NAMA : PUTRI NABILA AMIR

KELAS : TI.19.D5

NIM : 311910077

SISTEM MANAJEMEN BASIS DATA

1. **Tipe-tipe data yang ada pada SQL Server.**
   a. **Tipe Data Angka (Numerik)**

   Tipe Data Angka (Numerik) merupakan tipe data yang dapat kita gunakan pada suatu variabel konstanta yang dapat menyimpan nilai berupa angka.

No	Nama	Fungsi	Jangkauan	Ukuran
1	TINYINT	Menyimpan data bilangan bulat positif dan negatif.	-128 s/d 127	1 byte (8 bit).
2	SMALLINT	Menyimpan data bilangan bulat positif dan negatif.	-32.768 s/d 32.767	2 byte (16 bit).
3	MEDIUMINT	Menyimpan data bilangan bulat positif dan negatif.	-8.388.608 s/d 8.388.607	Ukuran : 3 byte (24 bit).
4	INT	Menyimpan data bilangan bulat positif dan negatif.	-2.147.483.648 s/d 2.147.483.647	4 byte (32 bit).
5	BIGINT	Menyimpan data bilangan bulat positif dan negatif.	± 9,22 x 1018	8 byte (64 bit).
6	FLOAT	menyimpan data bilangan pecahan positif dan negatif presisi tunggal	-3.402823466E+38 s/d -1.175494351E-38, 0, dan 1.175494351E-38 s/d 3.402823466E+38.	4 byte (32 bit)
7	DOUBLE	menyimpan data bilangan pecahan positif dan negatif presisi ganda.	-1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308.	5 byte (64 bit).
8	REAL	menyimpan data bilangan pecahan positif dan negatif presisi ganda.	-1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308.	6 byte (64 bit).
9	DECIMAL	menyimpan data bilangan pecahan positif dan negatif.	-1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308.	7 byte (64 bit).
10	NUMERIC	menyimpan data bilangan pecahan positif dan negatif.	-1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308.	8 byte (64 bit).

b. Tipe Data Teks (String)

Tipe Data Teks (String) merupakan tipe data yang bisa kita gunakan untuk menampung banyak karakter dengan jumlah maksimum data yang dapat ditampung yakni sebanyak 255 karakter.

No	Nama	Fungsi	Jangkauan
1	CHAR	menyimpan data string ukuran tetap.	0 s/d 255 karakter
2	VARCHAR	menyimpan data string ukuran dinamis.	0 s/d 255 karakter (versi 4.1), 0 s/d 65.535
3	TINYTEXT	menyimpan data text.	1 s/d 255 karakter (versi 4.1), 0 s/d 65.535
4	TEXT	menyimpan data text.	0 s/d 65.535
5	MEDIUMTEXT	menyimpan data text.	0 s/d 224 - 1 karakter
6	LONGTEXT	menyimpan data text.	1 s/d 224 - 1 karakter

c. Tipe Data Date.

Tipe Data Date digunakan untuk menyimpan data tanggal dengan format tahun, bulan, tanggal.

No	Nama	Fungsi	Jangkauan	Ukuran
1	DATE	menyimpan data tanggal	1000-01-01 s/d 9999-12-31 (YYYY-MM-DD)	3 byte.
2	TIME	menyimpan data waktu	-838:59:59 s/d +838:59:59 (HH:MM:SS)	3 byte.
3	DATETIME	menyimpan data tanggal dan waktu.	1000-01-01 00:00:00' s/d '9999-12-31 23:59:59'	8 byte
4	YEAR	menyimpan data tahun dari tanggal	1900 s/d 2155	1 byte

d. Tipe Data BLOB.

Tipe Data BLOB merupakan tipe data yang dapat digunakan untuk menampung gambar, musik, video dan lain-lain nya.

