Pemrosesan Query

Bahasa Basis Data

- Berdasarkan fungsinya, bahasa basis data dapat dipilah ke dalam 3 (tiga) bentuk yaitu :
- 1. *Data Definition Language (DDL)*
- 2. *Data Manipulation Language (DML)*
- 3. *Data Control Language (DCL)*

DDL

- Struktur / skema basis data yang menggambarkan / mewakili desain basis data secara keseluruhan dispesifikasikan dengan bahasa khusus yaituDDL.
- Dengan bahasa ini kita dapat membuat tabel (*create table) baru, indeks,* mengubah *table, menentukan struktur penyimpanan table, dan lainnya.*
- *Hasil* dari kompilasi perintah DDL, adalah kumpulan table yang disimpan dalam *file*khusus yang disebut kamus data (*data dictionary).*

DDL

- Contoh:
- • CREATE (untuk membentuk basis data, table atau index)
- • DROP (untuk menghapus basis data, table atau index)
- • ALTER (untuk mengubah struktur table)

DDL

- Contoh:
- CREATE DATABASE [nama database]

- CREATE DATABASE perpustakaan (Membuat database perpustakaan)
- DROP DATABASE perpustakaan (Menghapus database perpustakaan)
- DROP TABLE person (Menghapus tabel person)
- ALTER DATABASE perpustakaan MODIFY NAME=akademik (Merubah nama database perpustakaan menjadi akademik)

Membuat Table

- CREATE TABLE *table_name*
  (
  column_name1 data_type(*size*),
  column_name2 data_type(*size*),
  column_name3 data_type(*size*),
  ....
  );

- CREATE TABLE Persons
  (
  PersonID int,
  LastName varchar(255),
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
  );

SQL Constraint

- SQL Constraint digunakan untuk menambahkan aturan aturan dalam tabel.
- CREATE TABLE *table_name*
  (
  column_name1 data_type(*size*) *constraint_name*,
  column_name2 data_type(*size*) *constraint_name*,
  column_name3 data_type(*size*) *constraint_name*,
  ....
  );

- **NOT NULL** - Kolom harus diisi (tidak boleh dikosongkan)
- **UNIQUE** - Memastikan bahwa kolom harus memiliki nilai unik .
- **PRIMARY KEY** - Kombinasi dari not null dan unique (Contoh adalah Nim)
- **FOREIGN KEY** - Kunci Tetangga .
- **CHECK** - Memastikan agar isi dalam kolom harus sesuai dengan type data dan panjang data.
- **DEFAULT** - Memberikan nilai default kepada kolom.

Not Null

- CREATE TABLE PersonsNotNull
  (
  P_Id int NOT NULL,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
)

Unique

- CREATE TABLE PersonsUnique
  (
  P_Id int NOT NULL UNIQUE,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
)

Primary Key

- CREATE TABLE Persons
  (
  P_Id int NOT NULL PRIMARY KEY,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
)

Check

- CREATE TABLE Persons
  (
  P_Id int NOT NULL,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255),
  CHECK (P_Id>0)
)

Default

- CREATE TABLE Persons
  (
  P_Id int NOT NULL,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255) DEFAULT 'Sandnes'
)

ALTER TABLE

- ALTER TABLE table_name
  ADD column_name datatype
- ALTER TABLE table_name
  DROP COLUMN column_name
- ALTER TABLE table_name
  ALTER COLUMN column_name datatype
- sp_rename mahasiswa,mahasiswa2 (
  merubah nama table mahasiswa menjadi
  mahasiswa 2)

DML

- Bentuk bahasa basis data untuk melakukan menipulasi dan pengambilan data pada suatu basis data.
- Manipulasi data pada dabase dapat berupa :

1). Penyisipan / penambahan data pada *file / table dalam suatu basis data.*

2). Penghapusan data pada *file / table dalam suatu basis data.*

3). Pengubahan data pada *file / table dalam suatu basis data.*

4). Penelusuran data pada *file / table dalam suatu basis data.*

Query

- Query Adalah pernyataan yang diajukan untuk mengambil informasi di dalam suatu basis data.
- Query merupakan bagian dari DML untuk pengambilan informasi yang ada pada basis data.

SELECT

- **SELECT *column_name,column_name* FROM *table_name*;**
- **SELECT * FROM *table_name*;**

Contoh 1

SELECT ContactName,city FROM Customers; (Menampilkan kolom ContactName dan Kolom City dari table Customers)

Contoh 2

SELECT * FROM Customers; (Menampilkan seluruh informasi pada tabel customers)

SELECT DISTINCT

- Dengan SELECT DISTINCT akan menghasilkan informasi yang tidak duplikat.
- **SELECT DISTINCT *column_name,column_name* FROM *table_name***;
- Contoh
- **SELECT DISTINCT City FROM Customers**;

SQL WHERE

- Klausa WHERE digunakan untuk mengekstrak hanya informasi yang memenuhi kriteria tertentu.
- **SELECT *column_name,column_name***
  FROM *table_name*
  WHERE *column_name operator value*;

