Departamento de Engenharia Informática

Bases de Dados

Lab 10: Indices & Optimisation

Index Creation

1. Use the **index_data.sql** script to populate the **account** table with the **\i index_data.sql** command

2. Ensure that command timing is on by running the the command **"\timing"**

3. Write two queries: one to obtain the of accounts with a balance equal to €1000, and another to obtain the maximum balance.

4. Run the queries and note the time it takes the system to execute each command.

```
SELECT account_number FROM account WHERE balance = 1000;

SELECT MAX(balance) FROM ACCOUNT;
```

5. Create an index for the balance column with the command:

```
CREATE INDEX balance_idx ON account(balance);
```

   Is this index primary or secondary? Why?

6. Repeat step 4 and note the time. For both queries, how do you explain the possible time difference?

7. Delete the index created previously in step 5

```
DROP INDEX balance_idx;
```

8. Create a HASH index for the balance column with the command:

```
CREATE INDEX balance_idx ON account USING HASH(balance);
```

9. Repeat step 4 and note the time. How do you explain the possible time difference?

10. Delete the index created in paragraph 8:

```
DROP INDEX balance_idx;
```

Execution Plans

11. Run the **index_data.sql** script again to populate the account table with the command **\i index_data.sql**

12. Get execution plan for the query of step 4 with the command:

```
EXPLAIN SELECT MAX(balance) FROM ACCOUNT;
```

What access method is used? Justify.

13. Now create a B+TREE index on the balance attribute and check the access plan again:

```
CREATE INDEX balance_idx ON account (balance);

EXPLAIN SELECT MAX(balance) FROM ACCOUNT;
```

What difference do you see in the access method?

14. Create a HASH index for the balance column, compare the access plan with step 14

```
DROP INDEX balance_idx;

CREATE INDEX balance_idx ON account USING HASH (balance);

EXPLAIN SELECT MAX(balance) FROM ACCOUNT;
```

How do you explain that the hash index is never used?

Query Optimisation

15. Given a table:

```
CREATE TABLE employee (
  eid INTEGER PRIMARY KEY,
  ename VARCHAR(40) NOT NULL,
  address VARCHAR(255) NOT NULL,
  salary NUMERIC(12,4) NOT NULL,
  bdate DATE NOT NULL);
```

Which indexes can you create to make improve the efficiency of the execution of each of the following queries (supposing that each of them is quite common):

a) What is the identifier, name, and address of employees aged within a certain range?

b) What is the identifier and address of employees with a given name?

c) What is the maximum salary for employees?

d) What is the average salary of employees by age?

TIP: Consider writing down the SQL query and then analysing which indices would be more advantageous.