**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**

## **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.

## **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**

## **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 |



### **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.

### **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**

# **BAB IV IMPLEMENTASI**

## **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.

## **Query**

1. Membuat database

CREATE DATABASE db\_pembelian;

USE db\_pembelian;

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

1. Membuat table produk

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.

1. Membuat table pelanggan

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.

1. Membuat table pesanan

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.

1. Membuat tabel detail pesanan

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.

1. Mengisi table produk

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".

1. Mengisi table pelanggan:

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');

Penjelelasan: Masukkan data contoh ke dalam tabel "tb\_pelanggan".

1. Mengisi table pesanan

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".

1. 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".

1. 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.

1. 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.

1. 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.

1. 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

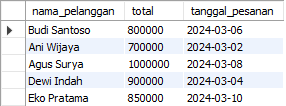
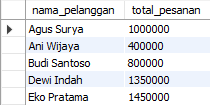
WHERE pes.tanggal\_pesanan >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH)

GROUP BY p.nama\_pelanggan;

SELECT \* FROM view\_total\_pesanan\_perpelanggan;

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

## **Hasil**

1. Lebih dari rata-rata total pesanan
2. Produk Terjual
3. Stock produk yang kurang dari 5
4. Jumlah pesanan dalam satu bulan

# **BAB V PENUTUP**

## **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.

## **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.