



Microsoft®
SQL Server®

MODUL PRAKTIKUM SISTEM MANAJEMEN BASIS DATA

NURFAIZAH, M.Kom., TRI ASTUTI,
S.Kom, M.Eng

UNIVERSITAS AMIKOM PURWOKERTO
FAKULTAS ILMU KOMPUTER
PROGRAM STUDI SISTEM INFORMASI

PRAKTIKUM 1

A. BASIS DATA

Basis data merupakan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudancy) yang tidak perlu digunakan untuk berbagai kebutuhan. Basis data bisa dibayangkan sebagai lemari arsip yang memiliki prinsip kerja dan tujuan yang sama yaitu pengaturan data/arsip. Tujuan utama yang akan dicapai yaitu kemudahan dan kecepatan dalam pengambilan data/arsip.

B. SQL SERVER 2008

Microsoft SQL Server diperkenalkan pada tahun 1990 untuk platform Microsoft OS/2 dalam kerjasamanya dengan Sybase. Produk ini berasal dari Sybase SQL Server 4.x untuk platform Unix. Dengan adanya Windows NT, muncul inisiatif untuk membangun SQL Server versi Windows NT. Kerjasama dengan Sybase masih berlanjut dan diluncurkan SQL Server 6.0 pada tahun 1995 dan setahun kemudian SQL Server versi 6.5 diluncurkan. SQL Server 6.5 memperbarui kemampuan transaksi dan menjadi produk database client/server yang banyak dipakai pada platform Windows NT. Pada tahun 2000 Microsoft meluncurkan SQL Server 2000, SQL Server 2008 R2 diluncurkan pada tahun 2012 sampai sekarang Microsoft terus meluncurkan SQL Server versi terbaru.

SQL singkatan dari Structured Query Language, **SQL SERVER** adalah sistem manajemen database relasional (RDBMS) yang dirancang untuk aplikasi dengan arsitektur client/server. Istilah client, server, dan client/server dapat digunakan untuk merujuk kepada konsep yang sangat umum atau hal yang spesifik dari perangkat keras atau perangkat lunak. Pada level yang sangat umum.

Jadi secara umum SQL Server adalah sebuah Software yang dibuat oleh perusahaan Microsoft yang digunakan untuk membuat database yang dapat diimplementasikan untuk Client Server.

CLIENT adalah setiap komponen dari sebuah sistem yang meminta layanan atau sumber daya (resource) dari komponen sistem lainnya.

SERVER adalah setiap komponen sistem yang menyediakan layanan atau sumber daya ke komponen sistem lainnya. Kemudian pada pengertian sebelumnya terdapat kata RDBMS. Berikut penjelasannya :

RDBMS adalah singkatan Relational Database Management System.

RDBMS adalah dasar untuk SQL, dan untuk semua sistem database modern seperti MS SQL Server, IBM DB2, Oracle, MySQL, dan Microsoft Access.

Data dalam RDBMS disimpan dalam objek database yang disebut tabel.

TABEL adalah kumpulan data entri terkait dan terdiri dari kolom dan baris.

C. FUNGSI SQL

Seperti yang sudah dijelaskan diatas, bahwa SQL merupakan singkatan dari Structured Query Language, yang diantaranya mempunyai fungsi :

- SQL memungkinkan anda mengakses dan memanipulasi database.
- SQL dapat mengeksekusi query terhadap database
- SQL dapat mengambil data dari database
- SQL dapat menyisipkan data dalam database
- SQL dapat memperbarui data dalam database
- SQL dapat menghapus data dari database
- SQL dapat membuat database baru
- SQL dapat membuat tabel baru dalam database
- SQL dapat membuat prosedur yang tersimpan dalam database
- SQL dapat menggabungkan beberapa tabel didalam database
- SQL dapat mengatur hak akses pada tabel, prosedur

D. TIPE DATA

Data yang terdapat dalam sebuah tabel berupa field-field yang berisi nilai dari data tersebut. Nilai data dalam field memiliki tipe sendiri-sendiri. SQL Server mengenal beberapa tipe data field yaitu:

1. Tipe data numerik

Tipe data numerik dibedakan dalam dua macam kelompok, yaitu **integer** dan **floating point**. **Integer** digunakan untuk bilangan bulat sedangkan **floating point** digunakan untuk bilangan desimal

2. Tipe data string

String adalah rangkaian karakter. Tipe-tipe data yang termasuk dalam tipe data string dapat dilihat pada tabel 1 berikut:

Tabel 1. Tipe data string

Tipe Data	Kisaran Nilai
CHAR	-255 karakter
VARCHAR	255 karakter
TINYTEXT	255 karakter
TEXT	655335 karakter
MEDIUMTEX	16777215 karakter
LONGTEXT	4294967295 karakter

3. Tipe data char() dan varchar()

Tipe data char() dan varchar() pada prinsipnya sama, berbedaannya terletak pada jumlah memori yang dibutuhkan untuk penyimpanannya. Memori yang dibutuhkan untuk tipe data char() bersifat statis, besarnya bergantung pada berapa jumlah karakter yang ditetapkan pada saat field tersebut dideklarasikan.

Pada tipe data varchar() besarnya memori penyimpanan tergantung pada jumlah karakter ditambah 1 byte, seperti pada tabel 2.

Tabel 2. Letak Perbedaan Jumlah Memori

Nilai	Char (4)	Memori Penyimpanan	Varchar (4)	Memori Penyimpanan
“	“	4 bytes	“	1 bytes
‘ab’	‘ab’	4 bytes	‘ab’	3 bytes
‘abcd’	‘abcd’	4 bytes	‘abcd’	5 bytes
‘abcd’	‘abcd’	4 bytes	‘abcd’	5 bytes

4. Tipe data tanggal

Untuk tanggal dan jam, tersedia tipe-tipe data field berupa DATETIME, DATE, TIMESTAMP, TIME dan YEAR. Masing-masing tipe mempunyai kisaran nilai tertentu. SQL Server akan memberikan peringatan kesalahan (error) apabila tanggal atau waktu yang dimasukkan salah. Kisaran nilai dan besar memori penyimpanan yang diperlukan untuk masing-masing tipe data, seperti pada tabel 3 berikut

Tabel 3 Tipe Data Tanggal

Tipe Data	Kisaran Nilai	Memori Penyimpanan
DATETIME	1000-01-01 00:00 sampai 9999-12-31 23:59:59	3 byte
DATE	1000-01-01 sampai 9999-12-31	8 byte
TIMESTAMP	1970-01-01 00:00:00 sampai 2037	4 byte
TIME	-839:59:59 sampai 838:59:59	3 byte

E. OPERATOR SQL SERVER

SQL Server mendukung penggunaan operator-operator dan fungsi-fungsi diantaranya:

1. Operator Aritmatika

Suatu ekspresi yang melibatkan tipe data bilangan (numerik) dan tanggal (date) menggunakan ekspresi aritmatika. Seperti pada tabel 4 berikut

Tabel 4

Opertor	Keterangan
+	Tambah
-	Kurang
*	Kali
/	Bagi
Mod()	Modulus

2. Operator Pembandingan

Suatu ekspresi yang dapat digunakan pada klausa WHERE dan mempunyai syntax sebagai berikut: WHERE expr operator value. Tabel 5 menunjukkan operator pembandingan pada SQL Server berikut ini:

Tabel 5 Operator Pembandingan SQL Server

Operator	Keterangan
=	Sama Dengan
>	Lebih Besar
<	Lebih Kecil
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan
<>	Tidak sama dengan

3. Operator Logika

Operator ini digunakan untuk membandingkan dua nilai variabel yang bertipe boolean

4. Operator Karakter

Operator untuk membentuk pencarian string yang sesuai dengan nilai yang mencantumkan pada kondisi. Kondisi pencarian dapat berisi karakter ada 3 symbol khusus seperti pada tabel 6 berikut:

Tabel 6 Operator Karakter

Operator	Keterangan
%	Sembarang karakter berapapun
_	Sembarang satu karakter
[]	Sembarang karakter yang terletak dalam

5. Operator Lain-lain

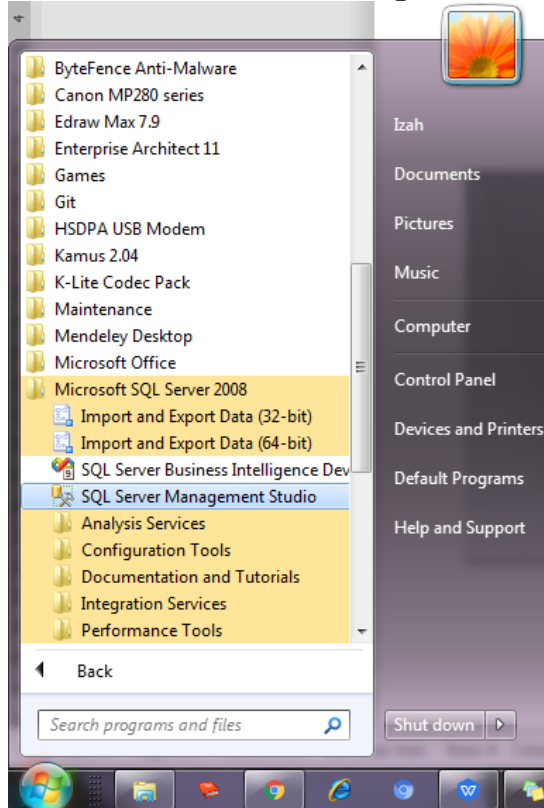
Operator yang digunakan untuk menguji nilai-nilai yang ada dalam list (tanda kurung) dan dapat juga untuk menampilkan baris berdasarkan suatu jangkauan (range) nilai. Ada 2 symbol seperti pada tabel 7 berikut:

Tabel 7 Operator Lain-lain

Operator	Keterangan
IN	Dalam
BETWEEN	Diantara

F. MEMULAI KERJA DENGAN SQL SERVER 2008

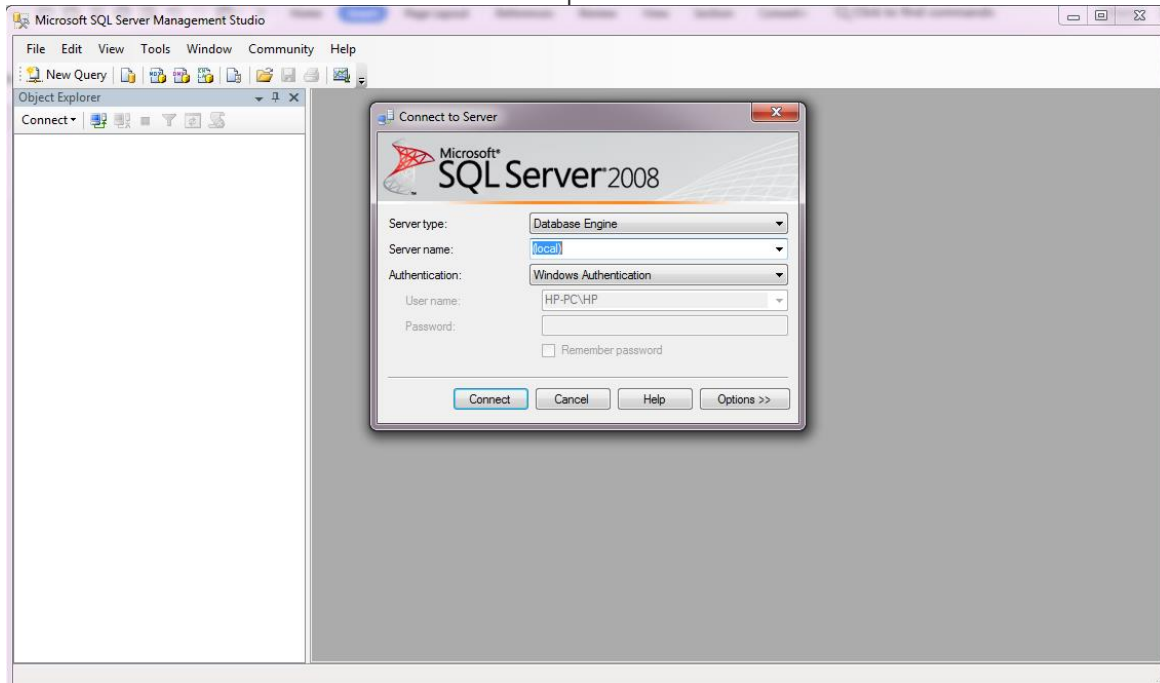
1. Untuk memulai kerja dengan SQL Server 2008, pilih start -> Microsoft SQL Server 2008 -> SQL Server Management Studio



