TUGAS NON COURSERA

PRAKTIKUM PERTEMUAN 10

Anisa Intania Putri

2024071010

PRAKTIKUM 1

1. **Isilah tabel jurusan dan tabel biodata yang anda buat dengan ketentuan sebagai berikut:**

   a. Tabel jurusan

```
CREATE TABLE jurusan (
  Kode_jurusan CHAR (4) PRIMARY KEY,
  Nama_jurusan VARCHAR(50) NOT NULL,
  Ketua_jurusan VARCHAR(50) NOT NULL
);

INSERT INTO jurusan (kode_jurusan, nama_jurusan,
ketua_jurusan) VALUES
('KD01', 'Sistem Informasi', 'Harnaningrum, S.Si'),
('KD02', 'Teknik Informatika', 'Enny Sela, S.Kom.,
M.Kom'),
('KD03', 'Tekhnik Komputer', 'Berta Bednar, S.Si,
M.T.');
```

Kode_jurusan	Nama_jurusan	Ketua_jurusan
KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
KD03	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
KD04	Teknik Elektro	Dr. Ir. Budi Santoso
KM01	Sistem Informasi	Harnaningrum, S.Si
NULL	NULL	NULL

*note tidak sesuai karena yang awal lupa di screenshoot, ini hasil setelah table di update

   b. Table biodata

```
CREATE TABLE biodata (
  No_mahasiswa CHAR (6) PRIMARY KEY,
  Kode_jurusan CHAR (4) NOT NULL,
  Nama_mahasiswa VARCHAR(50) NOT NULL,
  Alamat VARCHAR (100) NOT NULL,
  IPK DECIMAL (3, 2) NOT NULL,
  FOREIGN KEY (Kode_jurusan) REFERENCES
jurusan(Kode_jurusan)
);
select*from biodata;
```

```
INSERT INTO biodata (no_mahasiswa, kode_jurusan,
nama_mahasiswa, 2lamat, ipk) VALUES
('210089', 'KD01', 'Rina Gunawan', 'Denpasar',
3.00),
('210090', 'KD03', 'Gani Suprapto', 'Singaraja',
3.50),
('210012', 'KD02', 'Alexandra', 'Nusa dua', 3.00),
('210099', 'KD02', 'Nadine', 'Gianyar', 3.20),
('210002', 'KD01', 'Rizal Samurai', 'Denpasar',
3.70);
```

No_mahasiswa	Kode_jurusan	Nama_mahasiswa	Alamat	IPK
210002	KM01	Rizal Samurai	Denpasar	3.70
210012	KD02	Alexandra	Nusa dua	3.00
210089	KM01	Rina Gunawan Astuti	Denpasar	3.00
210090	KD03	Gani Suprapto	Singaraja	3.50
210099	KD02	Nadine	Gianyar	3.20

*note tidak sesuai karena yang awal lupa di screenshoot, ini hasil setelah table di update

2. **Masukkan data baru pada tabel biodata dengan kode jurusan "KD04"!**

```
INSERT INTO jurusan (kode_jurusan, nama_jurusan,
ketua_jurusan)
VALUES ('KD04', 'Teknik Elektro', 'Dr. Ir. Budi
Santoso');
INSERT INTO biodata (no_mahasiswa, kode_jurusan,
nama_mahasiswa, alamat, ipk)
VALUES ('210100', 'KD04', 'Sultan Hakim', 'Jakarta',
3.80);
```

No_mahasiswa	Kode_jurusan	Nama_mahasiswa	Alamat	IPK
210002	KM01	Rizal Samurai	Denpasar	3.70
210012	KD02	Alexandra	Nusa dua	3.00
210089	KM01	Rina Gunawan Astuti	Denpasar	3.00
210090	KD03	Gani Suprapto	Singaraja	3.50
210099	KD02	Nadine	Gianyar	3.20
210100	KD04	Sultan Hakim	Jakarta	3.80
NULL	NULL	NULL	NULL	NULL

3. **Dengan menggunakan perintah UPDATE, cobalah merubah isi tabel jurusan dan biodata pada basis data mahasiswa yang anda buat dengan ketentuan sebagai berikut :**
   a. Mengganti nama mahasiswa pada tabel biodata "Rina Gunawan" menjadi "Rina Gunawan Astuti"!

```
UPDATE biodata
SET Nama_mahasiswa = 'Rina Gunawan Astuti'
WHERE No_mahasiswa = '210089';
```

