LAPORAN RESMI

MODUL II

VIEW

SISTEM MANAJEMEN BASIS DATA

NAMA	**:**	**IMAM ARISHANDI IRFANTO**
N.R.P	**:**	**220441100034**
DOSEN	**:**	**FITRI DAMAYANTI, S.Kom,M.Kom**
ASISTEN	**:**	**MOHAMMAD IQBAL FIRMANSYAH**
TGL PRAKTIKUM	**:**	**29 MARET 2023**

Disetujui : 0 APRIL 2024
Asisten

MOHAMMAD IQBAL FIRMANSAY
21.04.411.00084

LABORATORIUM BISNIS INTELIJEN SISTEM
PRODI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Semakin berkembangnya teknologi saat ini maka kebutuhan penyimpanan data dalam skala yang besar sangat dibutuhkan terutama untuk perusahaanperusahaan besar, perangkat data sangat berperan penting dalam perkembangan usahanya. Perangkat lunak yang digunakan untuk mengelola dan memanggil query basis data disebut dengan system manajemen basis data (database management system, DBMS) dalam system basis data dapat dipelajari dalam ilmu informasi. Saat ini, view memang sedang banyak dibahas di beberapa forum programmer terkemuka. Hal ini disebabkan karena dengan view kita bisa membuat query dari banyak tabel lalu kita dapat memanggil view tersebut tanpa membuat query lagi. Seperti halnya stored procedure dan function. Ketika kita membuat sebuah tampilan, kita memerlukan SELECT JOIN. Lalu kita mulai khawatir jika SELECT JOIN tersebut akan sangat banyak memakan ram sehingga program tidak berjalan secara maksimal atau kecepatan menurun. Memang SELECT JOIN membuat program semakin berat karena dengan menggunakan perintah tersebut, maka program akan membaca tabel lebih sehingga jika banyak tabel yang di JOIN atau data yang tersimpan sudah mencapai ribuan menyebabkan pembacaan data dari program lebih lama dan terkesan berat. Namun hal tersebut sudah bisa diatasi dengan View. View dapat menyimpan query dari banyak tabel dan membuatnya pada satu query yang berbentuk tabel. Jadi kita membuat satu tabel dengan bentuk SELECT JOIN yang kemudian kita panggil tabel tersebut. Jika anda pernah mencoba menggunakan microsoft ACCES, disana ada menu query, itulah View.

1.2 Tujuan

- Mampu memahami konsep dasar view didalam basis data.
- Mampu memahami penerapan view.
- Mampu menyelesaikan pengambilan data dengan menggunakan pendekatan view.

BAB II
DASAR TEORI

2.1 View

View adalah tabel virtual yang terbuat dari suatu query terhadap satu tabel atau beberapa tabel. View tidak ada secara nyata dalam database. View hanya digunakan untuk menyederhanakan dan mempermudah persepsi pengguna dalam database. Tidak seperti pada umumnya tabel di dalam basis data relasional, view bukanlah bagian dari skema fisik. View bersifat dinamis, ia mengandung data dari tabel yang direpresentasikannya. Dengan demikian, ketika tabel yang menjadi sumber datanya berubah, data di view juga akan berubah. Kegunaan view antara lain :

1. Memfokuskan pada data tertentu,
2. Penyederhanaan manipulasi data,
3. Menyesuaikan data dengan kebutuhan user,
4. Export dan impor data,
5. Mengkombinasikan data terpatisi.

```
CREATE VIEW view_name [(column[,..n])] [with encryption]
AS select_statement [with check option]
```

Contoh : Buatlah view untuk membuat daftar seluruh nama kota yang ada dalam tabel
PLAYERS!

```
CREATE VIEW TOWNS (TOWN) AS SELECT DISTINCT TOWN
FROM PLAYERS
```

2.1.1 Updatable View

View dapat berisi read-only atau updatable. Kondisi ini sangat dipengaruhi oleh adanya pendefinisian view itu sendiri. Bagaimanapun, untuk menciptakan updatable view, pernyataan SELECT yang didefinisikan di view harus mengikuti aturan-aturan sebagai berikut :

- Pernyataan `SELECT` tidak boleh merujuk kelebih dari satu tabel

- Pernyataan `SELECT` tidak boleh menggunakan klausa `GROUP BY` atau `HAVING`.
- Pernyataan `SELECT` harus tidak menggunakan `DISTINCT`.
- Pernyataan `SELECT` harus tidak merujuk ke view lain yang tidak updatable.
- Pernyataan `SELECT` tidak boleh mengandung ekspresi apa pun, misalnya fungsi agregat.

