PRAKTIKUM 7

Materi:

1. DCL (Data Control Language)
2. Advanced SQL (Join, View, Nested Query, Procedure, Trigger)

Tujuan Praktikum:

Setelah mengikuti praktikum ini mahasiswa mampu memodifikasi (memberikan atau mencabut) hak akses pada suatu basis data dan mengetahui variasi perintah SQL lebih lanjut untuk beberapa contoh kasus pada PostgreSQL.

A. Penyajian dan Latihan

DCL (Data Control Language)

DCL (Data Control Language) merupakan instruksi untuk mengatur keamanan atau mengubah tipe kontrol pada lebih dari satu lingkungan basis data, meliputi *rights, permissions,* dan *controls*. Hanya *database administrator's* atau *owner's* yang mampu memberikan atau mencabut hak akses pada basis data. Contoh perintah: **GRANT**, **REVOKE.**

- **GRANT** → merupakan perintah untuk **memberikan** hak akses (*privileges*) ke basis data kepada pengguna.
- **REVOKE** → merupakan perintah untuk **mencabut** hak akses (*privileges*) yang telah diberikan menggunakan perintah GRANT kepada pengguna.

Bentuk umum sintaks untuk perintah GRANT:

```
GRANT privilege_name
ON object_name
TO {user_name |PUBLIC |role_name}
[WITH GRANT OPTION];
```

Bentuk umum sintaks untuk perintah REVOKE:

```
REVOKE privilege_name
ON object_name
FROM {user_name |PUBLIC |role_name};
```

Keterangan:

- ➢ `privilege_name` → tipe hak akses yang diberikan kepada pengguna, seperti **ALL**, **EXECUTE**, **SELECT**
- ➢ `object_name` → nama objek basis data, seperti **TABLE, VIEW, STORED PROC, SEQUENCE**
- ➢ `user_name` → nama pengguna yang ingin diberi atau dicabut hak akses
- ➢ `PUBLIC` → hak akses diberi atau dicabut dari seluruh pengguna
- ➢ `WITH GRANT OPTION` → mengizinkan pengguna untuk memberikan hak GRANT kepada pengguna lain

Latihan 1. Apa yang dilakukan oleh perintah GRANT berikut:

```
GRANT SELECT
ON employee
TO user1@localhost
WITH GRANT OPTION;
```

Latihan 2. Apa yang dilakukan oleh perintah REVOKE berikut:

```
REVOKE SELECT
ON  employee
FROM user1@localhost;

REVOKE ALL PRIVILEGES
FROM PUBLIC;
```

Advanced SQL (Structure Query Language)

Untuk mengelola dan mengatur basis data, SQL memiliki perintah lanjutan yang lebih kompleks dari perintah-perintah dasar yang pernah dipelajari di materi sebelumnya, seperti **JOIN, VIEW, NESTED QUERY, PROCEDURE, TRIGGER.**

Untuk mempraktikan advanced SQL ini diperlukan database company yang telah dibuat di pertemuan sebelumnya.

1. **SELECT** FROM MULTIPLE TABLE → melakukan selection dari beberapa tabel sekaligus.
   Contoh:
   ```
   SELECT SSN, FName, LName, DName
   FROM Employee, Department
   WHERE DNum = DNumber
   ORDER BY DNum;
   ```

```
ompany=# select ssn,fname,lname, dname from employee, department where dnum = d
umber order by dnum;
   ssn     |   fname    |     lname     |  dname
-----------+------------+---------------+----------
E001       | Hakim      | Arifin        | HRD
E029       | Fahmilu    | Kurniawan     | HRD
E005       | Vera       | Yunita        | HRD
E006       | Pritasri   | Palupiningsih | HRD
E025       | Idaliana   | K             | HRD
E021       | Karina     | Gusriani      | HRD
E007       | Rifki      | Haidar        | HRD
E008       | Muhammad   | Rosyidi       | HRD
E030       | Wikhdal    | Khusnaini     | FINANCE
E002       | Yuni       | Arti          | FINANCE
E009       | Ferry      | Pratama       | FINANCE
E010       | Andi       | Sasmita       | FINANCE
E011       | Yuhan      | Kusuma        | FINANCE
E012       | Ferdian    | Feisal        | FINANCE
E022       | Netty      | Sitohang      | FINANCE
E026       | Rifki      | Fauzie        | FINANCE
E014       | Benedika   | Hutabarat     | HUMAS
E013       | Albertus   | M             | HUMAS
E003       | Mutia      | Aziza         | HUMAS
E027       | Wisnu      | Priyambodo    | HUMAS
E023       | Diyan      | Kurniawan     | HUMAS
```

