_id int |
| 36 | 0.22598675 | alter table notice_board modify nb_id int |
| 37 | 0.18915100 | alter table flight_passenger_transmits modify nb_id int |
| 38 | 0.00078900 | select * from flight_passenger_transmits |
| 39 | 0.00378925 | show status like "qcache%" |
| 40 | 0.00217150 | show status like "%qcache%" |
| 41 | 0.00210100 | show status like "%q%" |
| 42 | 0.00213075 | show status like "%qcache%" |
| 43 | 0.00171700 | show status like "%cache%" |
| 44 | 0.00350050 | show variables like 'query_cache_size' |
| 45 | 0.00350950 | show variables like '%query_cache_size%' |
| 46 | 0.00360875 | select * from notice_board |
| 47 | 0.14153675 | alter table notice_board modify nb_id tinyint |
| 48 | 0.00066275 | select * from notice_board |
+-----+-----+-----+
15 rows in set, 1 warning (0.00 sec)
```

3. b) Modified the Employee Pincode (E\_pincode) datatype from 'INT' to 'MEDIUMINT'. This decreased the execution time, which is noticeable in Query\_ID 68 & 70.

**Explanation:** Smaller data types are usually faster, because they use less space on the disk, in memory, and in the CPU cache. They also generally require fewer CPU cycles to process.

```
mysql> show profiles;
+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+
| 56 | 0.00058425 | explain select * from passenger where pfirst_name like '%ak%' or plast_name like '%jd%' |
| 57 | 0.00725650 | desc employee |
| 58 | 0.00352025 | select * from employee |
| 59 | 0.00344550 | SHOW VARIABLES LIKE 'have_query_cache' |
| 60 | 0.00082125 | select * from employee |
| 61 | 0.00043850 | select * from employee |
| 62 | 0.00041350 | select * from employee |
| 63 | 0.00102300 | select * from employee |
| 64 | 0.00093400 | select * from employee |
| 65 | 0.06714050 | alter table employee modify E_pincode mediumint |
| 66 | 0.00097300 | select * from employee |
| 67 | 0.16199175 | alter table employee modify E_pincode int |
| 68 | 0.00257725 | select * from employee |
| 69 | 0.14948100 | alter table employee modify E_pincode mediumint |
| 70 | 0.00098050 | select * from employee |
+-----+-----+-----+
15 rows in set, 1 warning (0.00 sec)
```

4. Passenger table has the passenger date of birth (p\_dob), we optimized the searching for a particular date by creating an index on p\_dob. This reduced the execution time (observable in Query\_ID 123 & 125). As a result of optimizing using the index, we reduced the number of scans to 1 instead of 5195. Screenshot is attached below.

**Explanation:** Indexing helps in retrieving data more quickly.

```
mysql> show profiles;
```

Query_ID	Duration	Query
111	0.00618975	select * from passenger where p_dob = '1968-04-25'
112	0.37116475	alter table passenger modify p_dob varchar(10)
113	0.00721950	select * from passenger where p_dob = '1968-04-25'
114	0.42369350	alter table passenger modify p_dob char(10)
115	0.00655150	select * from passenger where p_dob = '1968-04-25'
116	0.35008950	alter table passenger modify p_dob date
117	0.00576675	select * from passenger where p_dob = '1968-04-25'
118	0.02131200	show index from passenger
119	0.21784600	create index idx_date on passenger(p_dob)
120	0.00217925	select * from passenger where p_dob = '1968-04-25'
121	0.03052775	drop index idx_date on passenger
122	0.00536675	select * from passenger where p_dob = '1968-04-25'
123	0.00302900	select * from passenger where p_dob = '1968-04-25'
124	0.14106225	create index idx_date on passenger(p_dob)
125	0.00149825	select * from passenger where p_dob = '1968-04-25'

15 rows in set, 1 warning (0.00 sec)

```
mysql> explain select * from passenger where p_dob = '1968-04-25';
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	passenger	NULL	ref	idx_date	idx_date	4	const	1	100.00	NULL

1 row in set, 1 warning (0.02 sec)

```
mysql> explain select * from passenger where p_dob = '1968-04-25';
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	passenger	NULL	ALL	NULL	NULL	NULL	NULL	5195	10.00	Using where

1 row in set, 1 warning (0.00 sec)

5. We updated 6 records as 'NULL' values to the 'State' attribute of 'City' table as desired. Then we used the 'COUNT' function to count all of the records in the table's attribute(State). It takes less time since it merely returns the count of non-null values after execution. When we change the value to '0,' it counts all the values and takes more time compared to the previous case with NULL values.

**Reason:** It takes more time to count the not null values. Execution time is shown below in Query\_ID 38-41.