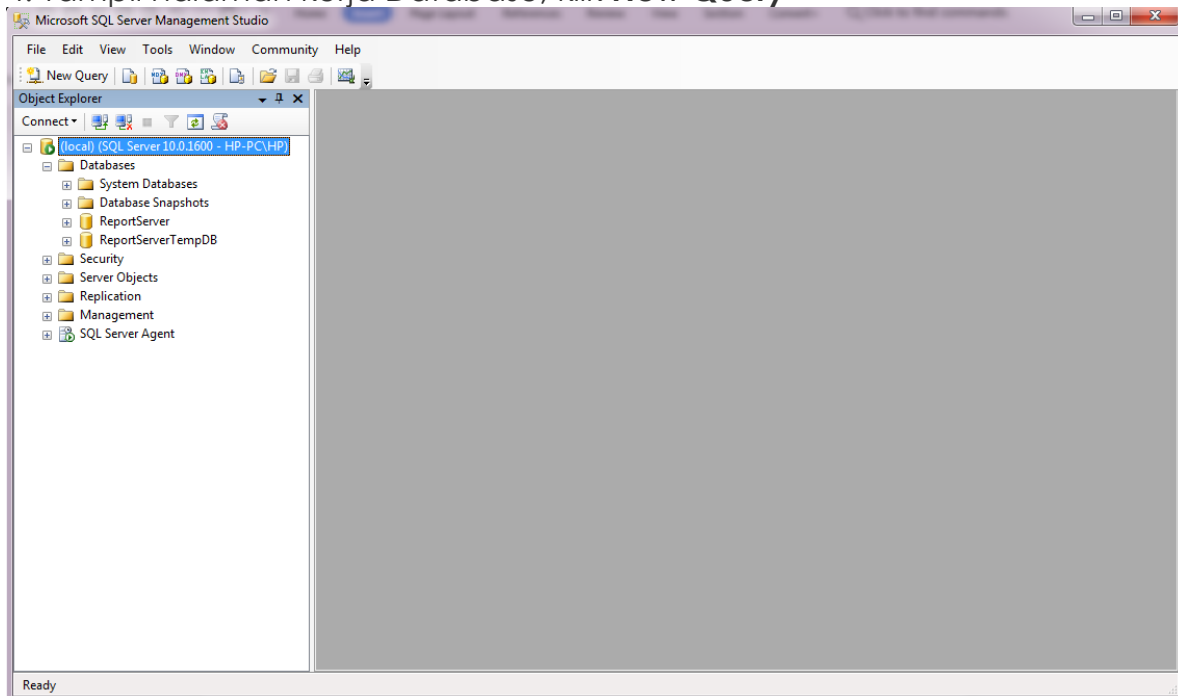
2. Akan tampil halaman loading seperti berikut



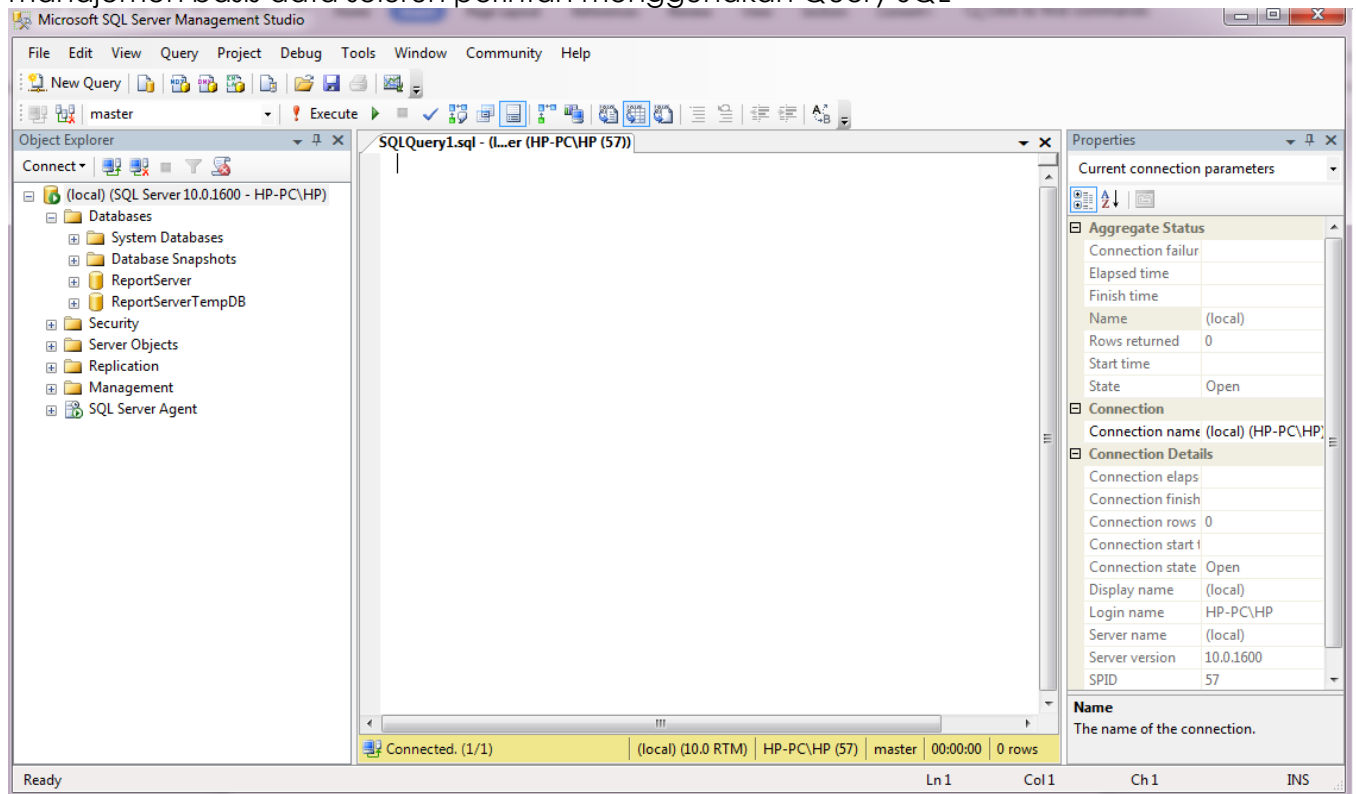
3. Muncul halaman untuk conect seperti berikut kemudian klik **connect**



4. Tampil halaman kerja Database, klik **New Query**



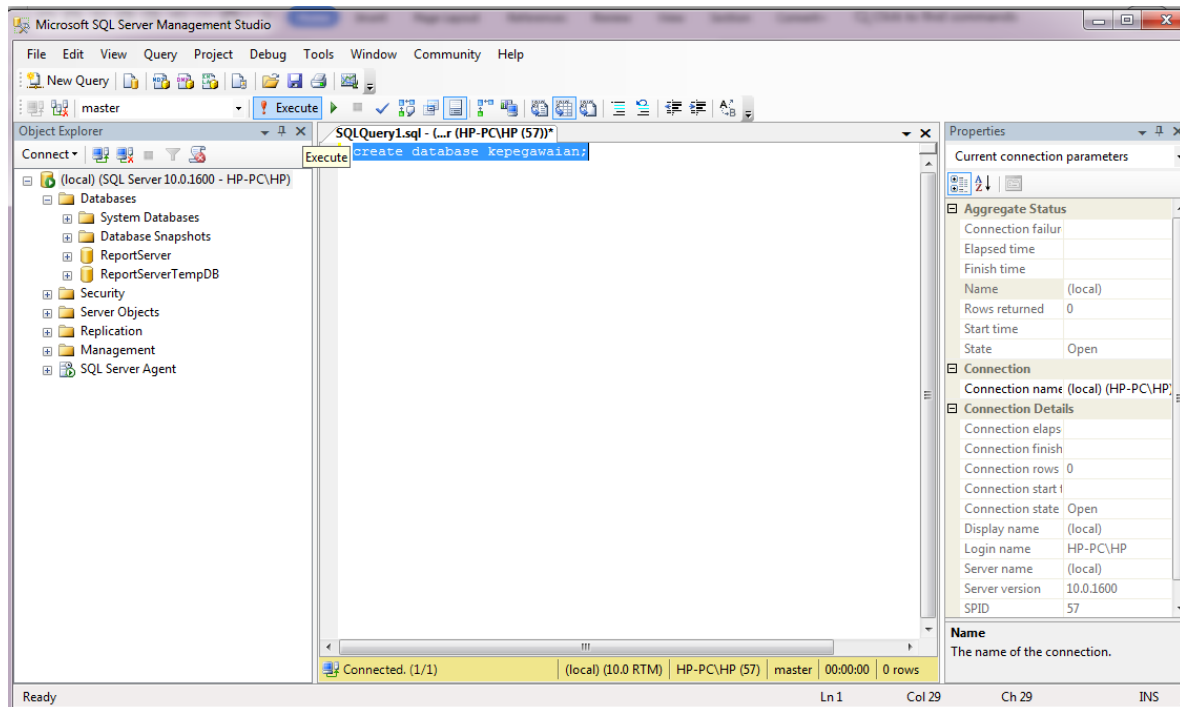
5. Halaman SQL setelah klik New Query akan tampil seperti berikut, dalam praktikum sistem manajemen basis data seluruh perintah menggunakan Query SQL



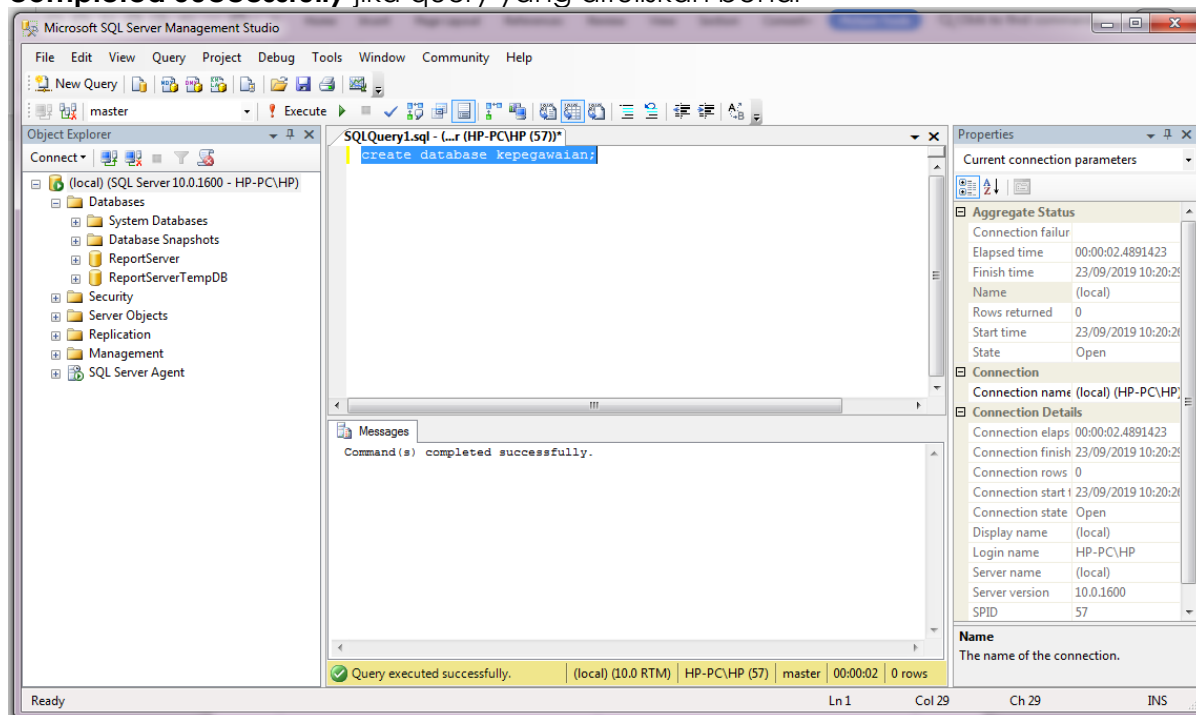
6. Membuat database baru, ketikan **create database kepegawaian** -> blok perintah tersebut kemudian klik Execute

MODUL PRAKTIKUM SISTEM MANAJEMEN BASISI DATA

...



