BAB VI
SUB QUERY DAN INDEKS

7.1 Bahasan dan Sasaran

7.1.1 Bahasan

- Pada bab kali ini akan membahas tentang sub query dan indeks
- Selain hal itu akan dibahas juga mengenai cluster, kolom unik dan trim

7.1.2 Sasaran

1. Mahasiswa memahami dan menggunakan sub query dan indeks dalam pengelolaan database.
2. Mahasiswa memahami cara penggunaan cluster, kolom unik dan trim.

7.2 Materi

7.2.1 SUB QUERY

Subquery atau query Nested merupakan bentuk query yang terdapat dalam query yang lain. *Subquery* dapat ditempatkan dalam klausa where, having, from bersama dengan operator perbandingan seperti = untuk baris tunggal dan untuk baris berganda menggunakan in, not in atau <>, < any, >, >=,<=. Penggunaan sub query dapat diterapkan pada pernyataan SELECT, UPDATE, DELETE, dan INSERT. Bentuk penggunaannya sebagai berikut :

Select nama_kolom from nama_tabel where nama_kolom operator (subquery);

Berikut contoh dari subquery menggunakan data pegawai :

Id_peg	Nama_peg	Alamat_peg	Telp_peg	Jabatan_peg	Gaji_peg
1	Hendro	Solo	081223300	Teknisi	900000
2	Tika	Semarang	0897735357	Sekretaris	2000000
3	Wijaya	Jogjakarta	0865433225	Kepala	3000000
4	Dodi	Banyuwangi	076544677	Teknisi	1000000

- Mencari nama pegawai yang memiliki jabatan yang sama dengan pak hendro bisa menggunakan query sebagai berikut :

Select nama_peg,jabatan_peg from pegawai where jabatan_peg in (select jabatan_peg from pegawai where nama_peg='Hendro');

Hasil :

```
MySQL [praktikumdb]> select * from pegawai;
+--------+----------+------------+-----------+-------------+----------+
| id_peg | nama_peg | alamat_peg | telp_peg  | jabatan_peg | gaji_peg |
+--------+----------+------------+-----------+-------------+----------+
|      1 | Hendro   | Solo       |  81223300 | Teknisi     |   900000 |
|      2 | Tika     | Semarang   | 897735357 | Sekretaris  |  2000000 |
|      3 | Wijaya   | Jogjakarta | 856433225 | Kepala      |  3000000 |
|      4 | Dodi     | Banyuwangi | 765489098 | Teknisi     |  1000000 |
+--------+----------+------------+-----------+-------------+----------+
4 rows in set (0.00 sec)

MySQL [praktikumdb]> Select nama_peg,jabatan_peg from pegawai where jabatan_peg in  (select jabatan_peg from pegawai whe
re nama_peg='Hendro');
+----------+-------------+
| nama_peg | jabatan_peg |
+----------+-------------+
| Hendro   | Teknisi     |
| Dodi     | Teknisi     |
+----------+-------------+
2 rows in set (0.09 sec)
```

- Mencari nama pegawai yang gajinya lebih besar dari pegawai dengan nama Dodi bisa menggunakan query sebagai berikut :

  select nama_peg,gaji_peg from pegawai where gaji_peg > any (select gaji_peg from pegawai where nama_peg ='Dodi');

  Hasil :

```
MySQL [praktikumdb]> select nama_peg,gaji_peg from pegawai where gaji_peg > any (select gaji_peg from pegawai where nama
_peg ='Dodi');
+----------+----------+
| nama_peg | gaji_peg |
+----------+----------+
| Tika     |  2000000 |
| Wijaya   |  3000000 |
+----------+----------+
2 rows in set (0.15 sec)
```

- Mencari nama pegawai yang gajinya lebih besar dari 950000 dan jabatannya bukan seperti jabatan pak hendro bisa menggunakan query sebagai berikut :

  select nama_peg, jabatan_peg, gaji_peg from pegawai where gaji_peg >= 950000 and jabatan_peg <> (select jabatan_peg from pegawai where nama_peg='Hendro');