Pada query di atas, untuk mendapatkan nama employee dan nama departemen tempat employee bekerja, maka perlu dilakukan selection dari table Employee dan table Departement. Untuk itu perlu ditambahkan pada query statement

```
WHERE DNum = DNumber
```

Dimana DNum adalah field pada table Employee dan DNumber adalah field pada table Departement. Statement where di atas digunakan untuk mencocokkan employee dan departemen tempatnya bekerja yang sesuai.

Menampilkan *Employee* yang pernah bekerja pada proyek yang dikelola oleh *Department*
tempat *Employee* tersebut bekerja.

```
SELECT SSN, FName, LName,
       PName, P.DNum,
       D.DNumber, D.DName
FROM Employee E, Project P, Department D, Works_On W
WHERE E.SSN = W.ESSN AND
      W.PNum = P.PNumber AND P.DNum = D.DNumber AND E.DNum =
      P.DNum
ORDER BY P.PName;
```

ssn	fname	lname	pname	dnum	dnumber	dname
E008	Muhammad	Rosyidi	AAA	1	1	HRD
E001	Hakim	Arifin	AAA	1	1	HRD
E009	Ferry	Pratama	BBB	2	2	FINANCE
E010	Andi	Sasmita	BBB	2	2	FINANCE
E002	Yuni	Arti	BBB	2	2	FINANCE
E022	Netty	Sitohang	CCC	2	2	FINANCE
E011	Yuhan	Kusuma	CCC	2	2	FINANCE
E002	Yuni	Arti	CCC	2	2	FINANCE
E010	Andi	Sasmita	CCC	2	2	FINANCE
E030	Wikhdal	Khusnaini	DDD	2	2	FINANCE
E011	Yuhan	Kusuma	DDD	2	2	FINANCE
E012	Ferdian	Feisal	DDD	2	2	FINANCE
E013	Albertus	M	EEE	3	3	HUMAS
E014	Benedika	Hutabarat	FFF	3	3	HUMAS
E013	Albertus	M	FFF	3	3	HUMAS
E004	Hanif	Affandi	GGG	4	4	PRODUKSI
E020	Edi	Firmasyah	GGG	4	4	PRODUKSI

Perhatikan bahwa query di atas melibatkan 4 tabel, sehingga perlu ditambahkan statement where untuk mengaitkan field-field antar table yang sesuai.

- **JOIN** → melakukan *selection* dari beberapa tabel sekaligus, secara langsung menggabungkan tabel-tabel tersebut dengan syarat-syarat tertentu. Perintah Join memberikan hasil yang sama dengan contoh query sebelumnya yang menggunakan statement where untuk mencocokkan antar table.

  Secara umum cara penulisannya seperti ini :

```
SELECT [aggregation] <column>
FROM <table> JOIN <table> ON <condition>;
```

  Contoh kasus :

1. **Menampilkan *Employee* beserta nama *Department* tempat *Employee* bekerja**

```
SELECT SSN, FName, LName, DName
 FROM Employee JOIN Department ON DNum = DNumber
```

```
ORDER BY DNum;
```

ssn	fname	lname	dname
E001	Hakim	Arifin	HRD
E029	Fahmilu	Kurniawan	HRD
E005	Vera	Yunita	HRD
E006	Pritasri	Palupiningsih	HRD
E025	Idaliana	K	HRD
E021	Karina	Gusriani	HRD
E007	Rifki	Haidar	HRD
E008	Muhammad	Rosyidi	HRD
E030	Wikhdal	Khusnaini	FINANCE
E002	Yuni	Arti	FINANCE
E009	Ferry	Pratama	FINANCE
E010	Andi	Sasmita	FINANCE
E011	Yuhan	Kusuma	FINANCE
E012	Ferdian	Feisal	FINANCE
E022	Netty	Sitohang	FINANCE
E026	Rifki	Fauzie	FINANCE