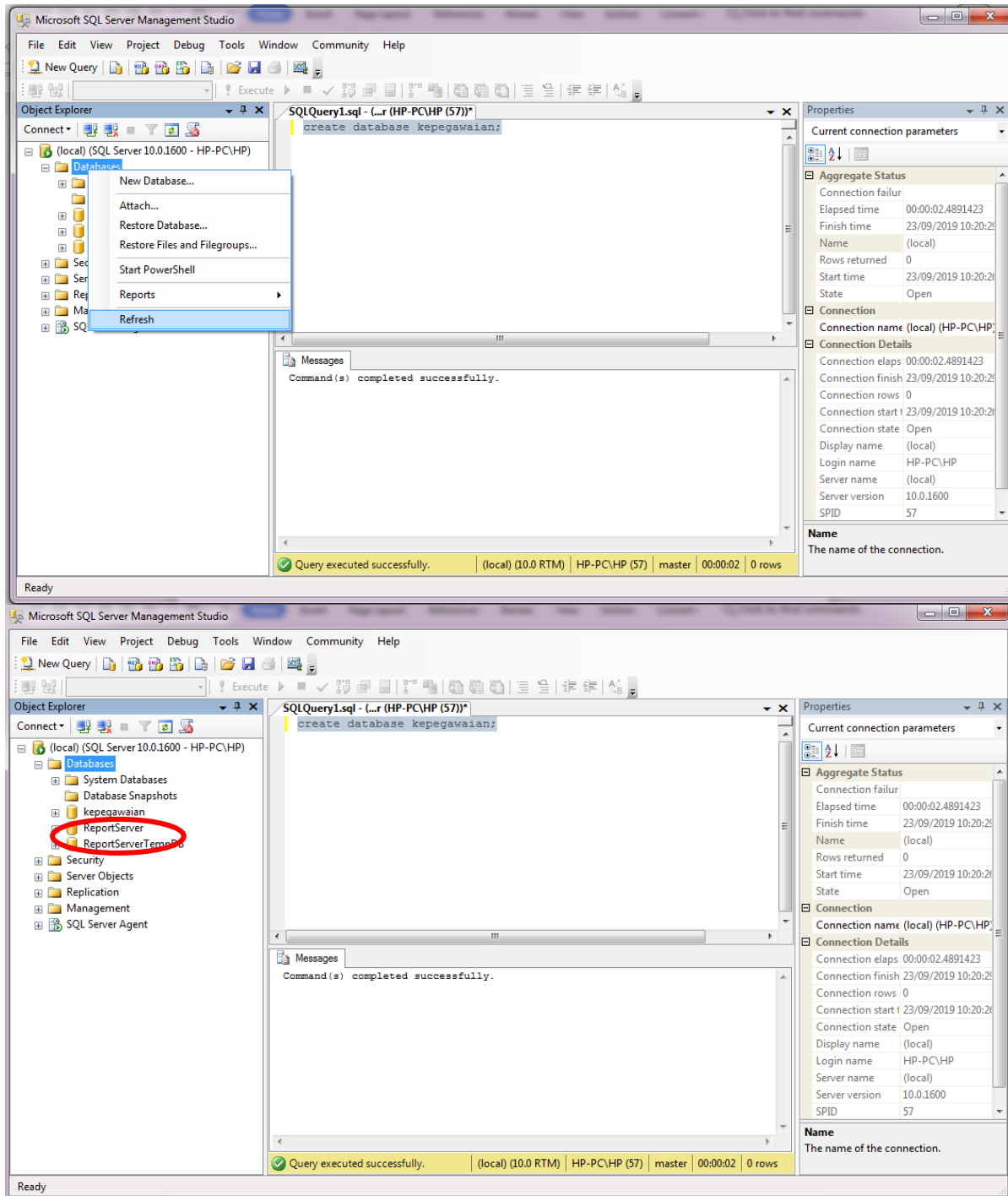
Hasil running query tersebut sebagai berikut, akan muncul di **blok Message Command (s) completed Successfully** jika query yang dituliskan benar



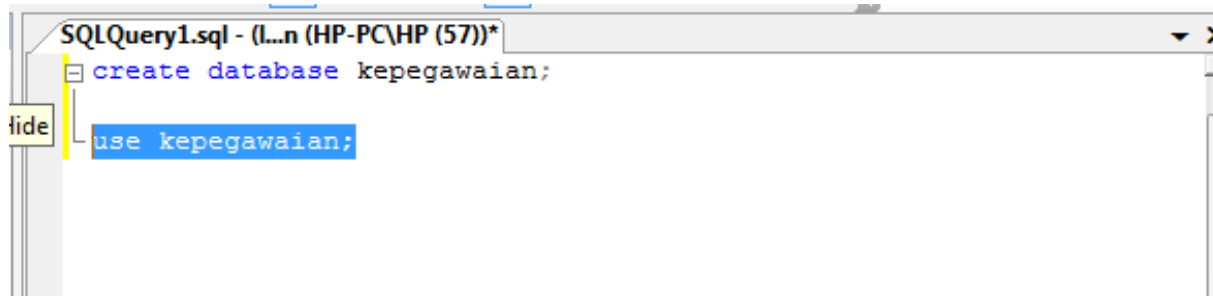
Klik kanan Refresh pada Database kemudian akan muncul database yang telah dibuat

MODUL PRAKTIKUM SISTEM MANAJEMEN BASISI DATA

...



7. Untuk memilih dan membuka database dapat menggunakan perintah berikut ini use namadatabase;
Contoh: `use kepegawaian;`



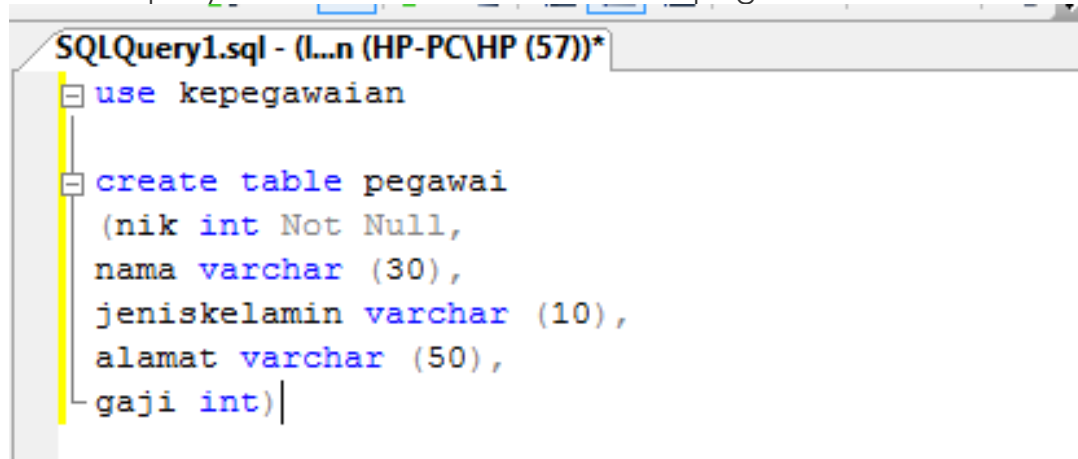
```
SQLQuery1.sql - (L...n (HP-PC\HP (57)))
create database kepegawaian;
use kepegawaian;
```

8. Membuat tabel pada database

Tabel Pegawai

Nama Field	Tipe Data
NIK	integer
Nama	Varchar (30)
JenisKelamin	Varchar (10)
Alamat	Varchar (50)
Gaji	integer

Ketikan query berikut untuk membuat tabel pegawai



```
SQLQuery1.sql - (L...n (HP-PC\HP (57)))
use kepegawaian

create table pegawai
(nik int Not Null,
nama varchar (30),
jeniskelamin varchar (10),
alamat varchar (50),
gaji int)
```

TUGAS 1

1. Tambahkan tabel divisi yang terdiri dari kode divisi dan nama divisi, identifikasi tipe data yang digunakan
2. Tambahkan tabel jabatan yang terdiri dari kode jabatan dan nama jabatan, identifikasi tipe data yang digunakan

PRAKTIKUM 2

DATA DEFINITION LANGUAGE (DDL)

DDL atau Data Definition Language adalah bagian dari SQL yang digunakan untuk mendefinisikan data dan objek database. Apabila perintah ini digunakan, entri akan dibuat ke dalam kamus data dari SQL. Didalam kategori ini terdapat perintah-perintah sebagai berikut :

PERINTAH	KETERANGAN
CREATE DATABASE	Membuat Database
CREATE TABLE	Membuat Tabel
ALTER TABLE	Mengubah dan menyisipkan kolom ke dalam tabel
DROP TABLE	Menghapus tabel dari database

Perintah CREATE

Instalasi sistem manajemen (DBMS) pada komputer memungkinkan anda untuk membuat dan mengelola banyak database independent. Untuk membuat database berikut querinya

```
CREATE DATABASE namadatabase
```

Nama database tidak boleh mengandung spasi dan tidak boleh memiliki nama yang sama antar database.

Sebelum membuat tabel dalam database pastikan terlebih dahulu database tempat membuat tabel sudah aktif, untuk mengaktifkan database yang anda buat dapat menggunakan statement **USE nm_Database**. Untuk membuat tabel berikut querinya.

```
CREATE TABLE nm_tabel  
(nm_field1 tippedata1 [NOT NULL | NULL]  
[{nm_field2 tippedata2 [NOT NULL | NULL]}...])
```

Nm_tabel adalah nama dari tabel yang akan anda buat, usahakan namanya sesuai dengan entity yang diwakilinya dan jangan lebih dari 128 karakter. Banyaknya tabel yang dapat ditampung oleh database dibatasi oleh banyaknya objek dalam databse. Sebuah database setidaknya dapat menampung 2 Miliar objek termasuk tabel, view (jenis lain tabel), stored procedure, trigger, dan constraint. **nm_field1**, **nm_field2** adalah nama field yang harus ada dalam dalam

sebuah tabel yang mewakili element entity tersebut. Tiap field akan menampung data dengan tipe data tertentu yang ditunjukkan oleh `tipe_data1`, `tipe_data2` dan seterusnya.

Opsi berikutnya adalah **constraint (syarat)** masing-masing tabel apakah **boleh kosong (NULL)** atau **harus diisi (NOT NULL)**. Nilai default dari constraint ini adalah NULL yang artinya field tersebut boleh kosong atau tidak diisi.

