Lab 03

CO527 Advanced Database Systems

E/17/318

Senanayake S.M.U.A

01. Use *explain* to analyze the outputs of the following two simple queries which use only one table access

a. SELECT * FROM departments WHERE deptname = ’Finance’;

```
mysql> EXPLAIN SELECT * FROM departments WHERE dept_name = 'Finance';
+----+-------------+-------------+------------+------+---------------+------+---------+------+------+----------+-------------+
| id | select_type | table       | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra       |
+----+-------------+-------------+------------+------+---------------+------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | departments | NULL       | ALL  | NULL          | NULL | NULL    | NULL |    9 |    11.11 | Using where |
+----+-------------+-------------+------------+------+---------------+------+---------+------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)
```

b. SELECT * FROM departments WHERE deptno =’d002’;

```
mysql> EXPLAIN SELECT * FROM departments WHERE dept_no = 'd002';
+----+-------------+-------------+------------+-------+---------------+---------+---------+-------+------+----------+-------+
| id | select_type | table       | partitions | type  | possible_keys | key     | key_len | ref   | rows | filtered | Extra |
+----+-------------+-------------+------------+-------+---------------+---------+---------+-------+------+----------+-------+
|  1 | SIMPLE      | departments | NULL       | const | PRIMARY       | PRIMARY | 16      | const |    1 |   100.00 | NULL  |
+----+-------------+-------------+------------+-------+---------------+---------+---------+-------+------+----------+-------+
1 row in set, 1 warning (0.00 sec)
```

What conclusions you can draw from the results?

When the department number is used in the where clause, the select query uses the primary key as an index.

Type is changed from ALL to const, which means MYSQL will not go through examining each record

Rows have been reduced from 9 to 1. So the filtering is 100% in the second query.

In conclusion, using the primary key for selecting rows will increase the query execution performance.

02. **Which employees have worked for more than 4000 days for an assigned title**

```
mysql> explain select first_name, period from emplist inner join titleperiod on titleperiod.emp_no = emplist.emp_no where period > 4000;
+----+-------------+-------------+------------+------+---------------+------+---------+------+--------+----------+-----------------------------------------------+
| id | select_type | table       | partitions | type | possible_keys | key  | key_len | ref  | rows   | filtered | Extra                                         |
+----+-------------+-------------+------------+------+---------------+------+---------+------+--------+----------+-----------------------------------------------+
|  1 | SIMPLE      | titleperiod | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 442081 |    33.33 | Using where                                   |
|  1 | SIMPLE      | emplist     | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 299823 |    10.00 | Using where; Using join buffer (hash join) |
+----+-------------+-------------+------------+------+---------------+------+---------+------+--------+----------+-----------------------------------------------+
2 rows in set, 1 warning (0.00 sec)
```

Type is ‘ALL’ which means MYSQL will have to examine each row. The total row combinations that MySQL needs to check is around 442081 * 299823. A large number of row counts shows that there is room for improvement.

3. Create indexes on the columns used to join the tables

i.
```
ALTER TABLE `company`.`emplist`
ADD PRIMARY KEY (`emp_no`);
```

ii.
```
ALTER TABLE `company`.`titleperiod`
ADD INDEX `empNoidx` (`emp_no` ASC) VISIBLE;
;
```

```
mysql> explain select first_name, period from emplist inner join titleperiod on titleperiod.emp_no = emplist.emp_no where period > 4000;
+----+-------------+-------------+------------+------+---------------+----------+---------+-----------------------+--------+----------+-------------+
| id | select_type | table       | partitions | type | possible_keys | key      | key_len | ref                   | rows   | filtered | Extra       |
+----+-------------+-------------+------------+------+---------------+----------+---------+-----------------------+--------+----------+-------------+
|  1 | SIMPLE      | emplist     | NULL       | ALL  | PRIMARY       | NULL     | NULL    | NULL                  | 299576 |   100.00 | NULL        |
|  1 | SIMPLE      | titleperiod | NULL       | ref  | empNoidx      | empNoidx | 5       | company.emplist.emp_no |      1 |    33.33 | Using where |
+----+-------------+-------------+------------+------+---------------+----------+---------+-----------------------+--------+----------+-------------+
2 rows in set, 1 warning (0.00 sec)
```

The query has been optimized and both the tables use the indexes. Total row combinations to check have been drastically decreased.

Type of the second operation has changed from ALL to ref, that way in the titleperiod table MYSQL will select one row for an employee without going through all the rows. This is due to MYSQL using empNoidx in the titleperiod table.

By adding an index for the period column, the query might be optimized further. Also, the first operation is not using the primary key as an index. It might be optimized if the index was forced.