TP05 *View Index and Explain*

Using database employees

1. Using view table

    - create view as dept_emp_view that join table dept_emp and departments
    - alter the previous view dept_emp join the dept_manager
    - create view that show employee information, salary information and the other column as income show that if salary equal or bigger than 80000 highest , and salary >=60000 medium and salary >= 50000 low.

    Employees information: show only CONCAT(first name," ",last name) as Employee's Name Salary: show only amount money and name it as Salary Income:show only medium, high, and low name it as Income.

    - Condition salary < 80000 and salary > 50000 1000 rows or less

2. Using Index

    - Alter employees table with 4 unique (birth_date, first_name, last_name,gender) to identified unique employee
    - alter fulltext in employees table like (first name and last name) in order to search text.

3. Using Explain or Describe

    - using explain to show data structure of the table titles
    - enable format explain tree in order to analyze the query.
    - using json format to explain the query and show about the evaluation.

TP05 *Index*

Objective: How to

- Understand execution plan
- Get execution time for answering a query
- Work with index (show, choose, create, drop)

Some SQL command:

- `EXPLAIN select_statement` # go get the execution plan of query; read mysql reference manual selection 8.8.2 for the detail about explain.
- `SELECT SQL_NO_CACHE …….` # the result of the query is not cache, and dbms won't use query cache to answer the query as well.
- `SHOW INDEX FROM table_name;` # show indexes of a table;
- `SELECT * FROM INFORMATION_SCHEMA.STATISTICS WHERE table_schema = 'db_name';` # show of a database.w32
- `CREATE INDEX index_name ON table_name(field_name) USING index_type;` example: `create index emp_index on employees(emp_no) using btree;` # use HELP index for more infomration

Run the following commands from command line

to inform the DBMS to store some information for showing profiling (execution time and time spent on each stage of the query execution) of each query.

```
UPDATE performance_schema.setup_instruments SET ENABLED = 'YES', TIMED =
'YES'
WHERE NAME LIKE '%statement%';
```

```
UPDATE performance_schema.setup_instruments SET ENABLED = 'YES', TIMED =
'YES'
WHERE NAME LIKE '%stage%';
```

```
UPDATE performance_schema.setup_consumers SET ENABLED = 'YES'
WHERE NAME LIKE '%events_statements_%';
```

```
UPDATE performance_schema.setup_consumers SET ENABLED = 'YES'
WHERE NAME LIKE '%events_stages_%';
```

Get execution time of query using command:

- run the statement you want to examin
- run the following command to get the execution time

```
SELECT EVENT_ID, TRUNCATE(TIMER_WAIT/1000000000000,6) as Duration, SQL_TEXT
FROM performance_schema.events_statements_history_long WHERE SQL_TEXT like
'%part_of
sql_statement%';
```

- run the following command to get spend spending on each stage of query evaluation

```
SELECT event_name AS Stage, TRUNCATE(TIMER_WAIT/1000000000000,6) AS
Duration
FROM performance_schema.events_stages_history_long WHERE
NESTING_EVENT_ID=event_id_from_previous_command;
```

Get execution time of query using Mysql workbench : Look at the Action Output of Mysql workbench. The Duration is the execution time for the query, and Fetch is the time spent on sending the result back to client; for example the execution time for the last query is 8.711 sec. Work

1. Write SQL query for the following queries.

- Find employees with the same first name.
- Find employees with the same first and last name.- Show employees number and hired date order by hired date in increasing order.
- Find the latest salary increasing for each employees. (see TPSQL correction for help)

1. Show all index of each table in the databases employees.

2. Use command SHOW CREATE TABLE table_name; to see how MySQL use index to improve query performance.

3. Give hint to the DBMS on index using to see the different between executing a query with and without index. Test with all queries in exercise 4. Remark: restart Mysql server between the same query run to get more accurate execution time. Simple syntax: (read mysql reference manual selection 8.9.4 for the detail about index hint.)

```
SELECT SQL_NO_CACHE attribute_list
FROM table_name USE INDEX (index_name) , table_name USE INDEX
(index_name) ,…..
WHERE condition_list;
```

```
SELECT SQL_NO_CACHE attribute_list
FROM table_name IGNORE INDEX (index_name) , table_name IGNORE INDEX
(index_name) ,…..
WHERE condition_list;
```

a) Run the following query groups and observe the result, and use EXPLAIN to see the difference in execution plans.

```
select sql_no_cache * from departments ignore index (primary, dept_name);

select sql_no_cache * from departments;
```

```
select sql_no_cache emp_no , hire_date from employees order by hire_date;

select sql_no_cache emp_no , hire_date from employees ignore index
(hired_date) order by hire_date;
```

```
select sql_no_cache e.*
from departments d natural join dept_emp de natural join employees e
where dept_name = 'finance' and to_date < curdate();

select sql_no_cache e.*
from departments d ignore index(dept_name) natural join
```

```
dept_emp de ignore index (primary, dept_no) natural join
employees e ignore index (primary)
where dept_name = 'finance' and to_date < curdate();
```

```
select sql_no_cache employees.*
from employees natural join salaries ignore index (salary_index)
where to_date = '9999-01-01' and gender = 'F'
    and salary = (select max(salary) from employees natural join salaries
    where to_date = '9999-01-01' and gender = 'F' );

select sql_no_cache employees.*
from employees natural join salaries
where to_date = '9999-01-01' and gender = 'F'
    and salary = (select max(salary) from employees natural join salaries
    where to_date = '9999-01-01' and gender = 'F' );
```

```
select sql_no_cache *
from employees where emp_no in
(select de1.emp_no from dept_emp de1 join dept_emp de2 on
de1.emp_no = de2.emp_no and de1.dept_no>de2.dept_no);

select sql_no_cache employees.*, count(dept_no) as numberOfDept
from employees natural join dept_emp
group by dept_emp.emp_no
having count(dept_no) >1;
```

1. How can you improve each query that takes more than 2 seconds to run?

- Queries in exercise 1.
- Queries in TPSQL: 21, 22, 23. (use TPSQLcorrection.sql)