CONSTRAINT

Constraint merupakan batasan atau aturan yang ada pada tabel. SQL menyediakan beberapa tipe constraint berikut:

1. **NOT NULL** merupakan suatu kolom yang mendefinisikan dengan constraint NOT NULL. Kolom yang berfungsi sebagai kunci primer (Primary Key) otomatis tidak boleh NULL.
2. **UNIQUE** mendefinisikan suatu kolom bersifat unik, artinya antara satu data dengan data lain namanya tidak boleh sama, misal alamat email.
3. **PRIMARY KEY** Constraint PRIMARY KEY membentuk key yang unik untuk suatu tabel.
4. **FOREIGN KEY** Constraint didefinisikan pada suatu kolom yang ada pada suatu tabel, dimana kolom tersebut juga dimiliki oleh tabel yang lain sebagai suatu PRIMARY KEY bisa digunakan untuk menghubungkan antara dua tabel.
5. **CHECK** constraint yang satu ini mendefinisikan sebuah kondisi untuk data agar dapat masuk dalam field artinya tiap pemasukan data atau editing terhadap data record, field yang kita masukkan akan selalu diperiksa apakah isinya ada diantara data-data yang kita masukkan, jika tidak ada maka SQL akan menampilkan pesan ERROR
6. **DEFAULT** digunakan untuk mendefinisikan nilai default dari field yang mana ketika ada baris baru yang dimasukkan kedalam tabel nilai default dari field akan digunakan apabila tidak ada nilai yang diberikan padanya.
7. **IDENTITY** hanya menampung nilai bertipe data integer. Standar property ini sudah ditentukan oleh sistem.

Perintah ALTER

Setelah membuat tabel dalam database, kita dapat memodifikasi field pada tabel yang dibuat. Dengan perintah ALTER kita dapat membuat perubahan pada struktur tabel tanpa menghapus dan menciptakan. Query ALTER :

Query ini digunakan untuk menambah field pada tabel

```
ALTER TABLE namatabel ADD nama_field tipe_data (lebar)
```

Query ini digunakan untuk merubah field pada tabel

```
ALTER TABLE namatabel ALTER COLUMN nama_field tipe_data
(lebar)
```

Queri ini digunakan untuk menghapus field pada tabel

```
ALTER TABLE namatabel DROP COLUMN nama_field.
```

Perintah DROP

Perintah terakhir dari Data Definition Language, DROP memungkinkan kita untuk menghapus seluruh objek dalam database dari DBMS. Gunakan perintah ini dengan hati-hati, perlu diingat bahwa perintah DROP akan menghapus data keseluruhan struktur dari database. Querinya sebagai berikut

```
DROP DATABASE namadatabase (Untuk menghapus database)
```

```
DROP TABLE namatabel (Untuk menghapus tabel)
```

LATIHAN

1. Buatlah database dengan nama “mart” yang mempunyai dua tabel yaitu tabel “Barang” dan tabel “Pembelian”, isilah kedua tabel sesuai dengan ketentuan tabel dibawah ini

CREATE DATABASE

```
create database mart
```

CREATE TABLE BARANG

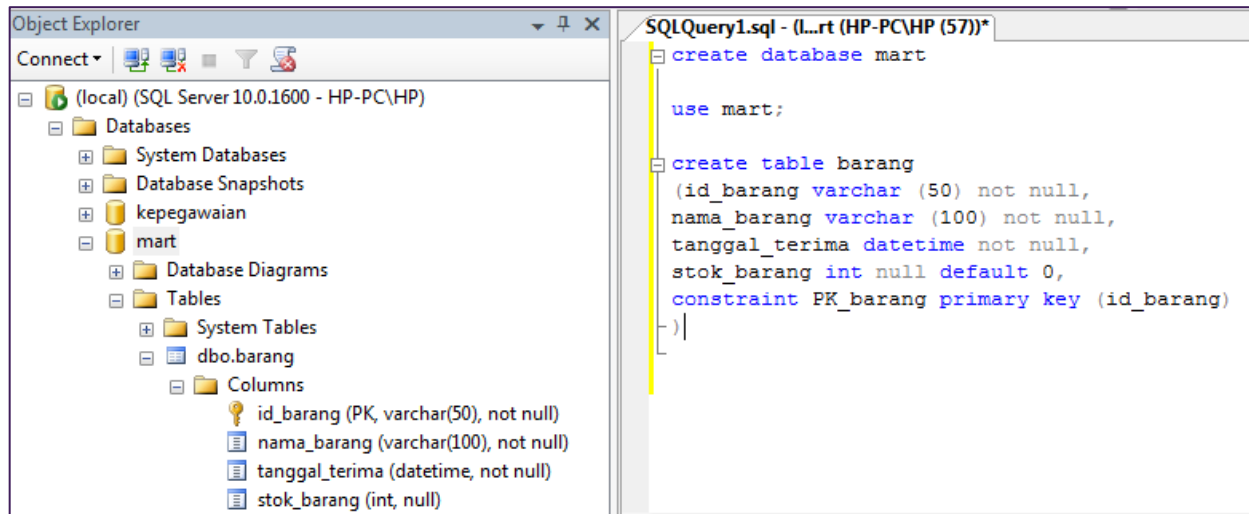
Tabel Barang

Nama Field	Tipe Data
Id_barang	varchar
Nama_barang	varchar
Tanggal_diterima	datetime
Stok_barang	int

TULISKAN QUERY TABEL BARANG

```
create table barang
(id_barang varchar (50) not null,
nama_barang varchar (100) not null,
tanggal_terima datetime not null,
stok_barang int null default 0,
constraint PK_barang primary key (id_barang)
)
```

HASIL RUNNING QUERY



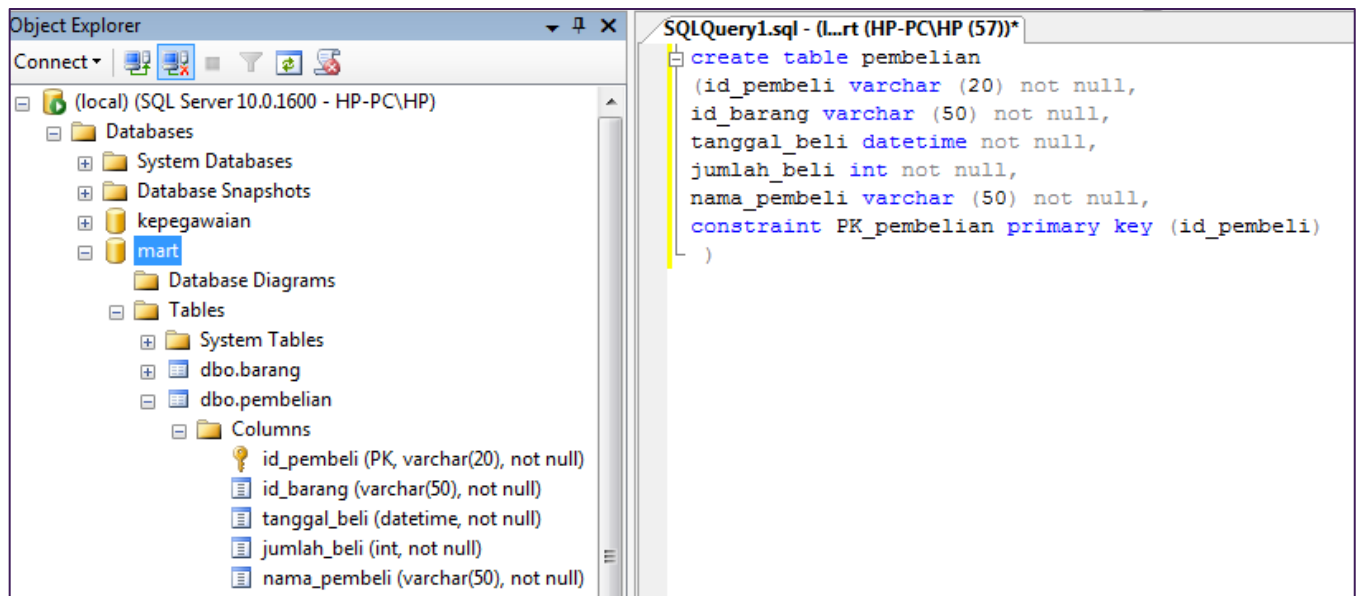
CREATE TABLE PEMBELIAN

Nama Field	Tipe Data
Id_pembeli	varchar
Id_barang	varchar
Tanggal_beli	datetime
Jumlah_beli	int
Nama_pembeli	varchar

TULISKAN QUERY TABEL PEMBELIAN


```
create table pembelian
(id_pembeli varchar (20) not null,
id_barang varchar (50) not null,
tanggal_beli datetime not null,
jumlah_beli int not null,
nama_pembeli varchar (50) not null,
constraint PK_pembelian primary key (id_pembeli)
)
```