  Hasil :

```
MySQL [praktikumdb]> select nama_peg, jabatan_peg, gaji_peg from pegawai where gaji_peg >= 950000 and jabatan_peg <> (se
lect jabatan_peg from pegawai where nama_peg='Hendro');
+----------+-------------+----------+
| nama_peg | jabatan_peg | gaji_peg |
+----------+-------------+----------+
| Tika     | Sekretaris  |  2000000 |
| Wijaya   | Kepala      |  3000000 |
+----------+-------------+----------+
2 rows in set (0.08 sec)
```

7.2.2 INDEKS

Indeks disini berguna dalam suatu pencarian nilai atau data dalam database. Dalam suatu kasus ketika mengakses sebuah tabel biasanya DBMS akan membaca seluruh tabel baris perbaris hingga selesai. Ketika baris

sangat banyak dan hasil dari query hanya sedikit, maka hal ini sangat tidak efisien. Seperti halnya ketika kita membaca sebuah buku dan ingin mencari kata atau istilah tertentu dalam buku maka biasanya akan di cari dengan membuka setiap halaman dari awal sampai akhir. Dengan adanya indeks buku maka kita cukup dengan membuka indeks, sehingga akan cepat dalam pencarian kata tersebut. MariaDB tidak bisa membuat indeks dengan otomatis, sehingga *user* dapat membuat indeks tersebut untuk sering kali digunakan kolom, biasanya dalam *clause* WHERE. Berikut struktur SQL :

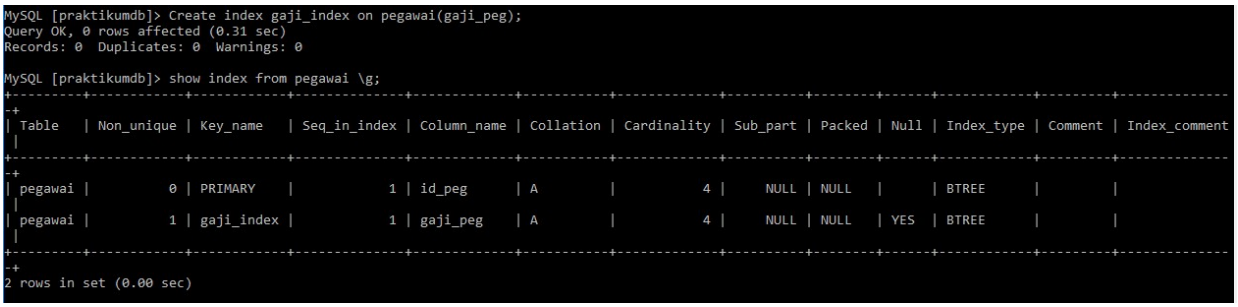
CREATE INDEX nama_index ON nama_tabel (nama kolom);

Contoh :

- Pada tabel pegawai kita berikan index pada kolom gaji untuk query sebagai berikut :

Create index gaji_index on pegawai(gaji_peg);

Hasil :



```
MySQL [praktikumdb]> Create index gaji_index on pegawai(gaji_peg);
Query OK, 0 rows affected (0.31 sec)
Records: 0  Duplicates: 0  Warnings: 0

MySQL [praktikumdb]> show index from pegawai \g;
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+
| Table   | Non_unique | Key_name   | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+
| pegawai |          0 | PRIMARY    |            1 | id_peg      | A         |           4 |     NULL | NULL   |      | BTREE      |         |               |
| pegawai |          1 | gaji_index |            1 | gaji_peg    | A         |           4 |     NULL | NULL   | YES  | BTREE      |         |               |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+
2 rows in set (0.00 sec)
```

indeks sebaiknya jangan digunakan pada tabel atau kolom yang sangat jarang atau tidak pernah diakses. Selain untuk perintah SELECT Indeks juga bermanfaat untuk UPDATE dan DELETE yang menggunakan kondisi pencarian. Sedangkan *Unique index* mirip dengan indeks tetapi lebih digunakan untuk mencegah duplikasi nilai yang terdapat dalam tabel. Jadi dengan adanya *unique index* berarti pembaca tidak dapat meng-*insert* nilai yang sama dalam sebuah tabel. Berikut struktur SQL nya :