210089	KM01	Rina Gunawan Astuti	Denpasar	3.00

b. Mengganti kode jurusan pada tabel jurusan “KD01” menjadi “KM01”!

```
SET FOREIGN_KEY_CHECKS = 0; -- matiin FK check
biar bisa ubah parent key
UPDATE jurusan
SET Kode_jurusan = 'KM01'
WHERE Kode_jurusan = 'KD01';

UPDATE biodata
SET Kode_jurusan = 'KM01'
WHERE Kode_jurusan = 'KD01';
SET FOREIGN_KEY_CHECKS = 1; -- nyalain lagi
```

KM01	Sistem Informasi	Harnaningrum, S.Si
NULL	NULL	NULL

c. Mengganti no mahasiswa pada tabel biodata “210089” menjadi “210098”!

```
UPDATE biodata
SET No_mahasiswa = '210098'
WHERE No_mahasiswa = '210089';
```

210098	KM01	Rina Gunawan Astuti	Denpasar	3.00

d. Mengganti nilai pada tabel biodata “3” menjadi “3.3”!

```
UPDATE biodata
SET IPK = '3.3'
WHERE No_mahasiswa = '210098';

UPDATE biodata
SET IPK = 3.3
WHERE no_mahasiswa = '210012';
```

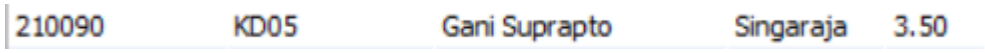
210012	KD02	Alexandra	Nusa dua	3.30
210090	KD05	Gani Suprapto	Singaraja	3.50
210098	KM01	Rina Gunawan Astuti	Denpasar	3.30

e. Mengganti kode jurusan pada tabel biodata “KD03” menjadi “KD05”!

```
SET FOREIGN_KEY_CHECKS = 0; -- matiin FK check
biar bisa ubah parent key
UPDATE jurusan
```

```
    SET Kode_jurusan = 'KD05'
    WHERE Kode_jurusan = 'KD03';

    UPDATE biodata
    SET Kode_jurusan = 'KD05'
    WHERE Kode_jurusan = 'KD03';
    SET FOREIGN_KEY_CHECKS = 1; -- nyalain lagi
```



4. Buatlah kesimpulan mengenai soal no.2 dan no.3!

- Kesimpulan No 2 jika ingin menambahkan data di table biodata dengan kode jurusan “KD04” kita harus masukkan dulu kode jururan yang baru (KD04) di table jurusan karena dia adalah table parent, baru kita bisa menambahkan data biodata mahasiswa di table biodata.
- Kesimpulan No 3 kita tidak bisa asal mengubah kode jurusan pada tabel jurusan “KD01” menjadi “KM01”, karena “KD01” masih dipakai oleh data mahasiswa di tabel biodata (Child). Walaupun kita sudah mengubah kode jurusan di table jurusan (Parent) terlebih dahulu itu tetap saja tidak bisa / error. Ibaratnya, kita mengganti kunci lemari (jurusan) padahal kuncinya masih nyantol di pintunya (biodata). Jadi solusinya kita gunakan fitur keamanan (`SET FOREIGN_KEY_CHECKS = 0`) untuk memaksa perubahan kunci di kedua tabel secara berurutan (ubah di table parent dulu baru ubah table child), lalu segera mengaktifkannya lagi (`SET FOREIGN_KEY_CHECKS = 1`).