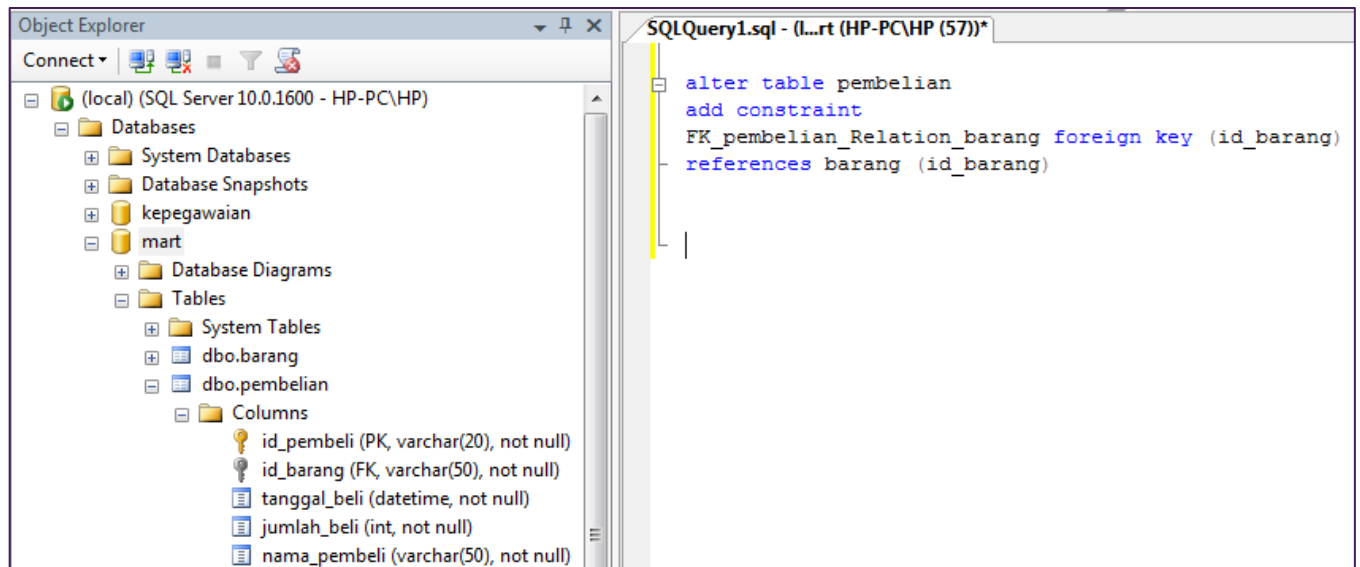
HASIL RUNNING QUERY



ALTER

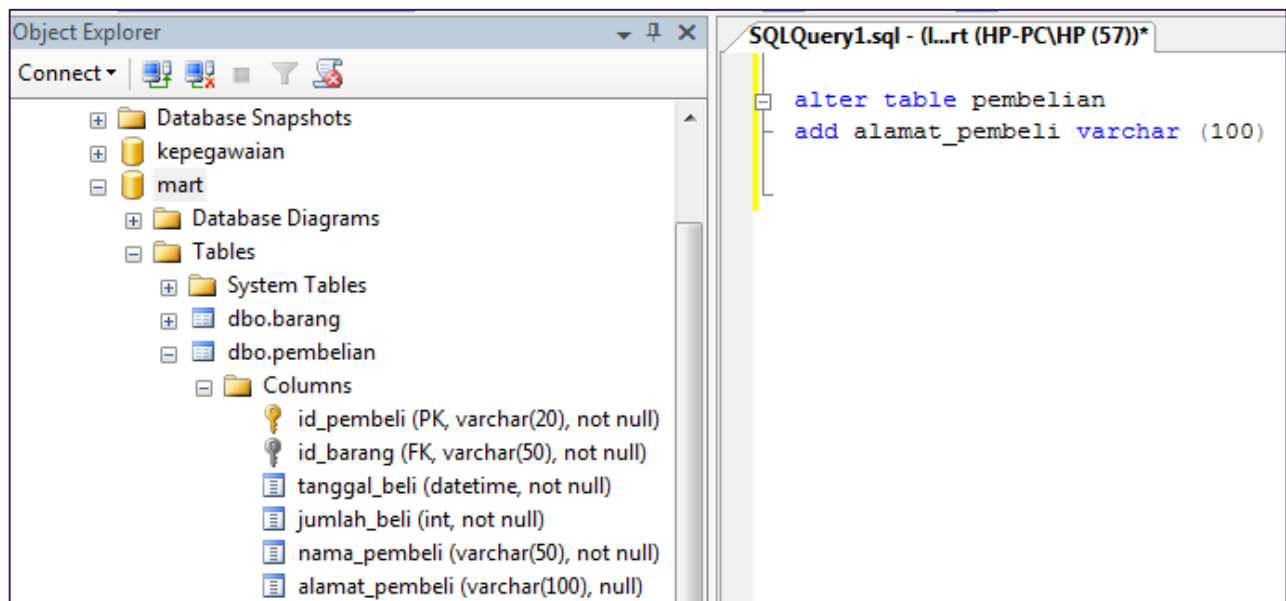
Karena kita belum menambahkan Foreign key pada saat kita membuat tabel Pembelian, maka kita dapat menambahkan Foreign key dengan perintah ALTER seperti query dibawah ini

```
alter table pembelian
add constraint
FK_pembelian_Relation_barang foreign key (id_barang)
references barang (id_barang)
```



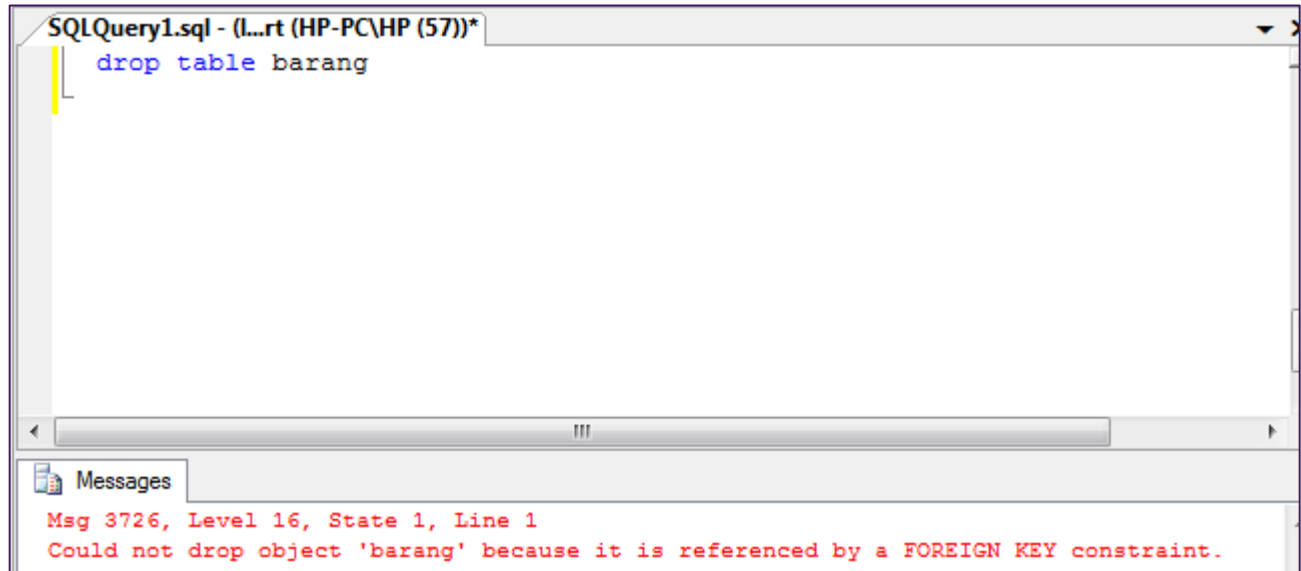
Menambahkan field “alamat_pembeli” pada tabel pembelian

```
alter table pembelian
add alamat_pembeli varchar (100)
```



Drop

Menghapus tabel Barang, maka akan muncul peringatan karena primary key pada tabel barang yaitu id_barang digunakan pada tabel pembelian. Tabel dapat dihapus jika tidak memiliki relasi dengan tabel lainnya.



TUGAS 2:

1. Buatlah database baru dengan perintah SQL dengan nama "kuliah" yang memiliki dua tabel yaitu tabel "prodi" dan tabel "mahasiswa"
2. Isikan kedua tabel dengan field masing-masing dibawah ini!
Tabel prodi : id_prodi, nama_prodi, ketua_prodi
Tabel mahasiswa : NIM, id_prodi, nama_mahasiswa, JK, alamat
3. Tentukan tipe data masing-masing field, Primary key serta Foreign Key nya agar kedua tabel saling berhubungan!

PRAKTIKUM 3 & 4

DATA MANIPULATION LANGUAGE (DML)

Merupakan bahasa yang memungkinkan pengguna mengakses atau memanipulasi data seperti yang diatur oleh model data. Manipulasi data adalah :

- ✓ Pengambilan informasi yang disimpan dalam basisdata
- ✓ Penempatan informasi dalam basisdata
- ✓ Penghapusan informasi dari basisdata
- ✓ Modifikasi informasi yang disimpan dalam basisdata

DML (Data Manipulation Language) merupakan bahasa yang bertujuan memudahkan pemakai untuk mengakses data sebagaimana direpresentasikan oleh model data. Ada 2 jenis DML, yaitu :

1. Prosedural, yang mensyaratkan agar pemakai menentukan, data apa yang diinginkan serta bagaimana cara mendapatkannya.
2. Nonprosedural, yang membuat pemakai dapat menentukan data apa yang diinginkan tanpa menyebutkan bagaimana cara mendapatkannya.

Query adalah pernyataan yang meminta pengguna mengambil informasi. Bagian DML yang terlibat dalam pengambilan informasi disebut bahasa query. Istilah bahasa

query sering disamakan dengan istilah bahasa manipulasi data. Sedangkan SQL adalah sebuah sintaks untuk mengeksekusi query.

a. MEMASUKKAN DATA

Pernyataan INSERT INTO digunakan untuk memasukkan data pada tabel. Perintah yang digunakan :

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

Urutan nilai yang diletakkan dalam tanda kurung disesuaikan dengan urutan kolom dalam tabel. Akan tetapi kita bisa menentukan kolom-kolom yang akan diisi dengan data baru, yaitu :

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

Kolom-kolom yang tidak disebutkan pada Insert secara otomatis akan diisi dengan

Null dan kolom yang tidak disebutkan dalam Insert haruslah yang tidak Not Null. Dengan menggunakan data pada praktikum 1, tambahkan data ke dalam tabel Pegawai seperti perintah berikut :

1. Jika database yang tersedia lebih dari satu maka gunakan perintah **use**

```
use kepegawaian
```

2. Kemudian ketikkan perintah **insert** untuk menginput data, jika menggunakan perintah langsung tanpa menyebutkan judul kolomnya maka perintahnya sebagai berikut:

```
insert into pegawai values (23456, 'Jhon', 'laki-laki', 'Purwokerto', 50000000)
```

Output

Results		Messages			
	nik	nama	jeniskelamin	alamat	gaji
1	23456	Jhon	laki-laki	Purwokerto	50000000

Jika input data dengan menyebutkan judul kolom terlebih dahuhulu, maka query sebagai berikut

```
insert into pegawai (nik, nama, jeniskelamin, alamat, gaji)
values (34256, 'Maryam', 'Perempuan', 'Bandung', 3500000)
```

Output

Results		Messages			
	nik	nama	jeniskelamin	alamat	gaji
1	23456	Jhon	laki-laki	Purwokerto	50000000
2	34256	Maryam	Perempuan	Bandung	3500000

b. PENGUBAHAN DATA

Pernyataan UPDATE digunakan untuk modifikasi data dalam tabel menggunakan perintah berikut :

```
UPDATE nama_tabel
SET nama_kolom = nilai_baru
WHERE nama_kolom = nilai;
```

SET digunakan untuk menentukan kolom yang akan diubah dan nilai penggantinya.
WHERE digunakan untuk menentukan kondisi dari baris3baris yang akan diganti.

Lakukan perintah berikut untuk mengubah data pegawai bernama 'Jhon' menjadi 'Jhon Smith':

```
update pegawai set nama='Jhon Smith' where nik=23456
```

Output

	nik	nama	jeniskelamin	alamat	gaji
1	23456	Jhon Smith	laki-laki	Purwokerto	50000000
2	34256	Maryam	Perempuan	Bandung	3500000

c. MENGHAPUS DATA

Pernyataan DELETE digunakan untuk menghapus baris pada tabel, perintah yang digunakan seperti berikut:

```
DELETE FROM nama_tabel
WHERE nama_kolom = nilai;
```

Lakukan perintah berikut untuk menghapus data dari tabel pegawai dengan NIK=23456

```
delete from pegawai where nik=23456;
```

Output

	nik	nama	jeniskelamin	alamat	gaji
1	34256	Maryam	Perempuan	Bandung	3500000

d. PERNYATAAN SELECT

Secara umum perintah SELECT hanya difungsikan untuk menampilkan data yang ada di dalam suatu tabel. Tetapi dalam pengembangannya, perintah ini akan menjadi sebuah perintah yang sangat penting dan berpengaruh hingga saat pemrograman di stored procedures dan triggers. Struktur perintah Select seperti berikut :

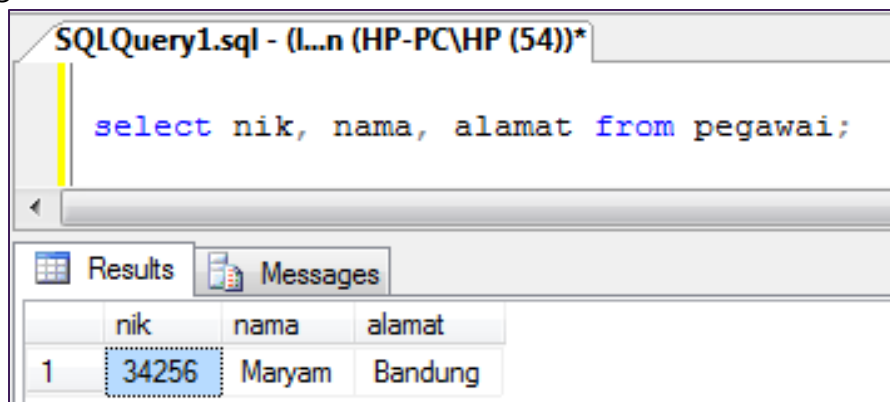
```
SELECT [DISTINCT] select_list  
FROM table_source  
[WHERE search_condition] ;
```

1. Memilih beberapa kolom dalam tabel

Untuk memilih beberapa kolom gunakan perintah sebagai berikut:

```
SELECT nama_kolom1, nama_kolom2, ... FROM nama_tabel;
```