CREATE UNIQUE INDEX nama_index ON nama_tabel (nama kolom);

Untuk menghapus index berikut strukturnya :

DROP INDEX Nama_index ON nama_tabel;

Contoh :

- Pada tabel pegawai kita berikan index yang bersifat unik pada kolom nama, untuk query sebagai berikut :

Create unique index unama_index on pegawai(nama_peg);

Hasil :

```
MySQL [praktikumdb]> Create unique index unama_index on pegawai(nama_peg);
Query OK, 0 rows affected (0.26 sec)
Records: 0  Duplicates: 0  Warnings: 0

MySQL [praktikumdb]> show index from pegawai \g;
+---------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-------------
--+
| Table   | Non_unique | Key_name    | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_commen
t |
+---------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-------------
--+
| pegawai |          0 | PRIMARY     |            1 | id_peg      | A         |           4 |     NULL | NULL   |      | BTREE      |         |
  |
| pegawai |          0 | unama_index |            1 | nama_peg    | A         |           4 |     NULL | NULL   | YES  | BTREE      |         |
  |
| pegawai |          1 | gaji_index  |            1 | gaji_peg    | A         |           4 |     NULL | NULL   | YES  | BTREE      |         |
  |
+---------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-------------
--+
3 rows in set (0.00 sec)
```

Apabila kita memasukkan dengan nama yang sudah terdapat pada data terdahulu akan terdapat error.

7.2.3 KOLOM UNIK

Unique berfungsi untuk menjaga agar tidak terjadinya duplikasi nilai (kesamaan data) dalam sebuah kolom, hal ini dapat ditangani dengan membuat sebuah indeks unik atau fungsi unik sendiri pada kolom yang dimaksud. Unique ini sering digunakan dalam pembuatan bukan primary key namun membutuhkan cek dupikasi agar tidak ada yang sama, karena dalam primary key sudah otomatis mempunyai sifat unik. Berikut Struktur SQL saat pembuatan tabel baru :

CREATE TABLE nama_tabel (nama_kolom tipe_data unique);

Ketika tabel sudah ada kita bisa menggunakan cara seperti pada BAB. 2 berikut struktur SQL nya :

ALTER TABLE nama_tabel ADD CONSTRAINT namaConstraint UNIQUE (nama_kolom);

Untuk menghapus unique berikut caranya :

ALTER TABLE nama_table DROP INDEX nama_constraint;

Contoh :

```
MySQL [praktikumdb]> alter table pegawai add unique (nama_peg);
Query OK, 0 rows affected, 1 warning (0.41 sec)
Records: 0  Duplicates: 0  Warnings: 1

MySQL [praktikumdb]> show index from pegawai \g;
+---------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-------------
--+
| Table   | Non_unique | Key_name    | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_commen
t |
+---------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-------------
--+
| pegawai |          0 | PRIMARY     |            1 | id_peg      | A         |           4 |     NULL | NULL   |      | BTREE      |         |
  |
| pegawai |          0 | unama_index |            1 | nama_peg    | A         |           4 |     NULL | NULL   | YES  | BTREE      |         |
  |
| pegawai |          0 | nama_peg    |            1 | nama_peg    | A         |           4 |     NULL | NULL   | YES  | BTREE      |         |
  |
| pegawai |          1 | gaji_index  |            1 | gaji_peg    | A         |           4 |     NULL | NULL   | YES  | BTREE      |         |
  |
+---------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-------------
--+
4 rows in set (0.00 sec)

ERROR: No query specified

MySQL [praktikumdb]> desc pegawai;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| id_peg      | int(11)     | NO   | PRI | NULL    |       |
| nama_peg    | varchar(15) | YES  | UNI | NULL    |       |
| alamat_peg  | varchar(50) | YES  |     | NULL    |       |
| telp_peg    | int(15)     | YES  |     | NULL    |       |
| jabatan_peg | varchar(15) | YES  |     | NULL    |       |
| gaji_peg    | int(15)     | YES  | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.41 sec)
```