2. Menampilkan *Employee* yang pernah bekerja pada proyek yang dikelola oleh *Department* tempat *Employee* tersebut bekerja.

```
SELECT SSN, FName, LName, PName, P.DNum, D.DNumber, D.DName
FROM Employee E JOIN Works_On W ON E.SSN = W.ESSN
JOIN Project P ON W.PNum = P.PNumber AND E.DNum = P.DNum
JOIN Department D ON P.DNum = D.DNumber
ORDER BY P.PName;
```

ssn	fname	lname	pname	dnum	dnumber	dname
E008	Muhammad	Rosyidi	AAA	1	1	HRD
E001	Hakim	Arifin	AAA	1	1	HRD
E009	Ferry	Pratama	BBB	2	2	FINANCE
E010	Andi	Sasmita	BBB	2	2	FINANCE
E002	Yuni	Arti	BBB	2	2	FINANCE
E022	Netty	Sitohang	CCC	2	2	FINANCE
E011	Yuhan	Kusuma	CCC	2	2	FINANCE
E002	Yuni	Arti	CCC	2	2	FINANCE
E010	Andi	Sasmita	CCC	2	2	FINANCE
E030	Wikhdal	Khusnaini	DDD	2	2	FINANCE
E011	Yuhan	Kusuma	DDD	2	2	FINANCE
E012	Ferdian	Feisal	DDD	2	2	FINANCE
E013	Albertus	M	EEE	3	3	HUMAS
E014	Benedika	Hutabarat	FFF	3	3	HUMAS
E013	Albertus	M	FFF	3	3	HUMAS
E004	Hanif	Affandi	GGG	4	4	PRODUKSI
E020	Edi	Firmasyah	GGG	4	4	PRODUKSI
E020	Edi	Firmasyah	HHH	4	4	PRODUKSI
E004	Hanif	Affandi	HHH	4	4	PRODUKSI

- **VIEW** → tabel bayangan yang dibangun dari satu atau beberapa tabel yang sudah ada. Secara fisik, VIEW tidak membuat penyimpanan data seperti tabel, melainkan hanya menyimpan referensi/ pointer ke record pada tabel-tabel yang berkaitan. Secara umum sintaks untuk membuat view adalah

```
CREATE VIEW ViewName (Field1, Field1) AS SELECT Field_1,
Field_1, …
FROM TableName
WHERE Condition;
```

Contoh :

```
CREATE VIEW v1 AS
SELECT SSN, FName, LName, DNum
```

```
FROM Employee;

CREATE VIEW v2 (Nomor, Nama) AS
SELECT DNumber, DName
FROM Department;
```

Membuat view dari beberapa tabel:

```
CREATE VIEW v4 AS
       SELECT SSN, FName+' '+LName AS EName, DName
       FROM Employee, Department
       WHERE DNum = DNumber;
```

Dalam View juga berlaku operasi **ALTER, DROP, DAN SELECTION.** Contoh :

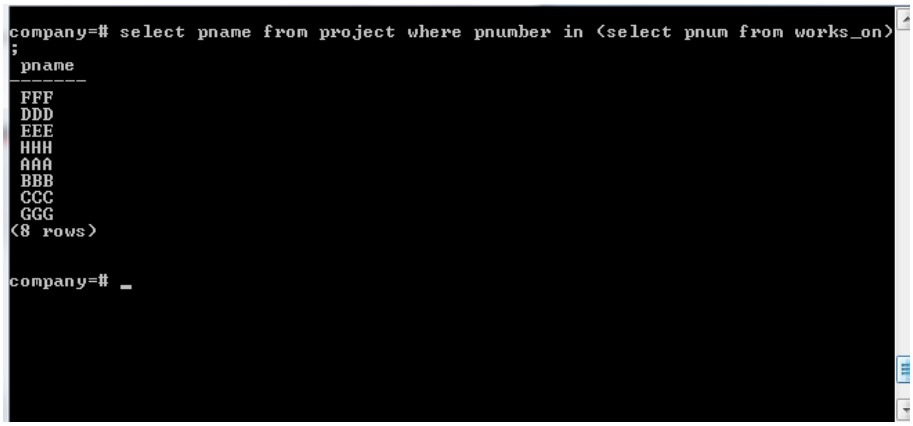
```
ALTER VIEW v1 AS
SELECT SSN, FName, LName, DNum
FROM Employee;

DROP VIEW v1 AS
SELECT SSN, FName, LName, DNum
FROM Employee;

SELECT * FROM v1;
```