- Contoh 2
- **SELECT * FROM Customers**
  WHERE Country='Mexico';
- Contoh 2
- **SELECT * FROM Produk**
  WHERE Kode_barang=34;

Operator

Operator	Penjelasan
=	Sama dengan
<>	Tidak Sama dengan
>	Lebih Besar Dari
<	Kurang Dari
>=	Lebih besar atau sama dengan
<=	Kurang dari atau sama dengan
BETWEEN	Antara (rentang)
LIKE	Seperti
IN	Didalam

AND - OR

- Operator AND menampilkan informasi jika kedua kondisi pertama dan kondisi kedua adalah benar.
- Operator OR menampilkan informasi jika salah satu kondisi pertama atau kondisi kedua adalah benar.

- **SELECT * FROM Customers**
  WHERE Country='Germany'
  AND City='Berlin';

- **SELECT * FROM Customers**
  WHERE City='Berlin'
  OR City='München';

Kombinasi AND dan OR

- **SELECT * FROM Customers WHERE Country='Germany' AND (City='Berlin' OR City='München');**

SQL Order By

- ORDER BY digunakan untuk melakukan Sorting atau pengurutan hasil query .
- **SELECT *column_name,column_name***
  FROM *table_name*
  ORDER BY *column_name,column_name* ASC|DESC;
- ASC – Urut dari Nilai Terkecil sampai terbesar (A-z atau 1-100)
- DESC – Urut dari nilai terbesar sampai terkecil (Z-A atau 100-1)
- Contoh
- **SELECT * FROM Customers**
  ORDER BY Country;
- **SELECT * FROM Customers**
  ORDER BY Country DESC;
- **SELECT * FROM Customers**
  ORDER BY Country,ContactName;

SELECT TOP

- **SELECT**
  TOP *number*|*percent column_name*(*s*)
  FROM *table_name*;
- Contoh
- **SELECT TOP 10 * FROM Customers;**
- **SELECT TOP 50 PERCENT * FROM Customers;**

SQL LIKE

- Operator LIKE digunakan dalam klausa WHERE untuk mencari pola yang telah ditentukan dalam kolom.
- **SELECT *column_name(s)***
  FROM *table_name*
  WHERE *column_name* LIKE *pattern*;
- **Contoh**
- **SELECT * FROM Customers**
  WHERE City LIKE 's%'; (City dengan awalan S)
- **SELECT * FROM Customers**
  WHERE City LIKE '%s'; (City dengan akhiran S)
- **SELECT * FROM Customers**
  WHERE Country LIKE '%land%'; (Country yang mengandung land)
- **SELECT * FROM Customers**
  WHERE Country NOT LIKE '%land%'; (Country yang tidak mengandung Land)

SQL WILDCHARD

- Karakter wildcard dapat digunakan untuk menggantikan karakter lain (s) dalam sebuah

Wildcard	Description
%	A substitute for zero or more characters
_	A substitute for a single character
[*charlist*]	Sets and ranges of characters to match
[^*charlist*] or [!*charlist*]	Matches only a character NOT specified within the brackets

- **SELECT * FROM Customers WHERE City LIKE '_erlin';**
- **SELECT * FROM Customers WHERE City LIKE 'L_n_on';**
- **SELECT * FROM Customers WHERE City LIKE '[bsp]%';(Menampilkan seluruh customers yang memiliki kota berawalan (b atau s atau p).**
- **SELECT * FROM Customers WHERE City LIKE '[a-e]%'; (Menampilkan seluruh customers yang memiliki kota berawalan a ,b,c, d atau e)**
- **SELECT * FROM Customers WHERE City LIKE '[!bsp]%'; (Menampilkan seluruh customers yang memiliki kota tidak berawalan (b atau s atau p).**

IN

- **Operator IN memungkinkan Anda untuk menentukan beberapa nilai dalam klausa WHERE.**
- **SELECT *column_name(s)***
  FROM *table_name*
  WHERE *column_name* IN
  (*value1*,*value2*,...);
- Contoh
- **SELECT * FROM Customers**
  WHERE City IN ('Paris','London');

BETWEEN

- Operator BETWEEN menampilkan nilai-nilai dalam jangkauan. Nilai-nilai dapat berupa angka, teks, ataupun tanggal.
- **SELECT *column_name(s)***
  FROM *table_name*
  WHERE *column_name* BETWEEN *value1* AND *value2;*
- *Contoh*
- **SELECT * FROM Products**
- **WHERE UnitPrice BETWEEN 10 AND 20;**

SQL ALIAS

- SQL Alias digunakan untuk mengubah nama sementara pada tabel atau heading kolom.
- SELECT *column_name* AS *alias_name*
  FROM *table_name;*
- SELECT *column_name(s)*
  FROM *table_name* AS *alias_name;*
- **SELECT CustomerName AS Customer, ContactName AS [Contact Person]**
  FROM Customers;
- **SELECT CustomerName, Address+', '+City+', '+PostalCode+', '+Country AS Address**
  FROM Customers;

