Practice 5

Query Optimization

KOREA UNIVERSITY
DATABASE LAB

hyunsikyoon@korea.ac.kr

Equivalent SQL Statements

- DB Setup & Table Information
- Equivalent SQL Statements Exercise

DB Setup & Table Information

- We'll reuse the "table1"
- If you don't have it, check the course material tab on blackboard

Attribute	Data type	Cardinality	Features
Sorted	Integer	*2000000*	Sorted
Unsorted	Integer	*2000000*	Unsorted
rndm	Integer	*100000*	Dummy field
Dummy	Char(40)	*1*	Dummy field

'table1' table

Equivalent SQL Statements Exercise

- Consider the following query and make corresponding SQL statements

  Select “unsorted” from table1 where the “unsorted” value is 967 or 968 or 969 (967~969)
  (Use the DISTINCT for the question a,b,c)

  a. Make an SQL statement using BETWEEN and AND operator
  b. Make an SQL statement using IN operator
  c. Make an SQL statement using = and OR operator
  d. Make an SQL statement using UNION operator

- Execute your SQL statements and discuss the results of the queries
- Create an index on “unsorted” column and repeat the previous exercise
  a. Btree index
  b. Hash index
- Compare each SQL statements’ performances on three cases(no index, Btree index, hash index)

Query Plan

- DB Setup & Table Information
- Query Plan Exercise

DB Setup & Table Information

- Create two synthetic data tables that has 5000000 rows with values between 0 and 500
  1. CREATE TABLE pool1 (val integer);
  2. INSERT INTO pool1(val) SELECT random()*500 FROM (SELECT generate_series(1,5000000)) AS T;
  3. CREATE TABLE pool2 (val integer);
  4. INSERT INTO pool2(val) SELECT random()*500 FROM (SELECT generate_series(1,5000000)) AS T;

Query Plan Exercise

- Following queries have different syntax but return same result. You have to use **UNION ALL** operator!(different from Equi-SQL statement problem)
  a. Union tables(poo11, pool2), and then perform aggregation with COUNT function
  b. Perform aggregation with COUNT function on each table, and then aggregate them again with SUM function on the union of the aggregated results
- Write the queries and use EXPLAIN ANALYZE to see how the query execution is actually planned

- Following queries also return same result but can be written in different ways. You have to use **UNION ALL** operator!(different from Equi-SQL statement problem)
  a. SELECT tuple WHERE value is above 250 on each table and then union them
  b. Union two tables and SELECT tuples WHERE value is above 250
- Write the queries and use EXPLAIN ANALYZE to see how query execution is actually planned
- Why does the user-level optimization important?

Homework 6

- Follow the directives in this slides
- Take screenshots of your queries and execution results
- Submit your homework online (blackboard)
  - Deadline
    - 10:30 am, December 10th, 2019 (before class)
  - **Only PDF files** are accepted
  - **No late submissions!**
  - **No plagiarism!**

End

hyunsikyoon@korea.ac.kr