- **NESTED QUERY** → suatu query dimana didalamnya terdapat terdapat query lain yang menjadi kondisi.
  Contoh:

```
SELECT PName
FROM Project WHERE PNumber
IN (SELECT PNum FROM Works_On);
```



- **PROCEDURE** → Adakalanya kita perlu melakukan beberapa operasi pada database yang perintah- perintahnya perlu disusun secara terstruktur layaknya pembuatan program. Program yang kita susun tersebut pada PL/SQL disebut dengan Procedure. **Procedure** adalah program yang dapat dipanggil/ dieksekusi oleh procedure lainnya, atau dieksekusi dari Sql-Prompt. Sintaksnya secara umum sebagai berikut :

```
CREATE FUNCTION <nama procedure> (<parameter>)
RETURN <tipe data>
AS
'BEGIN
<code>
END; '
LANGUAGE 'plpgsql';
```

Contoh :

```
CREATE FUNCTION sp_employee()
AS
'BEGIN
     SELECT * FROM Employee
END;'
LANGUAGE 'plpgsql';
```

Untuk Menghapus suatu procedure caranya dengan menggunakan DROP PROC. Contoh :

```
DROP FUNCTION sp_employee();
```

Cara menjalankan suatu procedure adalah :

```
EXECUTE PROCEDURE <nama procedure>(<parameter>);
```

- **TRIGGER** → jenis khusus dari procedures yang bereaksi terhadap *event* / aksi tertentu pada *database.* Trigger akan secara otomatis dijalankan ketika suatu *event*/aksi berlangsung pada *database*. Event yang dimaksud adalah insert, update, delete database. Cara membuat trigger secara umum sebagai berikut :

```
CREATE TRIGGER <trigger name>
FOR|AFTER|BEFORE <[INSERT][UPDATE][DELETE]>
ON <table or view name>
FOR EACH ROW
<SQL Statement……. >;
```

Contoh :

```
CREATE TRIGGER trigger_tes BEFORE INSERT
ON employee FOR EACH ROW
SELECT * FROM employee;
```

Cara menghapus Trigger :

```
DROP TRIGGER <nama tabel>.<nama trigger>;
```

LEMBAR KERJA PRAKTIKUM 7

Nama:	Tanggal Praktikum:
NRP :	Waktu Praktikum:
Nilai :	Nama Asisten :

1. Tentukan query untuk soal-soal di bawah ini ! (Screenshoot query dan hasilnya)
    a. MENCARI EMPLOYEE YANG MENJADI MANAGER SERTA PROYEK APA SAJA YANG PERNAH IA KERJAKAN (SSN, FName, LName, PName).
    b. MENCARI EMPLOYEE YANG HANYA BEKERJA PADA SATU PROYEK SAJA (ESSN, FName, LName, Dependent_Name, Relationship).
2. Buatlah virtual table (view) dari soal berikut !
    a. View yang memiliki atribut Nomor dan Nama yang isinya adalah Dnumber dan Dname.
    b. View yang isinya adalah employee yang memiliki total hours per week lebih besar dari 120 hours beserta nama departemen tempat dia bekerja dan memiliki atribut SSN, Ename (berasal dari Fname dan Lname), Dname.
3. Buatlah trigger yang akan menjalankan procedure dengan nama show jika ada employee yang dipecat dimana procedure show akan menunjukan banyaknya employee dari setiap Department dan urutkan berdasarkan urutan abjad Nama Department.
4. Buatlah trigger yang akan menjalankan procedure dengan nama display jika ada employee baru yang masuk dimana procedure display akan menampilkan SSN, FName, Lname, Hours.
5. Tentukan dependent name, sex, birthdate, dan relationship dari employee yang memiliki rata-rata hours per week = 50 jam dan bekerja pada 2 project.