

PERTEMUAN 12

VIEW

Tujuan Pembelajaran :

- Memahami definisi View
- Dapat membuat View
- Dapat Memanggil data melalui View
- Merubah definisi View
- Insert, Update, dan Delete data melalui View
- Menghapus (drop) view

TEORI DAN PERCOBAAN

12.1. Definisi View

View adalah salah satu object database, yang secara logika merepresentasikan sub himpunan dari data yang berasal dari satu atau lebih table.

Kegunaan dari view adalah :

- Membatasi akses database
- Membuat query kompleks secara mudah
- Mengijinkan independensi data
- Untuk menampilkan view (pandangan) data yang berbeda dari data yang sama.

Ada 2 (dua) tipe view, yaitu Simple View dan Complex View.

Berikut ini perbandingan antara Simple View dan Complex View :

Fitur	Simple View	Complex View
Jumlah table	Satu	Satu atau lebih
Berisi Fungsi	Tidak	Ya
Berisi Group Data	Tidak	Ya
DML melalui view	Ya	Tidak selalu

12.2. Membuat View

View dapat dibuat dengan perintah **CREATE VIEW**. Subquery dapat dicantumkan dalam CREATE VIEW, tapi subquery yang digunakan tidak boleh berisi klausa ORDER BY.

Percobaan 1 : Buat view EMPVU10 yang berisi detail dari pegawai yang bekerja pada department 10.

```
SQL> CREATE VIEW EMPVU10
  2 AS SELECT empno,ename,job
  3 FROM emp
  4 WHERE deptno=10;

View created.
```

Untuk menampilkan struktur dari view diberikan perintah **DESCRIBE namaview**;

```
SQL> DESCRIBE EMPVU10;
Name                               Null?    Type
-----
EMPNO                               NOT NULL NUMBER(4)
ENAME                               VARCHAR2(10)
JOB                                  VARCHAR2(9)
```

Pada view juga dapat dibuat kolom alias.

Percobaan 2 : Buat view SALVU30 yang berisi nomer, nama dan gaji pegawai yang bekerja di department 30. Beri nama kolom baru yaitu EMPLOYEE_NUMBER, NAME dan SALARY.

```
SQL> CREATE VIEW SALVU30
  2 AS
  3 SELECT empno EMPLOYEE_NUMBER, ename NAME, sal SALARY
  4 FROM EMP
  5 WHERE deptno=30;

View created.
```

12.3. Memanggil data dari View

Untuk memanggil data dari view, digunakan perintah yang sama seperti memanggil data dari table.

Percobaan 3 : Tampilkan semua data yang ada pada view SALVU30

```
SQL> SELECT * FROM SALVU30;
```

EMPLOYEE_NUMBER	NAME	SALARY
7698	BLAKE	2850
7654	MARTIN	1250
7499	ALLEN	1600
7844	TURNER	1500
7900	JAMES	950
7521	WARD	1250

```
6 rows selected.
```

12.4. Memodifikasi View

Untuk memodifikasi View digunakan klausa CREATE OR REPLACE VIEW.

Percobaan 4 : Modifikasi judul kolom dari EMPVU10 (percobaan 2) menjadi seperti berikut :

```
SQL> CREATE OR REPLACE VIEW empvu10  
2 (employee_number, employee_name, job_title)  
3 AS SELECT empno,ename,job  
4 FROM EMP  
5 WHERE deptno=10;
```

```
View created.
```

```
SQL> select * from empvu10;
```

EMPLOYEE_NUMBER	EMPLOYEE_N	JOB_TITLE
7839	KING	PRESIDENT
7782	CLARK	MANAGER
7934	MILLER	CLERK

12.5. Membuat Complex View

Berikut ini akan dicontohkan pembuatan Complex View yang berisi fungsi group untuk menampilkan nilai yang berasal dari dua table.

Percobaan 4 : Buat Complex View DEPT_SUM_VU yang berisi nama department, minimum gaji, maksimum gaji, rata-rata gaji dari seluruh pegawai pada tiap-tiap department

```
SQL> CREATE VIEW dept_sum_vu
 2 (name,minsal,maxsal,avgsal)
 3 AS
 4 SELECT d.dname,min(e.sal),max(e.sal),avg(sal)
 5 FROM EMP e, DEPT d
 6 WHERE e.deptno=d.deptno
 7 GROUP BY d.dname;
```

View created.

```
SQL> SELECT * FROM dept_sum_vu;
```

NAME	MINSAL	MAXSAL	AUGSAL
ACCOUNTING	1300	5000	2916.66667
RESEARCH	800	3000	2175
SALES	950	2850	1566.66667

12.6. Aturan untuk membentuk operasi DML pada View

Berikut ini aturan untuk membentuk operasi DML pada View :

- Operasi DML dapat dibentuk pada Simple View
- Baris data pada View tidak dapat dihapus, jika berisi :
 - Fungsi Group
 - Klausula GROUP BY
 - Keyword DISTINCT
- Data pada View tidak bisa dimodifikasi jika berisi :
 - 3 Kondisi yang sudah disebutkan diatas
 - Kolom yang didefinisikan oleh suatu ekspresi
 - Kolom ROWNUM
- Pada View tidak bisa ditambahkan data, jika :
 - View berisi 5 kondisi yang sudah disebutkan diatas
 - Terdapat kolom NOT NULL pada base table (table asal darimana view dibuat) yang tidak dipilih oleh View.