No	Nama	Fungsi	Jangkauan
1	BIT	Menyimpan data biner.	64 digit biner
2	TINYBLOB	menyimpan data biner/ Gambar ukuran kecil	255 byte
3	BLOB	Menyimpan data biner/ Gambar	4
4	MEDIUMBLOB	Menyimpan data biner/ Gambar kuran sedang	224-1 byte
5	LONGBLOB	Menyimpan data biner/ Gambar ukuran besar	232- 1 byte

Contoh tipe data yang diterima di tabel :

a. Tipe Tabel MyISAM

Tipe tabel MyISAM merupakan tipe tabel yang sederhana, stabil dan mudah digunakan. Jika kita akan menyimpan data sederhana yang tidak terlalu rumit maka gunakanlah tipe tabel ini. Kelebihan utama MyISAM adalah kecepatan dan kestabilannya. Jika kita memilih tipe tabel MyISAM, maka MySQL secara otomatis akan menentukan salah satu dari tiga jenis tabel MyISAM, yaitu:

1. MyISAM Static.
   Jenis ini digunakan ketika semua kolom dalam tabel didefinisikan dengan ukuran yang pasti (fixed). Dengan kata lain, tidak ada kolom yang memiliki tipe seperti VARCHAR, Text, dan Blob. Karena sifatnya yang fixed, maka jenis ini akan lebih cepat, aman dan stabil.
2. MyISAM dynamic.
   Jenis ini digunakan ketika terdapat kolom dengan tipe yang dinamis, seperti tipe kolom Varchar. Keutungan utama dari jenis ini adalah ukuran yang dinamis. Jadi sifatnya lebih efektif karena ukuran data (file) menyesuaikan isi dari masing-masing kolom (field).
3. MyISAM Compressed.
   Kedua jenis MyISAM, static dan dynamic dapat dikompresi menjadi satu jenis yaitu MyISAM Compressed dengan perintah myisamchk. Tentu hasilnya lebih kecil dari segi ukuran. Tabel yang terkompresi tidak dapat dikenakan operasi seperti Insert, Update, dan Delete.

b. **Tipe Tabel InnoDB**

Tipe tabel InnoDB merupakan tipe tabel MySQL yang mendukung proses transaksi. Tipe ini memiliki beberapa keunggulan, yaitu:

1. Mendukung transaksi antar tabel.
2. Mendukung row-level-locking.
3. Mendukung Foreign-Key Constraints.
4. Crash recovery.

c. Tipe Tabel HEAP

Tipe tabel dengan tipe HEAP tidak menyimpan datanya di hardisk, tetap menyimpan di memori (RAM). Tipe tabel ini biasanya digunakan untuk tabel sementara (temporary). Tabel secara otomatis akan dihapus dari MySQL saat koneksi keserver diputus.

2. 15 Perintah utama yang ada pada SQL Server.

a. Perintah SELECT

Perintah SELECT merupakan perintah dasar SQL yang di gunakan untuk memilih data dari database. Data yang di kembalikan di simpan dalam tabel yang di sebut result-set.

Sintaks : `SELECT kolom1, kolom2, ... FROM nama_tabel;`
`SELECT * FROM nama_tabel;`

Contoh : `SELECT nis, nama, alamat FROM siswa;`
`SELECT * FROM siswa;`

Fungsinya : Perintah pertama di atas di gunakan untuk menyeleksi kolom NIS, NAMA, dan ALAMAT dari tabel SISWA. sedangkan perintah kedua di gunakan untuk menyeleksi semua kolim dari tabel SISWA.

b. Perintah SELECT DISTINCT

Perintah SELECT DISTINCT merupakan perintah dasar SQL yang di gunakan untuk mengembalikan hanya nilai yang berbeda dari dalam sebuah tabel, dengan kata lain semua record duplikat (record dengan nilai yang sama) yang terdapat pada tabel akan di anggap sebagai satu record/nilai.

Sintaks : `SELECT DISTINCT kolom1, kolom2, ... FROM nama_tabel;`

Contoh : `SELECT DISTINCT alamat FROM siswa;`

Fungsinya : Perintah di atas di gunakan untuk menampilkan kolom ALAMAT dari tabel SISWA dengan mengabaikan nilai yang duplikat, misalnya terdapat 10 siswa dengan alamat ‘Jakarta’, 15 siswa dengan alamat ‘Bandung’, 20 siswa dengan alamat ‘Depok’, dan seterusnya, result-set hanya menampilkan daftar alamat di antaranya ‘Jakarta’, ‘Bandung’, dan ‘Depok’ masing-masing 1 record.

c. Perintah WHERE

Perintah WHERE merupakan perintah dasar SQL yang di gunakan untuk mem-filter hasil SELECT dengan mengekstrak record yang memenuhi persyaratan tertentu.