PRAKTIKUM 2

1. **Cobalah masing-masing perintah join yang sudah anda praktikkan dengan menggunakan data base Mahasiswa yang anda buat!**
   a. **CROSS JOIN (Semua kombinasi antar tabel (tanpa syarat))**

```
SELECT biodata.no_mahasiswa, biodata.nama_mahasiswa,
jurusan.kode_jurusan, jurusan.nama_jurusan,
jurusan.ketua_jurusan
FROM biodata
CROSS JOIN jurusan;
```

no_mahasiswa	nama_mahasiswa	kode_jurusan	nama_jurusan	ketua_jurusan
210002	Rizal Samurai	KM01	Sistem Informasi	Harnaningrum, S.Si
210002	Rizal Samurai	KD05	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
210002	Rizal Samurai	KD04	Teknik Elektro	Dr. Ir. Budi Santoso
210002	Rizal Samurai	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210012	Alexandra	KM01	Sistem Informasi	Harnaningrum, S.Si
210012	Alexandra	KD05	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
210012	Alexandra	KD04	Teknik Elektro	Dr. Ir. Budi Santoso
210012	Alexandra	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210090	Gani Suprapto	KM01	Sistem Informasi	Harnaningrum, S.Si
210090	Gani Suprapto	KD05	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
210090	Gani Suprapto	KD04	Teknik Elektro	Dr. Ir. Budi Santoso

Notes: Query ini membuat tabel baru sementara yang berisi semua kombinasi mahasiswa dan jurusan (semua kombinasi antar tabel (tanpa syarat)). Hasilnya di dapat dari perkalian jumlah baris dari tabel biodata(6) × jurusan(4) = 24 baris kebawah.

   b. **INNER JOIN (Hanya data yang cocok di kedua tabel)**

```
SELECT
  biodata.no_mahasiswa,
  biodata.nama_mahasiswa,
  biodata.kode_jurusan,
  jurusan.nama_jurusan,
  jurusan.ketua_jurusan
FROM biodata
INNER JOIN jurusan
  ON biodata.kode_jurusan = jurusan.kode_jurusan;
```

no_mahasiswa	nama_mahasiswa	kode_jurusan	nama_jurusan	ketua_jurusan
210012	Alexandra	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210099	Nadine	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210100	Sultan Hakim	KD04	Teknik Elektro	Dr. Ir. Budi Santoso
210090	Gani Suprapto	KD05	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
210002	Rizal Samurai	KM01	Sistem Informasi	Harnaningrum, S.Si
210098	Rina Gunawan Astuti	KM01	Sistem Informasi	Harnaningrum, S.Si

Notes: Query ini membuat tabel baru sementara yang berisi data yang kode jurusannya sama antara biodata dan jurusan (hanya data yang cocok di kedua tabel), karena ada (`ON biodata.kode_jurusan = jurusan.kode_jurusan;`).

c. LEFT JOIN (Semua dari tabel kiri + cocok dari kanan)

```
SELECT
  biodata.no_mahasiswa,
  biodata.nama_mahasiswa,
  biodata.kode_jurusan,
  jurusan.nama_jurusan,
  jurusan.ketua_jurusan
FROM biodata
LEFT JOIN jurusan
  ON biodata.kode_jurusan = jurusan.kode_jurusan;
```

no_mahasiswa	nama_mahasiswa	kode_jurusan	nama_jurusan	ketua_jurusan
210002	Rizal Samurai	KM01	Sistem Informasi	Harnaningrum, S.Si
210012	Alexandra	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210090	Gani Suprapto	KD05	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
210098	Rina Gunawan Astuti	KM01	Sistem Informasi	Harnaningrum, S.Si
210099	Nadine	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210100	Sultan Hakim	KD04	Teknik Elektro	Dr. Ir. Budi Santoso

*Notes: Query ini menampilkan semua data dari tabel kiri (biodata), dan menambahkan data dari tabel kanan (jurusan) jika kode jurusannya sama. Kalau tidak ada data yang cocok di tabel jurusan, maka kolom dari jurusan akan berisi NULL.

d. RIGHT JOIN (Semua dari tabel kanan + cocok dari kiri)

```
SELECT
  biodata.no_mahasiswa,
  biodata.nama_mahasiswa,
  biodata.kode_jurusan,
  jurusan.nama_jurusan,
  jurusan.ketua_jurusan
FROM biodata
RIGHT JOIN jurusan
  ON biodata.kode_jurusan = jurusan.kode_jurusan;
```

no_mahasiswa	nama_mahasiswa	kode_jurusan	nama_jurusan	ketua_jurusan
210012	Alexandra	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210099	Nadine	KD02	Teknik Informatika	Enny Sela, S.Kom., M.Kom
210100	Sultan Hakim	KD04	Teknik Elektro	Dr. Ir. Budi Santoso
210090	Gani Suprapto	KD05	Tekhnik Komputer	Berta Bednar, S.Si, M.T.
210002	Rizal Samurai	KM01	Sistem Informasi	Harnaningrum, S.Si
210098	Rina Gunawan Astuti	KM01	Sistem Informasi	Harnaningrum, S.Si