AGREGASI

- Agregasi dalam SQL merupakan proses untuk mendapatkan nilai dari sekumpulan data yang telah dikelompokkan.
- Pengelompokan data didasarkan pada kolom atau kombinasi kolom yang dipilih

Beberapa fungsi untuk agregasi adalah:

1. MAX : mencari data terbesar dari sekelompok data
2. MIN : mencari data terkecil dari sekelompok data
3. COUNT : mencari cacah data (data NULL tidak akan dimasukkan dalam perhitungan,kecuali disebutkan secara khusus)
4. SUM : mencari jumlah dari sekumpulan data numeris
5. AVG : mencari rerata dari sekumpulan data numeris

- **SELECT MAX(UnitPrice) AS HighestPrice FROM Products;**
- **SELECT COUNT(CustomerID) AS OrdersFromALFKI FROM Orders WHERE CustomerID='ALFKI';**
- **SELECT COUNT(*) AS NumberOfOrders FROM Orders;**
- **SELECT COUNT(DISTINCT CustomerID) AS NumberOfCustomers FROM Orders;**
- **SELECT SUM(Quantity) AS TotalItemsOrdered FROM [Order Details];**
- **SELECT ProductName, UnitPrice FROM Products**
- **WHERE UnitPrice>(SELECT AVG(UnitPrice) FROM Products);**

NPM	KodePro	NamaMahasiswa	TempatLahir	TanggalLahir	Alamat
08110167	11	Andi	Jakarta	12/03/1988	Gunung Kidul
08110231	11	Joko	Sleman	01/02/1989	Sleman
08210232	21	Budi	Bantul	15/09/1988	Bantul
08210233	21	Cici	Purwokerto	21/02/1989	Bantul
08310234	31	Didi	Bandung	11/07/1987	Kodya
08320235	32	Alfin	Makassar	22/09/1986	Kodya
08320236	32	Dodi	Gunung Kidul	24/03/1979	Kodya
08320237	32	Derri	Pangkal Pinang	09/02/1984	Sleman
08410121	41	Dudung	Kebumen	25/02/1985	Sleman
08410122	41	Afgan	Palembang	21/11/986	Kulon Progo
08420123	42	Didi	Kutoarjo	11/09/1986	Kulon Progo
08430124	43	Firza	Purworejo	11/09/1986	Bantul
08440125	44	Zahir	Temom	11/09/1986	Kulon Progo

KodeFakultas	NamaFakultas	NamaDekan
1	Teknik	Ahmad Riyadi
2	Pertanian	Paiman
3	Ekonomi	Sukhemi
4	Keguruan	Suharni

KodeProdi	KodeFakultas	NamaProdi	NamaKetuaProdi
11	1	Teknik Informatika	Bachtiar Dwi Effendi
21	2	Agroteknologi	Bahrum
31	3	Manajemen	Vita
32	3	Akintansi	Siti Maisaroh
41	4	PPKN	Sigit
42	4	Sejarah	Gunawan
43	4	Pendidikan Matematika	Tri
44	4	Bimbingan Konseling	Siswanti
45	4	PGSD	Haniek

Buatlah Perintah SQL

1. Menampilkan seluruh field pada tabel Fakultas
2. Menampilkan seluruh field pada tabel Prodi
3. Tampilkan seluruh field pada tabel Mahasiswa
4. Tampilkan Nama dan Alamat pada Tabel Mahasiswa.
5. Tampilkan NamaFakultas dan Dekan pada tabel Fakultas
6. Tampilkan Namaprodi saja pada tabel Prodi
7. Tampilkan KodeProdi dan NamaProdi pada tabel Prodi
8. Tampilkan semua isi field tabel Mahasiswa yang tinggal di Bantul
9. Tampilkan semua isi field pada tabel Prodi yang kode Fakultasnya = 4
10. Tampilkan NamaProdi dan KetuaProdi dimana KodeFakultas != 2 (dari tabel prodi)
11. Tampilkan semua isi field 10 Mahasiswa saja (Urut berdasarkan tempat lahir Descending (z-a)) (dari tabel mahasiswa)
12. Tampilkan semua isi field pada tabel fakultas dimana kode prodi antara 30 dan 42 .

1. select *from fakultas
2. select *from prodi
3. select *from mahasiswa
4. select nama,alamat from mahasiswa
5. select namaFakultas,dekan from fakultas
6. select namaProdi frm prodi
7. select kodeProdi , namaProdi from prodi
8. select *from mahasiswa where city='bantul'
9. select *from prodi where kodeFakultas=4
10. select namaprodi,ketuaprodi from prodi where kodefakultas<>2
11. Select TOP 10 *From mahasiswa order by alamat DESC
12. Select *from fakultas where kodefakultas between 30 and 42