Sintaks : `SELECT kolom1, kolom2, ... FROM nama_tabel WHERE kondisi;`

Contoh : `SELECT nis, nama FROM siswa WHERE alamat='jakarta';`

Fungsinya : Perintah di atas di gunakan untuk mengekstraksi data (NIS dan NAMA) dari tabel SISWA dengan kondisi “field ALAMAT berisi nilai JAKARTA”. Perintah di atas menggunakan operator sama dengan (‘=’), untuk operator lain yang di dukung perintah WHERE, lihat list di bawah ini.

- = Sama dengan
- <> Tidak sama dengan,
  pada beberapa versi SQL, operator yang di gunakan adalah !=

- > Lebih besar dari
- < Lebih kecil dari
- >= Lebih besar sama dengan
- <= Lebih kecil sama dengan
- BETWEEN Antara rentang inklusif
- LIKE Cari pola yang 'seperti' parameter
- IN Menentukan kemungkinan nilai dari beberapa kolom

d. Perintah (operator) AND, OR dan NOT

Operator AND, OR dan NOT merupakan perintah dasar SQL yang biasanya di kombinasikan dengan perintah WHERE. Ketiganya di gunakan untuk mem-filter record berdasarkan suatu kondisi, operator AND akan menampilkan record apabila semua kondisi bernilai TRUE, operator OR akan menampilkan record apabila salah satu kondisi bernilai TRUE, sedangkan operator NOT akan menampilkan record apabila semua kondisi bernilai FALSE.

Sintaks AND : `SELECT kolom1, kolom2, ... FROM nama_tabel WHERE kondisi1 AND kondisi2 AND kondisi3 ...;`

Contoh AND : `SELECT nis, nama FROM siswa WHERE alamat='Jakarta' AND tahun_lahir='2000'`

Fungsinya : Perintah di atas akan menampilkan record NIS dan NAMA dari tabel SISWA dengan ALAMAT di JAKARTA dan TAHUN_LAHIR "2000", artinya record siswa yang lahir di tahun "2000" namun tidak beralamat di "Jakarta: atau siswa yang beralamat di "Jakarta" namaun lahir bukan pada tahun "2000" tidak akan di tampilkan.

Sintaks OR : `SELECT kolom1, kolom2, ... FROM nama_tabel WHERE kondisi1 OR kondisi2 OR kondisi3 ...;`

Contoh OR : `SELECT nis, nama FROM siswa WHERE alamat='Jakarta' OR alamat='Bandung'`

Fungsinya : Perintah di atas akan menampilkan record NIS dan NAMA dari tabel SISWA dengan ALAMAT di "JAKARTA" atau di "Bandung", artinya record siswa yang beralamat di "Jakarta" dan di "Bandung" saja yang akan di tampilkan.

Sintaks NOT : `SELECT kolom1, kolom2, ... FROM nama_tabel WHERE NOT kondisi;`

Contoh NOT : `SELECT nis, nama FROM siswa WHERE NOT alamat='Jakarta'`

Fungsinya : Perintah di atas akan menampilkan semua record NIS dan NAMA dari tabel SISWA kecuali record siswa yang beralamat di 'Jakarta

e. **Perintah ORDER BY**

Perintah ORDER BY merupakan perintah dasar SQL yang di gunakan untuk mengurutkan result-set dalam pengurutan 'ascending' atau 'descending'. Secara default perintah ORDER BY menampilkan record dalam pengurutan 'ascending' ('ASC'). Untuk mengurutkan 'descending', gunakan kata kunci 'DESC'.

Sintaks : `SELECT kolom1, kolom2, ... FROM nama_tabel ORDER BY column DESC;`

Contoh : `SELECT nis, nama FROM siswa ORDER BY tahun_lahir DESC;`

Fungsinya : Perintah di atas akan menampilkan result-set berupa field NIS dan NAMA dari tabel SISWA dengan di urutkan berdasarkan TAHUN_LAHIR secara descending, artinya tahun lahir akan di tampilkan mulai dari yang terbesar (siswa termuda) hingga terkecil (siswa termuda).

f. **Perintah INSERT INTO**

Dalam SQL, perintah INSERT INTO merupakan perintah dasar SQL bagian dari perintah untuk DML (Data Manipulation Language) Saya asumsikan Anda telah faham perbedaan DDL, DCL, dan DML. Perintah INSERT INTO dapat di gunakan untuk menambahkan record baru ke dalam tabel.