Pada hakikatnya, jika sistem database mampu menentukan pemetaan balik dari skema view ke skema tabel dasar, maka view memungkinkan untuk di update. Dalam kondisi ini, operasi-operasi INSERT, UPDATE dan DELETE dapat diterapkan pada view.

2.1.2 Hapus View

Suatu view dari query selalu menampilkan data yang terbaru, sebagai contoh bila kita memodifikasi sebagian tupel dalam tabel dasarnya dimana view tersebut didefinisikan maka secara otomatis akan berpengaruh pada view di query. Jika tidak membutuhkan view lagi, kita bisa menggunakan perintah :

Syntax

```
DROP VIEW view_name
```

BAB III
TUGAS PENDAHULUAN

3.1 Soal

1. Apa itu view dalam konteks database dan Apa kegunaannya?
2. jelaskan query pembuatan view dalam sebuah database!
3. Apa perbedaan antara view dan table dalam database?
4. Mengapa penggunaan view dapat membantu dalam manajemen data dan pengembangan aplikasi database?
5. Apa saja keuntungan dan kerugian menggunakan view dalam pengembangan Aplikasi database?

3.2 jawaban

1. View adalah table virtual yang terbuat dari suatu query terhadap satu table atau beberapa table. View digunakan untuk menyederhanakan dan mempermudah persepsi pengguna database.
2. CREATE VIEW view_name AS
   SELECT column1, column2, ...
   FROM table_name
   Pada query diatas view_name merupakan nama view yang akan dibuat. column1 dan column2 merupakan kolom yang akan ditampilkan didalam view. table_name adalah nama table yang digunakan didalam view.
3. Table dapat menyimpan data fisik sedangkan view tidak, hanya menampilkan data. table memiliki struktur tetap sedangkan view dinamis, mengandung data dari table yang direpresentasikan.
4. karena melalui view dapat meningkatkan keamanan data dengan hanya menampilkan beberapa kolom saja. mengembangkan kompleksitas data dan mudah bagi pengguna mengolah data.
5. keuntungan :
   1. mengurangi kompleksitas data
   2. meningkatkan keamanan data
   3. mengurangi penggunaan proses join
   4. meningkatkan kinerja query
   5. mengurangi penggunaan proses map

   kerugian :
   1. ketika table dihilangkan, tampilan menjadi tidak aktif
   2. Tidak dapat menggunakan operasi DML

BAB IV
IMPLEMENTASI

4.1 Soal

1. Definisikan view untuk menampilkan nama pelanggan,total,dan tanggal pesanan untuk semua pesenan yang memiliki total pesanan lebih dari rata-rata total pesanan.
2. Buatlah view yang menampilkan nama produk,harga satuan ,jumlah produk terjual,dan total pendapatan untuk setiap produk yang telah terjual dalam setiap pesanan.
3. Definisikan view untuk menampilkan nama produk dan jumlah stok yang tersisa untuk produk-produk yang memiliki stok kurang dari 5.
4. Buatlah view yang menampilkan nama pelanaggan dan jumlah total pesanan yang telah dilakukan oleh setiap pelanggan dalam satu bulan terakhir.

4.2 Query

1. Membuat database

```
CREATE DATABASE db_pembelian;
USE db_pembelian;
```

Penjelasan: Step yang pertama membuat query database dan juga nama database

2.
```
CREATE TABLE tb_produk (
    id_produk INT (11) NOT NULL PRIMARY KEY,
    nama_produk VARCHAR (100) NOT NULL,
    harga INT (11) NOT NULL,
    stock INT (11) NOT NULL );
```

Penjelasan : Membuat tabel "tb_produk" dengan 4 kolom id_produk, nama_produk, harga, dan stok.

3.
```
CREATE TABLE tb_pelanggan(
    id_pelanggan INT (11) NOT NULL PRIMARY KEY,
    nama_pelanggan VARCHAR (100) NOT NULL,
    email VARCHAR (50) NOT NULL,
    alamat VARCHAR (255) NOT NULL
    );
```

Penjelasan: Membuat tabel "tb_pelanggan" dengan 4 kolom id_pelanggan, nama_pelanggan, email, dan alamat.

4.
```
CREATE TABLE tb_pesanan(
        id_pesanan INT (11) NOT NULL PRIMARY KEY,
    id_pelanggan INT (11) NOT NULL,
    tanggal_pesanan DATE NOT NULL,
    total INT (11) NOT NULL,
    FOREIGN KEY (id_pelanggan) REFERENCES tb_pelanggan
(id_pelanggan)
    );
```

Penjelasan: Membuat tabel "tb_pesanan" dengan 4 kolom id_pesanan, nama_pelanggan, tanggal_pesanan, dan total.