12.7. Menggunakan Klausula **WITH CHECK OPTION**

Jika klausula WITH CHECK OPTION digunakan, maka tidak diperbolehkan terjadi perubahan data pada kolom yang punya relasi ke table yang lain. Misal pada view EMPVU20 kolom deptno punya relasi ke kolom deptno pada table department, maka perubahan data yang dilakukan pada kolom ini tidak diperbolehkan.

Percobaan 5 : Buat view EMPVU20 yang berisi semua data pegawai pada table EMP yang bekerja di department 20, beri klausula WITH CHECK OPTION.

```
SQL> CREATE OR REPLACE VIEW empvu20
 2 AS
 3 SELECT *
 4 FROM EMP
 5 WHERE deptno=20
 6 WITH CHECK OPTION CONSTRAINT empvu20_ck;
View created.
```

Perubahan yang dilakukan pada view EMPVU20 untuk kolom deptno akan menimbulkan pesan kesalahan.

```
SQL> UPDATE empvu20
 2 SET deptno=10
 3 WHERE empno=7788;
UPDATE empvu20
      *
ERROR at line 1:
ORA-01402: view WITH CHECK OPTION where-clause violation
```

12.8. **Mengabaikan** (tidak memperbolehkan) dilakukan **Operasi DML**

Klausula **READ ONLY** digunakan jika kita ingin mengabaikan atau tidak mengijinkan semua operasi DML yang dilakukan pada data.

Percobaan 6 : Buat view EMPVU10 yang berisi data nomer, nama, dan pekerjaan pegawai untuk pegawai yang bekerja di department 10. Gunakan klausula READ ONLY untuk mengabaikan operasi DML pada view.

```
SQL> CREATE OR REPLACE VIEW empvu10
 2 as
 3 SELECT empno,ename,job
 4 FROM EMP
 5 WHERE deptno=10
 6 WITH READ ONLY;
View created.
```

Setelah dibuat view EMPVU10 tersebut, sembarang operasi DML yang dilakukan pada view, akan menimbulkan pesan kesalahan, seperti pada contoh berikut :

```
SQL> DELETE FROM empvu10
      2 WHERE empno=7782;
DELETE FROM empvu10
      *
ERROR at line 1:
ORA-01752: cannot delete from view without exactly one key-preserved table
```

12.9. Menghapus View

View dapat dihapus dengan menggunakan perintah **DROP VIEW** *nama_view*;

Percobaan 7 : Hapus view EMPVU10

```
SQL> DROP VIEW empvu10;
View dropped.
```

LATIHAN SOAL

1. Buat view EMP_VU yang berisi nomer pegawai, nama pegawai, nomer department yang berasal dari table pegawai. Ubah judul kolom nama pegawai menjadi PEGAWAI.
2. Tampilkan view EMP_VU

EMPNO	PEGAWAI	DEPTNO
7839	KING	10
7698	BLAKE	30
7782	CLARK	10
7566	JONES	20
7654	MARTIN	30
7499	ALLEN	30
7844	TURNER	30
7900	JAMES	30
7521	WARD	30
7902	FORD	20
7369	SMITH	20

EMPNO	PEGAWAI	DEPTNO
7788	SCOTT	20
7876	ADAMS	20
7934	MILLER	10

14 rows selected.

3. Tampilkan nama view dan teks-nya dari data dictionary USER_VIEWS

```
SQL> SELECT view_name,text from user_views;
```

VIEW_NAME	TEXT
EMP_VU	SELECT empno,ename PEGAWAI, deptno FROM EMP

4. Buat view dengan nama DEPT20 yang berisi nomer, nama dan gaji dari pegawai yang bekerja di department 20. Beri judul kolom EMPLOYEE_ID, EMPLOYEE, dan DEPARTMENT_ID. Jangan perbolehkan pegawai untuk mendaftar kembali (atau mengisi datanya lagi) ke department yang lain melalui view.
5. Tampilkan struktur dari view DEPT20.

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(4)
EMPLOYEE		VARCHAR2(10)
DEPARTMENT_ID	NOT NULL	NUMBER(2)

6. Buat view SALARY_VU yang berisi nama pegawai, nama department, gaji dan grade dari gaji untuk semua pegawai. Beri judul PEGAWAI, DEPARTMENT, GAJI, GRADE. Tampilkan data pada SALARY_VU.

ENAME	DNAME	SAL	GRADE
JAMES	SALES	950	1
SMITH	RESEARCH	800	1
ADAMS	RESEARCH	1100	1
MARTIN	SALES	1250	2
WARD	SALES	1250	2
MILLER	ACCOUNTING	1300	2
ALLEN	SALES	1600	3
TURNER	SALES	1500	3
BLAKE	SALES	2850	4
CLARK	ACCOUNTING	2450	4
JONES	RESEARCH	2975	4
FORD	RESEARCH	3000	4
SCOTT	RESEARCH	3000	4
KING	ACCOUNTING	5000	5

14 rows selected.