Sintaks : `INSERT INTO nama_tabel VALUES (nilai1, nilai2, nilai3, ...);`

Contoh : `INSERT INTO siswa VALUES ('12345', 'Abdul', 'Jakarta');`

Fungsinya : Perintah di atas di gunakan untuk menambahkan nilai '12345', 'Abdul' dan 'Jakarta' pada tabel SISWA, pastikan urutan nilai ('values') dalam urutan yang sama seperti kolom dalam tabel. Jika urutan nilai tidak sama dengan urutan kolom pada tabel, maka sintaks INSERT INTO yang di gunakan adalah sebagai berikut:

Sintaks : `INSERT INTO nama_tabel (kolom1, kolom2) VALUES (nilai1, nilai2);`

Contoh : `INSERT INTO siswa (nim, nama) VALUES ('12345', 'Abdul');`

Fungsinya : Perintah di atas di gunakan untuk menambahkan nilai '12345' pada kolom NIM, dan nilai 'Abdul' pada kolom NAMA dengan mengabaikan kolom lain yang tidak di isi, misalnya kolom ALAMAT

g. **Perintah UPDATE**

Perintah UPDATE merupakan perintah dasar SQL yang di gunakan untuk memperbarui atau mengubah nilai suatu record berdasarkan kriteria tertentu.

Sintaks : `UPDATE nama_tabel SET kolom1 = nilai1, kolom2 = nilai2, ... WHERE kondisi;`

Contoh : `UPDATE siswa SET nama = 'Ahmad', alamat = 'Bandung' WHERE nim = '12345';`

Fungsinya : Perintah di atas di gunakan untuk memperbarui kolom NAMA menjadi ‘Ahmad’ dan kolom ALAMAT menjadi ‘Jakarta’ pada record dengan NIM ‘12345’ (ingat sebelumnya NIM ‘12345’ di gunakan oleh siswa bernama ‘Abdul’ yang beralamat di ‘Jakarta’, sebut saja data tersebut keliru dan harus di perbarui).

h. Perintah DELETE

Hampir sama dengan perintah UPDATE, perintah DELETE juga merupakan perintah dasar SQL yang di gunakan untuk menghapus nilai suatu record berdasarkan kriteria tertentu.

Sintaks : `DELETE FROM table_name WHERE condition;`

Contoh : `DELETE FROM siswa WHERE nim = '12345';`

Fungsinya : Perintah di atas di gunakan untuk menghapus record dengan NIM ‘12345’, ingat bahwa NIM tersebut di gunakan oleh siswa bernama ‘Ahmad’ (sebelumnya bernama ‘Abdul’) dan dengan di eksekusinya perintah DELETE ini maka record tersebut akan terhapus.

i. Perintah (fungsi) MIN()

Fungsi MIN() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan nilai terkecil dari suatu kolom, Anda dapat menerapkannya pada kolom ‘harga’, ‘nilai’, ‘qty’ atau kolom yang semisal dengan itu, berbeda dengan perintah ORDEY BY, fungsi MIN() hanya menampilkan satu record saja yang memenuhi kriteria yang Anda tentukan.

Sintaks : `SELECT MIN(nama_kolom) FROM nama_tabel WHERE kondisi;`

Contoh : `SELECT MIN(harga) FROM barang WHERE kategori='atk';`

Perintah di atas di gunakan untuk mencari nilai terendah dari kolom ‘harga’ di tabel BARANG dengan KATEGORI ‘atk’ (Alat Tulis dan Kertas).

j. Perintah (fungsi) MAX()

Fungsi MAX() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan nilai terbesar dari suatu kolom, seperti halnya fungsi MIN() Anda dapat menerapkannya pada kolom ‘harga’, ‘nilai’, ‘qty’ atau kolom yang semisal dengan itu.