5.
```
CREATE TABLE tb_detail_pesanan(
    id_detail INT (11) NOT NULL PRIMARY KEY,
    id_pesanan INT (11) NOT NULL,
    id_produk INT (11) NOT NULL,
    jumlah INT (11) NOT NULL,
    FOREIGN KEY (id_pesanan) REFERENCES tb_pesanan(id_pesanan),
    FOREIGN KEY (id_produk) REFERENCES tb_produk(id_produk)
    );
```

Penjelasan: Membuat tabel "tb_detail_pesanan" dengan 4 kolom id_detail, id_pesanan, id_produk, dan jumlah.

6.
```
INSERT INTO tb_produk (id_produk, nama_produk, harga, stock)
VALUES
    (1, 'Kaos', 10000, 5),
    (2, 'Kemeja', 15000, 8),
    (3, 'Pollo', 20000, 7),
    (4, 'Celana Pendek', 25000, 6),
    (5, 'Celana Panjang', 30000, 9),
    (6, 'Celana Jeans', 35000, 4),
    (7, 'Celana Kargo', 40000, 3);
```

Penjelasan: Masukkan data contoh ke dalam tabel "tb_produk".

7.
```
INSERT INTO tb_pelanggan (id_pelanggan, nama_pelanggan, email, alamat)
VALUES
(1, 'Budi Santoso', 'budi.santoso@example.com', 'Jl. Merdeka No. 10, Jakarta'),
(2, 'Ani Wijaya', 'ani.wijaya@example.com', 'Jl. Pahlawan No. 20, Surabaya'),
(3, 'Agus Surya', 'agus.surya@example.com', 'Jl. Diponegoro No. 30, Bandung'),
(4, 'Dewi Indah', 'dewi.indah@example.com', 'Jl. A. Yani No. 40, Yogyakarta'),
(5, 'Eko Pratama', 'eko.pratama@example.com', 'Jl. Gajah Mada No. 50, Semarang');
```

Penjelasan: Masukkan data contoh ke dalam tabel "tb_pelanggan".

8.
```
INSERT INTO tb_pesanan (id_pesanan, id_pelanggan, tanggal_pesanan, total)
VALUES
(1, 1, '2024-03-01', 500000),
(2, 2, '2024-03-02', 700000),
(3, 3, '2024-03-03', 300000),
(4, 4, '2024-03-04', 900000),
(5, 5, '2024-03-05', 600000),
(6, 1, '2024-03-06', 800000),
(7, 2, '2024-03-07', 400000),
(8, 3, '2024-03-08', 1000000),
(9, 4, '2024-03-09', 450000),
(10, 5, '2024-03-10', 850000);
```

Penjelasan: Masukkan data contoh ke dalam tabel "tb_pesanan".

9. Mengisi data detail pesanan

```
INSERT INTO tb_detail_pesanan (id_detail, id_pesanan, id_produk, jumlah)
VALUES
(1, 1, 1, 5),
(2, 1, 2, 3),
(3, 2, 3, 2),
(4, 2, 1, 4),
(5, 3, 2, 6),
(6, 3, 3, 2),
(7, 4, 1, 3),
(8, 4, 2, 5),
(9, 5, 3, 4),
(10, 5, 1, 2),
(11, 6, 2, 3),
(12, 6, 3, 3),
(13, 7, 1, 4),
(14, 7, 2, 4),
(15, 8, 3, 5),
(16, 8, 1, 6),
(17, 9, 2, 2),
(18, 9, 3, 3),
(19, 10, 1, 4),
(20, 10, 2, 4);
```

Penjelasan; Masukkan data contoh ke dalam tabel "tb_detail_pesanan".

10. Soal 1

```
CREATE VIEW view_total_atas_rata_rata AS
SELECT p.nama_pelanggan, pes.total, pes.tanggal_pesanan
FROM tb_pesanan pes
JOIN tb_pelanggan p ON pes.id_pelanggan = p.id_pelanggan
WHERE pes.total > (SELECT AVG(total) FROM tb_pesanan);
SELECT * FROM view_total_atas_rata_rata;
```

Penjelasan:Buat data sesuai table diatas untuk menampilkan view yang pertama.