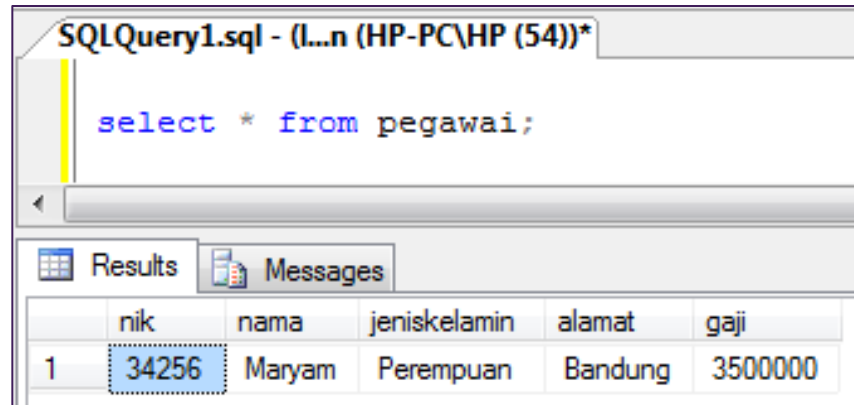
Lakukan perintah berikut untuk memilih Kolom NIK, nama dan alamat dari tabel pegawai :



Untuk memilih semua kolom dari tabel, dapat menuliskan tanda asterisk sesudah kata Select. Simbol * berarti semua kolom, seperti berikut :

```
SELECT * FROM nama_tabel;
```

Untuk menampilkan semua kolom pada tabel pegawai menggunakan perintah berikut:



The screenshot shows a SQL Query Editor window titled "SQLQuery1.sql - (L...n (HP-PC\HP (54)))". The query entered is `select * from pegawai;`. Below the query editor, there are tabs for "Results" and "Messages". The "Results" tab is active, displaying a table with 6 columns: nik, nama, jeniskelamin, alamat, and gaji. The first row of data is highlighted.

	nik	nama	jeniskelamin	alamat	gaji
1	34256	Maryam	Perempuan	Bandung	3500000

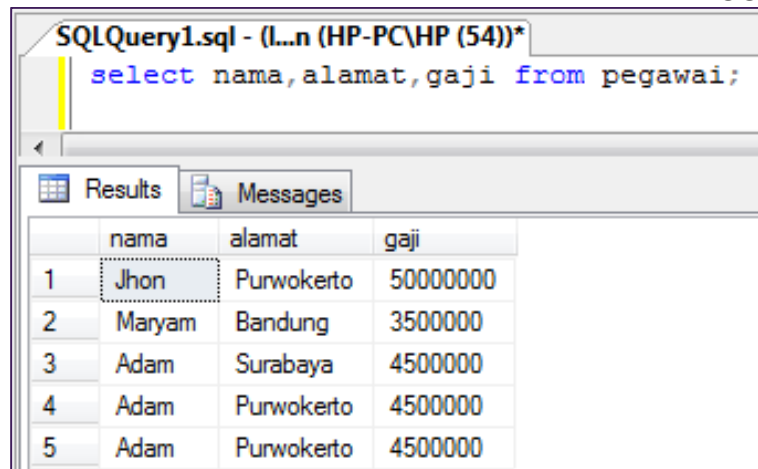
2. Pernyataan Distinct

DISTINCT adalah salah satu operator yang digunakan untuk mencegah adanya duplikasi data atau record. Misalkan ada 10 orang yang bernama 'AHMAD', maka dengan menggunakan operator Distinct, Anda bisa mengeliminasi data 'AHMAD' menjadi satu nama saja.

Untuk memilih hanya nilai yang berbeda atau nilai data yang unik. Perintah SELECT DISTINCT berikut :

```
SELECT DISTINCT nama_kolom FROM nama_tabel;
```

Lakukan perintah dibawah berikut untuk membedakan penggunaan Distinct:



The screenshot shows a SQL Query Editor window titled "SQLQuery1.sql - (L...n (HP-PC\HP (54)))". The query entered is `select nama,alamat,gaji from pegawai;`. Below the query editor, there are tabs for "Results" and "Messages". The "Results" tab is active, displaying a table with 4 columns: nama, alamat, and gaji. The results show 5 rows of data, with the first row highlighted.

	nama	alamat	gaji
1	Jhon	Purwokerto	5000000
2	Maryam	Bandung	3500000
3	Adam	Surabaya	4500000
4	Adam	Purwokerto	4500000
5	Adam	Purwokerto	4500000

SQLQuery1.sql - (L...n (HP-PC\HP (54)))*

```
select distinct nama,alamat,gaji from pegawai;
```

Results Messages

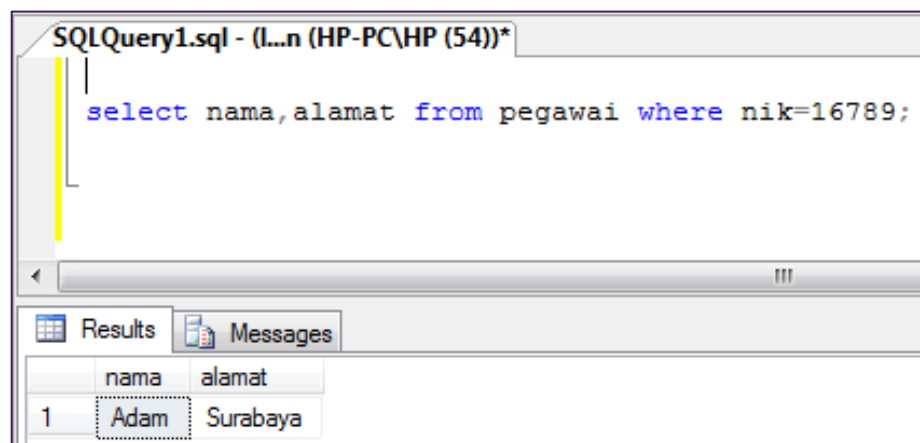
	nama	alamat	gaji
1	Adam	Purwokerto	4500000
2	Adam	Surabaya	4500000
3	Jhon	Purwokerto	50000000
4	Maryam	Bandung	3500000

3. Klausula WHERE

Klausula WHERE digunakan untuk menentukan kriteria seleksi. Untuk memilih data suatu tabel dengan kriteria tertentu, klausula WHERE dapat ditambahkan pada pernyataan SELECT. Perintah yang digunakan seperti berikut:

```
SELECT nama_kolom  
FROM nama_tabel  
WHERE kolom operator nilai;
```

Lakukan perintah berikut dengan menggunakan klausula Where untuk menampilkan data nama dan alamat pegawai dengan NIK:16789



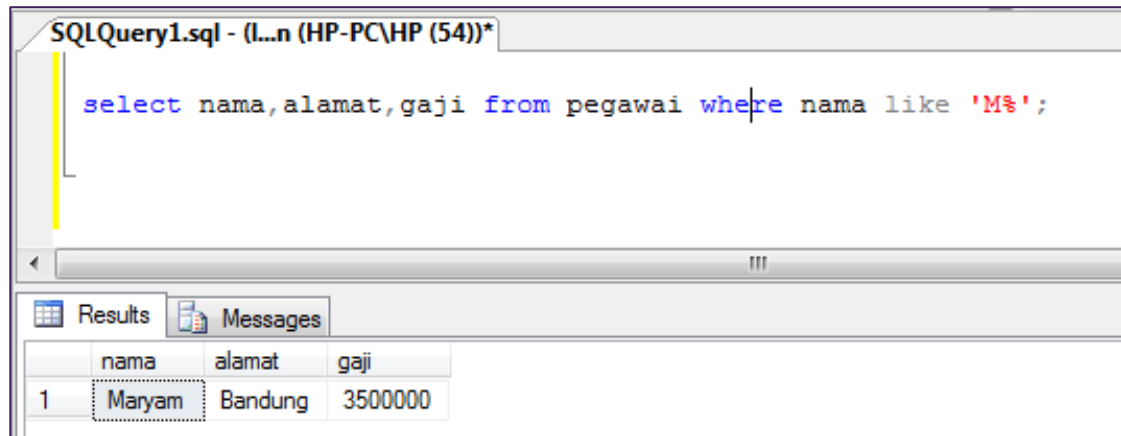
4. Kondisi LIKE

Kondisi LIKE digunakan untuk menentukan pencarian berdasarkan pola tertentu pada suatu kolom, perintah yang digunakan seperti berikut :

```
SELECT nama_kolom  
FROM nama_tabel  
WHERE nama_kolom LIKE pola;
```

anda " %" dapat digunakan untuk menentukan Wildcard (sembarang huruf), baik sebelum maupun sesudah pola. Perlu diingat operasi ini, hanya untuk perbandingan nilai bertipe string.

Lakukan perintah berikut untuk menampilkan nama depan pegawai yang dimulai dengan huruf M :



TUGAS

1. Mengubah data yang terdiri dari 2 atau lebih field dalam satu tabel, Misal; mengubah nama = 'maryam' , alamat='Bandung' menjadi nama ='Maryam', Alamat ='Purwokerto'.
2. Menampilkan data pegawai yang mempunyai alamat Purwokerto.
3. Menampilkan data pegawai yang memiliki gaji lebih besar dari 4000000.
4. Menampilkan data pegawai yang mempunyai alamat 'Purwokerto' dan mempunyai gaji lebih kecil dari 3000000.

PRAKTIKUM 5&6

DATA MANIPULATION LANGUAGE (DML) II

Perintah **SELECT** adalah perintah untuk menampilkan data yang ada di dalam suatu tabel, yang mana dalam pengembangannya perintah ini akan menjadi sebuah perintah yang sangat penting dan berpengaruh. Selain perintah-perintah **SELECT** yang dibahas dalam modul praktikum 3 masih ada banyak perintah-perintah **SELECT** lain, antara lain perintah **ALIAS**, menampilkan data lebih dari 2 tabel, **ORDER BY**, **GROUP BY**, **HAVING**, fungsi **AGREGATE**, dan beberapa perintah yang lain.

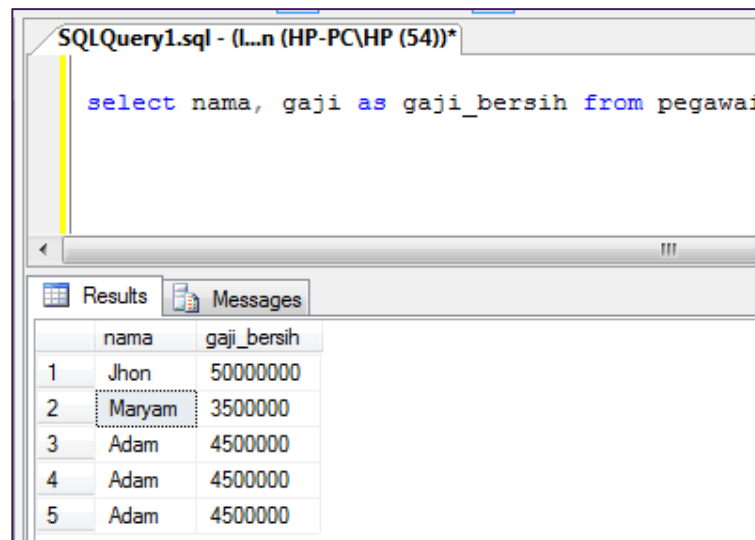
1. PEMBERIAN NAMA LAIN (ALIAS)

Perintah Select dapat dilakukan dengan memberikan nama lain atau alias yaitu dapat digunakan untuk memberikan alias pada **nama_kolom** dan **nama_tabel**.