Sintaks : `SELECT MAX(nama_kolom) FROM nama_tabel WHERE kondisi;`

Contoh : `SELECT MAX(harga) FROM barang WHERE kategori='atk';`

Fungsinya : Perintah di atas di gunakan untuk mencari nilai tertinggi dari kolom ‘harga’ di tabel BARANG dengan KATEGORI ‘atk’ (Alat Tulis dan Kertas).

k. Perintah (fungsi) COUNT()

Fungsi COUNT() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan jumlah hitungan record yang memenuhi suatu kriteria.

Sintaks : `SELECT COUNT(nama_kolom) FROM nama_tabel WHERE kondisi;`

Contoh : `SELECT COUNT(id) FROM barang WHERE kategori='atk';`

Fungsinya : Perintah di atas di gunakan untuk mencari tahu jumlah hitungan record pada tabel BARANG yang memenuhi kriteria/kondisi yaitu record dengan KATEGORI 'atk' (Alat Tulis dan Kertas).

l. Perintah (fungsi) AVG()

Fungsi AVG() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan rata-rata record yang memenuhi suatu kriteria, tentunya nilai pada kolom harus numerik.

Sintaks : `SELECT AVG(nama_kolom) FROM nama_tabel WHERE kondisi;`

Contoh : `SELECT AVG(harga) FROM barang WHERE kategori='atk';`

Fungsinya : Perintah di atas di gunakan untuk mencari tahu HARGA rata-rata pada tabel BARANG yang memenuhi kriteria/kondisi yaitu record dengan KATEGORI 'atk' (Alat Tulis dan Kertas).

j. **Perintah (fungsi) SUM()**

Fungsi SUM() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan jumlah record yang memenuhi suatu kriteria, tentunya nilai pada kolom harus numerik.

Sintaks : `SELECT SUM(nama_kolom) FROM nama_tabel WHERE kondisi;`

Contoh : `SELECT SUM(qty) FROM barang WHERE kategori='atk';`

Fungsinya : Perintah di atas di gunakan untuk mencari tahu jumlah kuantitas ('qty') pada tabel BARANG yang memenuhi kriteria/kondisi yaitu record dengan KATEGORI 'atk' (Alat Tulis dan Kertas).

k. **Perintah INNER JOIN**

INNER JOIN merupakan perintah dasar SQL yang di gunakan untuk menggabungkan beberapa tabel dan mengambil nilai yang cocok (identik) di antara kedua tabel tersebut.

Sintaks : `SELECT nama_kolom1, nama_kolom2, ... FROM tabel1 INNER JOIN tabel2 ON tabel1.nama_kolom = tabel2.nama_kolom;`

Contoh : `SELECT pesanan.id_pesanan, pelanggan.nama FROM pesanan INNER JOIN pelanggan ON pesanan.id_pelanggan = pelanggan.id_pelanggan;`

Fungsinya : Perintah di atas akan menampilkan kolom ID_PESANAN dari tabel PESANAN dan kolom NAMA dari tabel PELANGGAN yang memenuhi kriteria yaitu ID_PELANGGAN pada tabel PESANAN yang sama dengan ID_PELANGGAN pada tabel PELANGGAN.

l. Perintah LEFT JOIN

LEFT JOIN merupakan perintah dasar SQL yang di gunakan untuk menggabungkan beberapa tabel dan mengambil nilai yang cocok (identik) di antara kedua tabel tersebut dan nilai lain dari tabel pada ruas kiri meskipun tak ada nilai yang cocok dengan tabel pada ruas kanan.

Sintaks : `SELECT nama_kolom1, nama_kolom2, ... FROM tabel1 LEFT JOIN tabel2 ON tabel1.nama_kolom = tabel2.nama_kolom;`

Contoh : `SELECT pelanggan.nama, pesanan.barang_pesanan FROM pelanggan LEFT JOIN pesanan ON pesanan.id_pelanggan = pelanggan.id_pelanggan;`

Fungsinya : Perintah di atas akan menampilkan semua nilai kolom NAMA dari tabel PELANGGAN, yang memenuhi kriteria yaitu ID_PELANGGAN pada tabel PESANAN yang sama dengan ID_PELANGGAN pada tabel PELANGGAN.