11. Soal 2

```
CREATE VIEW view_produk_terjual AS

SELECT pr.nama_produk, pr.harga AS harga_satuan, dp.jumlah AS
jumlah_terjual, pr.harga * dp.jumlah AS total_pendapatan

FROM tb_detail_pesanan dp

JOIN tb_produk pr ON dp.id_produk = pr.id_produk;

SELECT * FROM view_produk_terjual;
```

Penjelasan: Buat data sesuai table diatas untuk menampilkan view yang kedua.

12. Soal 3

```
CREATE VIEW view_produk_stok_kurang_5 AS

SELECT nama_produk, stock AS jumlah_stok_tersisa

FROM tb_produk

WHERE stock < 5;

SELECT * FROM view_produk_stok_kurang_5;
```

Penjelasan: Buat data sesuai table diatas untuk menampilkan view yang ketiga.

13. Soal 4

```
CREATE VIEW view_total_pesanan_perpelanggan AS

SELECT p.nama_pelanggan, SUM(pes.total) AS total_pesanan

FROM tb_pesanan pes

JOIN tb_pelanggan p ON pes.id_pelanggan = p.id_pelanggan

WHERE pes.tanggal_pesanan >= DATE_SUB(CURDATE(), INTERVAL 1
MONTH)

GROUP BY p.nama_pelanggan;

SELECT * FROM view_total_pesanan_perpelanggan;
```

Penjelasan: Buat data sesuai table diatas untuk menampilkan view yang keempat.

4.3 Hasil

1. Lebih dari rata-rata total pesanan

nama_pelanggan	total	tanggal_pesanan
Budi Santoso	800000	2024-03-06
Ani Wijaya	700000	2024-03-02
Agus Surya	1000000	2024-03-08
Dewi Indah	900000	2024-03-04
Eko Pratama	850000	2024-03-10

2. Produk Terjual

nama_produk	harga_satuan	jumlah_terjual	total_pendapatan
Kaos	10000	5	50000
Kaos	10000	4	40000
Kaos	10000	3	30000
Kaos	10000	2	20000
Kaos	10000	4	40000
Kaos	10000	6	60000
Kaos	10000	4	40000
Kemeja	15000	3	45000
Kemeja	15000	6	90000
Kemeja	15000	5	75000
Kemeja	15000	3	45000
Kemeja	15000	4	60000
Kemeja	15000	2	30000
Kemeja	15000	4	60000
Pollo	20000	2	40000
Pollo	20000	2	40000
Pollo	20000	4	80000
Pollo	20000	3	60000
Pollo	20000	5	100000
Pollo	20000	3	60000

3. Stock produk yang kurang dari 5

nama_produk	jumlah_stok_tersisa
Celana Jeans	4
Celana Kargo	3

4. Jumlah pesanan dalam satu bulan

nama_pelanggan	total_pesanan
Agus Surya	1000000
Ani Wijaya	400000
Budi Santoso	800000
Dewi Indah	1350000
Eko Pratama	1450000

BAB V
PENUTUP

5.1 Analisa

Dari hasil praktikum, praktikan menganalisa bahwa View adalah perintah query yang disimpan pada database dengan suatu nama tertentu, sehingga bisa digunakan setiap saat untuk melihat data tanpa menuliskan ulang query tersebut.

Dari hasil implementasi praktikan mengalami sedikit kesulitan untuk pemahaman dari HAVING dan juga GROUP BY, tetapi praktikan dapat mempraktekanya dengan baik dan tanpa kendala. Selanjutnya praktikan juga sedikit kesulitan dalam menangani CONSTRAINT dan FOREIGN KEY, tetapi itu juga tidak menghambat berjalanya program dan masih bisa berjalan sebgai mana mestinya.

5.2 Kesimpulan

View dalam konteks basis data adalah sebuah mekanisme yang memungkinkan untuk membuat tabel virtual yang terdiri dari hasil query terhadap satu tabel atau beberapa tabel yang ada dalam database.

Dalam hal ini, view tidak memiliki representasi fisik yang terpisah dari tabel induknya, melainkan hanya terbentuk secara virtual dengan menggunakan hasil dari operasi query yang dilakukan terhadap tabel induk. Salah satu keuntungan dari penggunaan view adalah dapat mempermudah dan menyederhanakan cara penggunaan database.

Dengan menggunakan view, pengguna dapat melihat data dengan cara yang lebih spesifik dan terfokus hanya pada informasi yang mereka butuhkan tanpa perlu mengakses tabel asli yang mungkin terdiri dari banyak kolom dan baris. Selain itu, view juga dapat mengurangi kompleksitas query dan memungkinkan pengguna untuk mengakses data dengan cara yang lebih efisien.