➤ Pemberian alias pada kolom

```
SELECT nama_kolom AS nama_alias  
FROM namatabel; Atau  
SELECT nama kolom nama alias  
FROM nama_tabel;
```

Lakukan perintah berikut untuk memberikan nama alias pada kolom Gaji menjadi Gaji Bersih :



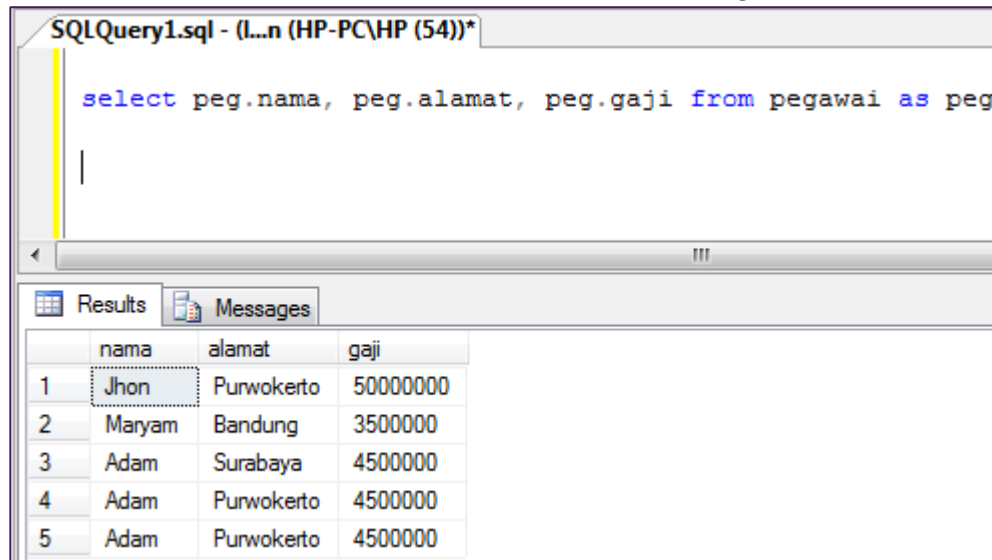
The screenshot shows a SQL query editor window titled "SQLQuery1.sql - (L...n (HP-PC\HP (54)))". The query entered is: `select nama, gaji as gaji_bersih from pegawai`. Below the query, there is a "Results" tab showing the output of the query. The results are displayed in a table with two columns: "nama" and "gaji_bersih". The table contains five rows of data.

	nama	gaji_bersih
1	Jhon	5000000
2	Maryam	3500000
3	Adam	4500000
4	Adam	4500000
5	Adam	4500000

➤ **Pemberian alias pada tabel Pemberian nama alias**

```
SELECT nama alias.nama kolom  
FROM namatabel AS nama_alias;  
Atau  
SELECT nama alias.nama kolom  
FROM nama tabel nama alias
```

Lakukan perintah berikut untuk memberikan alias Peg pada tabel pegawai



The screenshot shows a SQL query window titled 'SQLQuery1.sql - (L...n (HP-PC\HP (54)))'. The query is: `select peg.nama, peg.alamat, peg.gaji from pegawai as peg`. Below the query, there is a 'Results' tab showing a table with 5 rows and 3 columns: nama, alamat, and gaji.

	nama	alamat	gaji
1	Jhon	Purwokerto	50000000
2	Maryam	Bandung	3500000
3	Adam	Surabaya	4500000
4	Adam	Purwokerto	4500000
5	Adam	Purwokerto	4500000

2. MENAMPILKAN DATA DARI 2 TABEL

Perintah Select juga dapat dilakukan untuk beberapa tabel sekaligus dengan syarat bahwa tabel3tabel yang akan ditampilkan semua mempunyai relasi baik secara implisit maupun eksplisit. Perintah untuk menampilkan data lebih dari dua tabel seperti berikut :

```
SELECT * FROM namatabel1, namatabel2, namatabel-n;
```

Jika dalam database kepegawaian belum ada tabel departemen tambahkan terlebih dahulu tabel department sebagai berikut:

```
SQLQuery1.sql - (L...n (HP-PC\HP (54))*)
create table departement
( nomor int not null primary key,
nama_dept varchar (30),
jml_pegawai int,
nik int
- )
```

Selanjutnya lakukan insert data pada tabel departement

```
SQLQuery1.sql - (L...n (HP-PC\HP (54))*)
insert into departement values (01,'Marketing', 8, 45678)
insert into departement values (02,'HRD', 5,16789)
```

	nomor	nama_dept	jml_pegawai	nik
1	1	Marketing	8	45678
2	2	HRD	5	16789

Lakukan perintah berikut untuk menampilkan nama, alamat dari tabel pegawai dan Nama Departemen dari tabel departeme.

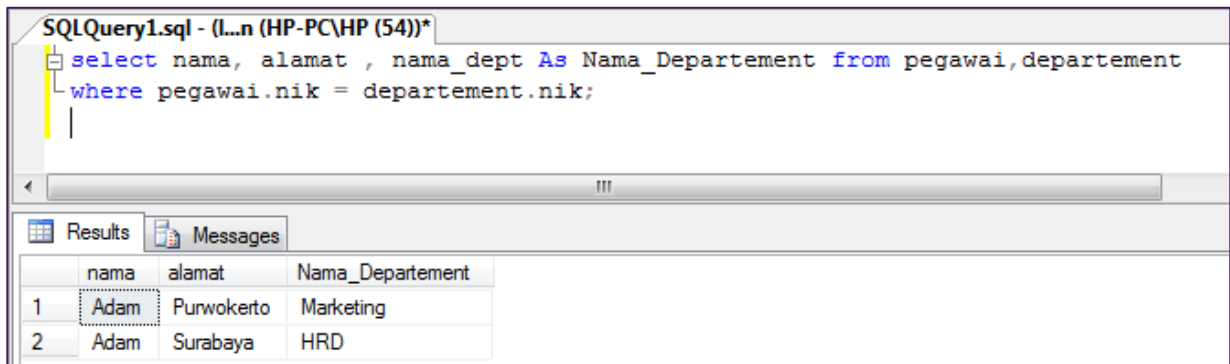
```
SQLQuery1.sql - (L...n (HP-PC\HP (54))*)
select nama, alamat , nama_dept As Nama_Departement from pegawai,departement
```

	nama	alamat	Nama_Departement
1	Adam	Surabaya	Marketing
2	Jhon	Purwokerto	Marketing
3	Maryam	Bandung	Marketing
4	Adam	Purwokerto	Marketing
5	Adam	Purwokerto	Marketing
6	Adam	Surabaya	HRD
7	Jhon	Purwokerto	HRD
8	Maryam	Bandung	HRD
9	Adam	Purwokerto	HRD
10	Adam	Purwokerto	HRD

3. KONDISI WHERE UNTUK RELASI

Jika kita mengambil informasi dari lebih dari sebuah tabel, maka kita perlu menambahkan kondisi untuk relasi antara kedua tabel tersebut. Untuk relasi kita harus memperhatikan kolom dari 2 atau lebih tabel yang mempunyai data yang sama.

Jika ada nama field yang sama dari tabel yang disertakan, maka sebelum nama field, berikan nama tabel diikuti sebuah titik (.) Pada latihan 2, data yang ditampilkan seluruh pasangan baris data departemen di tiap baris pegawai, yang tentu saja bukan informasi yang benar. **Maka** supaya informasi akurat maka ditambah relasi dengan menggunakan kolom yang sama yaitu **Departemen.nik dan Pegawai.nik**.

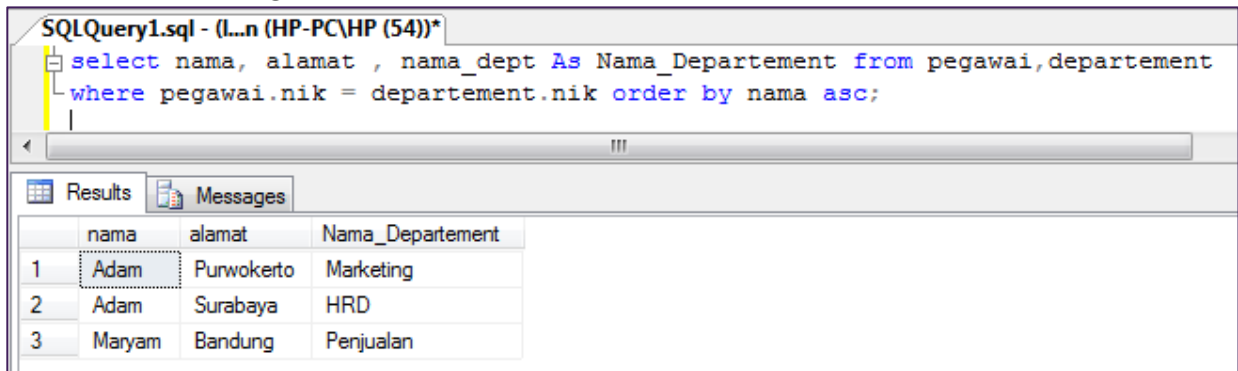


The screenshot shows a SQL query window titled "SQLQuery1.sql - (L...n (HP-PC\HP (54)))". The query is: `select nama, alamat , nama_dept As Nama_Departement from pegawai,departement where pegawai.nik = departement.nik;`. Below the query, the "Results" tab is active, displaying a table with 4 columns: nama, alamat, and Nama_Departement. There are 2 rows of data.

	nama	alamat	Nama_Departement
1	Adam	Purwokerto	Marketing
2	Adam	Surabaya	HRD

4. KLAUSA ORDER BY

Klausula ORDER BY digunakan untuk mengurutkan data. Jika kita ingin menampilkan nama, alamat dan departemen berdasarkan urutan nama secara ascending, maka perintah



The screenshot shows the same SQL query window as before, but the query now includes an order by clause: `select nama, alamat , nama_dept As Nama_Departement from pegawai,departement where pegawai.nik = departement.nik order by nama asc;`. The "Results" tab displays a table with 4 columns: nama, alamat, and Nama_Departement. There are 3 rows of data, sorted by the 'nama' column in ascending order.

	nama	alamat	Nama_Departement
1	Adam	Purwokerto	Marketing
2	Adam	Surabaya	HRD
3	Maryam	Bandung	Penjualan

SQLQuery1.sql - (L...n (HP-PC\HP (54))*)

```
select nama, alamat , nama_dept As Nama_Departement from pegawai,departement
where pegawai.nik = departement.nik order by nama desc;
```

Results Messages

	nama	alamat	Nama_Departement
1	Maryam	Bandung	Penjualan
2	Adam	Purwokerto	Marketing
3	Adam	Surabaya	HRD

5. OPERATOR IN

Ekspresi ini digunakan untuk membandingkan dengan sebuah kumpulan nilai. Kumpulan nilai bisa berupa:

- Nilai-nilai yang diisikan
- query tunggal yaitu query yg hasilnya hanya terdapat sebuah kolom saja

Lakukan perintah berikut untuk menampilkan nama, alamat dari pegawai yang mempunyai alamat Purwokerto dan Surabaya:

SQLQuery1.sql - (L...n (HP-PC\HP (54))*)

```
select nama, alamat from pegawai where alamat in ('purwokerto', 'surabaya')
```

Results Messages

	nama	alamat
1	Adam	Surabaya
2	Jhon	Purwokerto
3	Adam	Purwokerto
4	Adam	Purwokerto

6. OPERATOR BETWEEN...AND

Operator BETWEEN...AND memilih data antara dua nilai. Nilai dapat berupa angka, teks, atau tanggal, perintah yang digunakan sebagai berikut:

```
SELECT nama kolom
FROM nama tabel
WHERE nama kolom
BETWEEN/NOT BETWEEN nilai1 AND nilai2;
```


Buat terlebih dahulu tabel tanggungan seperti berikut:

SQLQuery1.sql - (L...n (HP-PC\HP (57))*)

```

create table tanggungan
(
    nik int not null,
    nama_anak varchar (50),
    jenis_kelamin varchar (10),
    tgl_lahir date,
    hubungan varchar (20)
)

select * from tanggungan

insert into tanggungan values (23456, 'Yusuf', 'laki-laki', '10-09-2013', 'anak')
insert into tanggungan values (16789, 'Hasan', 'laki-laki', '06-12-2010', 'anak')
insert into tanggungan values (45876, 'Husen', 'laki-laki', '07-08-2011', 'anak')
    
```

Results Messages

	nik	nama_anak	jenis_kelamin	tgl_lahir	hubungan
1	23456	Yusuf	laki-laki	2013-10-09	anak
2	16789	Hasan	laki-laki	2010-06-12	anak
3	45876	Husen	laki-laki	2011-07-08	anak

Lakukan perintah berikut untuk menampilkan data tanggungan yang mempunyai tahun kelahiran dari tahun 2010 sampai 2012

SQLQuery1.sql - (L...n (HP-PC\HP (57))*)

```

select * from tanggungan where YEAR (tgl_lahir) BETWEEN 2010 AND 2012
    
```

Results Messages

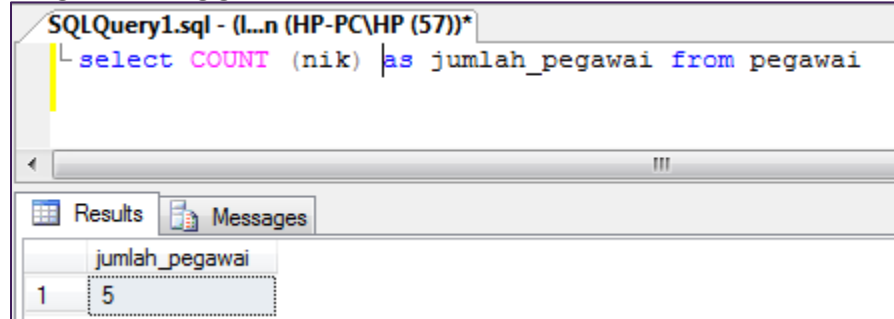
	nik	nama_anak	jenis_kelamin	tgl_lahir	hubungan
1	16789	Hasan	laki-laki	2010-06-12	anak
2	45876	Husen	laki-laki	2011-07-08	anak

7. FUNGSI AGGREGATE

Fungsi aggregate merupakan fungsi3fungsi yang digunakan untuk melakukan perhitungan statistik dalam tabel. Secara umum, hanya field yang bertipe numerik (int, money dan sejenisnya) yang bisa dikenakan fungsi ini.

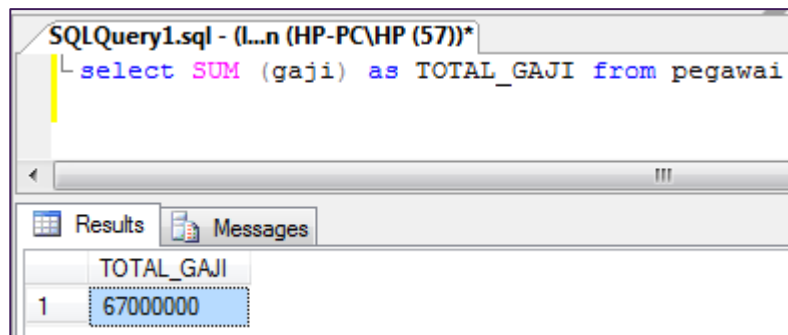
a. COUNT

Perintah yang digunakan untuk menghitung jumlah baris suatu kolom pada tabel. Perintah berikut digunakan untuk **menghitung jumlah data pada tabel pegawai dengan menggunakan kolom nik.**



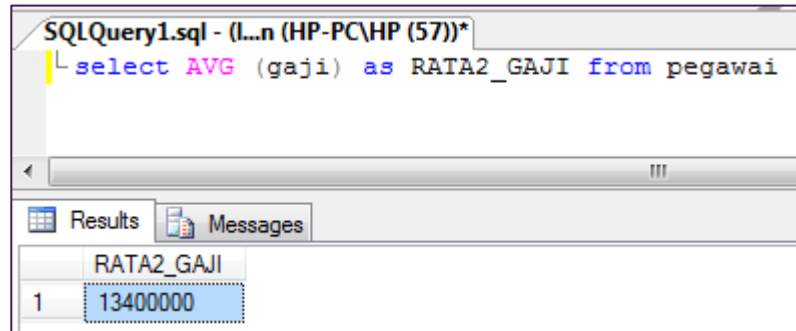
b. SUM

Perintah yang digunakan untuk menghitung jumlah nilai suatu kolom pada tabel. Perintah berikut digunakan untuk menghitung jumlah nilai kolom gaji pada tabel pegawai :



c. AVG

Perintah yang digunakan untuk menghitung rata-rata dari nilai suatu kolom pada tabel. Perintah berikut digunakan untuk menghitung rata-rata dari kolom Gaji pada tabel pegawai :

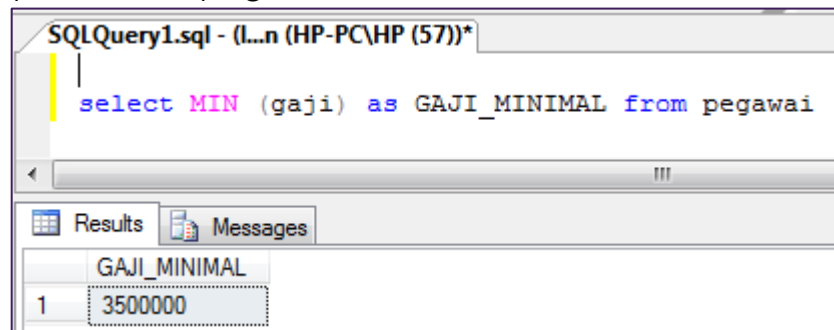


The screenshot shows a SQL Query window titled 'SQLQuery1.sql - (L...n (HP-PC\HP (57)))'. The query is: `select AVG (gaji) as RATA2_GAJI from pegawai`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, showing a table with one column 'RATA2_GAJI' and one row with the value '13400000'.

	RATA2_GAJI
1	13400000

d. MIN

Perintah yang digunakan untuk menampilkan nilai terkecil dari suatu kolom pada tabel. Perintah berikut digunakan untuk menampilkan nilai terkecil dari kolom gaji pada tabel pegawai :

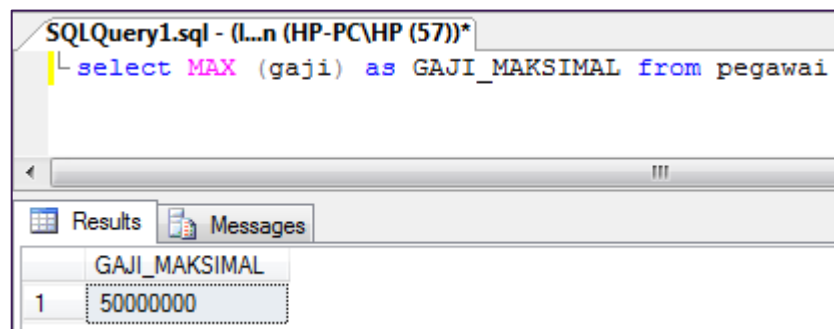


The screenshot shows a SQL Query window titled 'SQLQuery1.sql - (L...n (HP-PC\HP (57)))'. The query is: `select MIN (gaji) as GAJI_MINIMAL from pegawai`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, showing a table with one column 'GAJI_MINIMAL' and one row with the value '3500000'.

	GAJI_MINIMAL
1	3500000

e. MAX

Perintah yang digunakan untuk menampilkan nilai terbesar dari suatu kolom pada tabel. Perintah berikut digunakan untuk menampilkan nilai terbesar dari kolom Gaji pada tabel pegawai :



The screenshot shows a SQL Query window titled 'SQLQuery1.sql - (L...n (HP-PC\HP (57)))'. The query is: `select MAX (gaji) as GAJI_MAKSIMAL from pegawai`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, showing a table with one column 'GAJI_MAKSIMAL' and one row with the value '50000000'.

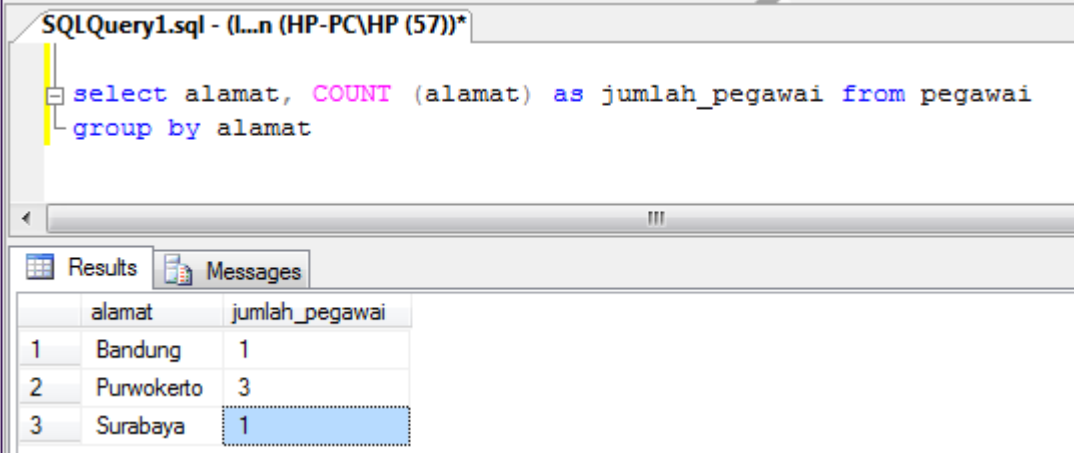
	GAJI_MAKSIMAL
1	50000000

8. KLAUSA GROUP BY

Hasil dari Select juga dapat dikelompokkan dengan menambahkan klausa GROUP BY. Perintah yang digunakan sebagai berikut :

```
SELECT nama kolom, COUNT (nama kolom)
FROM nama tabel
GROUP BY nama kolom;
```

Perintah berikut digunakan untuk menampilkan jumlah pegawai berdasarkan alamat kotanya :



The screenshot shows a SQL Query Editor window titled "SQLQuery1.sql - (L...n (HP-PC\HP (57)))". The query entered is:

```
select alamat, COUNT (alamat) as jumlah_pegawai from pegawai
group by alamat
```

Below the query editor, the "Results" tab is active, displaying a table with the following data:

	alamat	jumlah_pegawai
1	Bandung	1
2	Purwokerto	3
3	Surabaya	1

9. HAVING

Klausa HAVING disediakan untuk mendukung klausa GROUP BY. Kegunaannya adalah **untuk menentukan kondisi bagi GROUP BY**. Kelompok yang memenuhi kondisi HAVING saja yang akan dihasilkan. Perintah yang digunakan sebagai berikut :

```
SELECT nama_kolom, COUNT (nama_kolom)
FROM nama_tabel
GROUP BY nama_kolom
HAVING COUNT (nama_kolom) kondisi nilai;
```

TUGAS

1. Dengan menggunakan klausa Order By, tampilkan data nama, alamat dan nama departemen diurutkan berdasarkan nama secara Descending dan Alamat berdasarkan ascending.
2. Tampilkan nama dan alamat pegawai yang alamatnya bukan Purwokerto
3. Tampilkan Nomor, nama departemen dan jumlah pegawai untuk masing-masing departemen.
4. Lengkapi data pada tabel tanggungan dan tampilkan data pegawai yang mempunyai tidak mempunyai tanggungan.
5. Tampilkan data tanggungan yang tahun lahir diluar 2010 sampai 2015.
6. Tampilkan jumlah pegawai yang mempunyai alamat Purwokerto
7. Tampilkan nik, nama, alamat dan Jumlah tanggungan untuk masing-masing pegawai.