```
mysql> select count(state) from city;
+-----+
| count(state) |
+-----+
|          494 |
+-----+
1 row in set (0.00 sec)

mysql> select count(state) from city where state is null;
+-----+
| count(state) |
+-----+
|            0 |
+-----+
1 row in set (0.00 sec)
```

```
mysql> show profiles;
+-----+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+-----+
| 27 | 0.00365050 | select count(state) from city |
| 28 | 0.00051425 | select count(state) from city where state is null |
| 29 | 0.00119125 | select count(state) from city where state = '0' |
| 30 | 0.00072875 | select count(state) from city where state is null |
| 31 | 0.00070975 | select count(state) from city |
| 32 | 0.00071200 | select count(state) from city where state = '0' |
| 33 | 0.00092250 | select count(state) from city where state != '0' |
| 34 | 0.00077075 | select count(state) from city where state = '0' |
| 35 | 0.00872750 | update city set state = null where state = '0' |
| 36 | 0.01338000 | update city set state = null limit 5 |
| 37 | 0.00044425 | update city set state = null limit 4 |
| 38 | 0.00228750 | select count(state) from city |
| 39 | 0.00075100 | select count(state) from city where state is null |
| 40 | 0.01206500 | update city set state = '0' where state is null |
| 41 | 0.00361250 | select count(state) from city |
+-----+-----+-----+-----+
15 rows in set, 1 warning (0.00 sec)
```

6. mysql caches the SELECT query as well as the result set, allowing identical selects to run faster as data is fetched from the memory cache. We are not able to perform caching because the mysql version 8.0 does not support it.

7. a) Performed join on city & airport tables and compared it with a subquery.

**Explanation:** Subquery runs the internal queries first and then from the result set again filters out the actual results. Wherein join runs on both the tables and produces the result in one go.

```
mysql> show profiles;
+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+
| 1 | 0.00177125 | select city_name from city where state = (select a_state from airport where a_state = 'Jjrxqnhh') |
| 2 | 0.00103325 | select city.city_name from city natural join airport where airport.a_state = 'Jjrxqnhh' |
+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)
```

7. b) More joins in a query implies the database server needs to work harder, which means the data retrieval process takes longer which can be a major drawback.

[illegible]

8. The execution time decreased (observable in Query\_ID 1 and 2) as we used natural join instead of subqueries. Number of scans was also reduced when we optimized the 3 subqueries.

**Explanation:** Subquery runs the internal queries first and then from the result set again filters out the actual results. Whereas natural join runs on both the tables and produces the result in one go.

```
mysql> show profiles;
```

Query_ID	Duration	Query
1	0.00161198	select country from city where state = (select state from city where city_name = (select city_name from airport where a_name = (select a_name from employee where emp_id = 'EMP 28742')));
2	0.00964525	select city.country from city natural join airport natural join employee where emp_id = 'EMP 28742';

2 rows in set, 1 warning (0.00 sec)

  

```
mysql> explain select country from city where state = (select state from city where city_name = (select city_name from airport where a_name = (select a_name from employee where emp_id = 'EMP 28742')));
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	city	NONE	ALL	NONE	NONE	NONE	NONE	1004	10.00	Using where
2	SUBQUERY	city	NONE	const	PRIMARY	PRIMARY	122	const	1	100.00	NONE
3	SUBQUERY	airport	NONE	const	PRIMARY	PRIMARY	282	const	1	100.00	NONE
4	SUBQUERY	employee	NONE	const	PRIMARY	PRIMARY	62	const	1	100.00	NONE

4 rows in set, 1 warning (0.01 sec)

  

```
mysql> explain select city.country from city natural join airport natural join employee where emp_id = 'EMP 28742';
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	employee	NONE	const	PRIMARY,A_Name	PRIMARY	62	const	1	100.00	NONE
1	SIMPLE	airport	NONE	const	PRIMARY,city_name	PRIMARY	282	const	1	100.00	NONE
1	SIMPLE	city	NONE	const	PRIMARY	PRIMARY	122	const	1	100.00	NONE

3 rows in set, 1 warning (0.00 sec)