7.2.4 Penggunaan TRIM

Suatu ketika pasti akan memiliki data yang di dalamnya terdapat spasi kosong yang tidak diperlukan, misalnya spasi ganda. Jika ada masalah seperti ini, kita dapat membersihkan spasi-spasi kosong yang tidak diperlukan menggunakan fungsi TRIM, RTRIM, dan LTRIM. Ketiga fungsi ini memiliki bentuk penggunaan sebagai berikut :

- RTRIM : digunakan untuk membersihkan spasi kosong yang ada di bagian kanan (Right) String.
- LTRIM : digunakan untuk membersihkan spasi kosong yang ada di bagian kiri (Left) String.
- TRIM : digunakan untuk membersihkan spasi kosong yang ada di bagian kiri, kanan, maupun tengah String

Berikut Struktur SQL nya :

Select trim(nama_kolom) from nama_tabel;

Dalam penggunaannya, fungsi TRIM memiliki tiga opsi. Ketiga opsi ini dapat digunakan untuk menentukan karakter apa yang akan dihapus dari suatu String. Jadi, fungsi TRIM juga dapat menghilangkan karakter tertentu (bukan spasi kosong saja) dari suatu string. Opsinya sebagai berikut :

- LEADING : merupakan opsi untuk menghilangkan karakter terpilih yang ada di sebelah kiri. Parameter Leading diartikan sebagai sufik dari karakter yang ada.
- TRAILING : merupakan opsi untuk menghilangkan karakter terpilih yang ada di sebelah kanan String. Parameter Trailing diartikan sebagai sufik dari karakter yang ada.
- BOTH : merupakan opsi yang dapat menangani parameter Leading maupun Trailing.

Contoh:

```
 1
MySQL [praktikumdb]> select trim(LEADING ' 'from 'pegawai');
+---------------------------------+
| trim(LEADING ' 'from 'pegawai') |
+---------------------------------+
| pegawai                         |
+---------------------------------+
1 row in set (0.00 sec)

MySQL [praktikumdb]> select trim(TRAILING ' 'from 'pegawai');
+----------------------------------+
| trim(TRAILING ' 'from 'pegawai') |
+----------------------------------+
| pegawai                          |
+----------------------------------+
1 row in set (0.00 sec)

MySQL [praktikumdb]> select trim(LEADING 'a'from 'aaa987');
+--------------------------------+
| trim(LEADING 'a'from 'aaa987') |
+--------------------------------+
| 987                            |
+--------------------------------+
1 row in set (0.00 sec)
```

Tugas Praktikum

Tugas praktikum kali ini masih menggunakan tabel pada praktikum sebelumnya :

1. Tampilkan nama fakultas dan jumlah mahasiswa yang mampunyai ketentuan fakultas yang dimunculkan dengan jumlah mahasiswanya terkecil!
2. Tampilkan nama mahasiswa, nama fakultas, alamat dengan syarat nama fakultas sama dengan edi dan alamatnya tidak sama dengan luki!
3. Buatlah index di tabel mahasiswa(alamat). Kemudian buat lagi index yang bersifat unik pada tabel fakultas(fak_nama) kemudian amati perbedaannya ketika memasukkan data yang sama!
4. Buat kolom nama di mahasiswa menjadi unik dan inputkan 2 data yang sama. Kemudian amati perbedaannya !
5. Pindahkan data dari tabel mahasiswa, fakultas ambil kolom nim, nama mahasiswa, alamat, nama fakultas ke tabel baru yang dinamai 'tabel identitas'.
6. Munculkan data mahasiswa dengan hilangkan karakter "+" di akhir data dan karakter "a" di awal kata pada kolom nama!

Next week

- Middle test
- Close book
- Introduction – sub query index