*Notes: Query ini menampilkan semua data dari tabel kanan (jurusan), dan menambahkan data dari tabel kiri (biodata) jika kode jurusannya sama. Kalau tidak ada data yang cocok di tabel biodata, maka kolom dari biodata akan berisi NULL.

2. **Buatlah kesimpulan tentang perbedaan klausa antaran Right join dan Left join!**
    - LEFT JOIN menampilkan semua data dari tabel kiri (tabel yang disebut setelah FROM), dan hanya menampilkan data dari tabel kanan jika ada yang cocok.
    - RIGHT JOIN kebalikannya, menampilkan semua data dari tabel kanan (tabel setelah JOIN), dan hanya menampilkan data dari tabel kiri jika ada yang cocok.
    - Jika tidak ada data yang cocok, maka kolom dari tabel lain akan berisi **NULL**.

PRAKTIKUM 3

1. **Buatlah perintah SQL yang menggunakan single row function (Masing-masing 1)!**
   a. **LOWER (Mengubah teks menjadi huruf kecil semua.)**

```
SELECT LOWER (Nama_mahasiswa) AS Huruf_kecil
FROM biodata;
```

Huruf_kecil
rizal samurai
alexandra
gani suprapto
rina gunawan astuti
nadine
sultan hakim

   b. **UPPER (Mengubah teks menjadi huruf besar semua.)**

```
SELECT UPPER (Nama_mahasiswa) AS Huruf_kecil
FROM biodata;
```

Huruf_kecil
RIZAL SAMURAI
ALEXANDRA
GANI SUPRAPTO
RINA GUNAWAN ASTUTI
NADINE
SULTAN HAKIM

   c. **SUBSTRING (Mengambil nama mahasiswa yang dimulai dari huruf kedua sebanyak lima huruf)**

```
SELECT SUBSTRING(Nama_mahasiswa, 2, 5) AS
potongan_nama
FROM biodata;
```

potongan_nama
izal
lexan
ani S
ina G
adine
ultan

   d. **LTRIM (Menghapus spasi di sebelah kiri teks.)**

```
SELECT LTRIM('  Alexandra') AS hapus_spasi_kiri
FROM biodata;
```

hapus_spasi_kiri
Alexandra

e. RTRIM (Menghapus spasi di sebelah kanan teks.)

```
SELECT RTRIM('Alexandra  ') AS hapus_spasi_kanan
FROM biodata;
```

hapus_spasi_kiri
Alexandra

f. RIGHT (Mengambil 4 karakter nama mahasiswa dari sebelah kanan)

```
SELECT RIGHT(Nama_mahasiswa, 4) AS potongan_nama
FROM biodata;
```

potongan_nama
urai
ndra
apto
tuti
dine
akim

g. LEFT (Mengambil 4 karakter nama mahasiswa dari sebelah kiri)

```
SELECT LEFT(Nama_mahasiswa, 4) AS potongan_nama
FROM biodata;
```

potongan_nama
Riza
Alex
Gani
Rina
Nadi
Sult

h. LENGTH (Menghitung banyak karakter pembentuk nama barang termasuk spasinya.)

```
SELECT Nama_mahasiswa, LENGTH (Nama_mahasiswa) AS
banyak_karakter
```

```
    FROM biodata;
```

Nama_mahasiswa	banyak_karakter
Rizal Samurai	13
Alexandra	9
Gani Suprapto	13
Rina Gunawan Astuti	19
Nadine	6
Sultan Hakim	12

i. REVERSE (Membalik nama mahasiswa)

```
    SELECT Nama_mahasiswa, REVERSE(Nama_mahasiswa) AS
    balik
    FROM biodata;
```

Nama_mahasiswa	balik
Rizal Samurai	iarumaS laziR
Alexandra	ardnaxelA
Gani Suprapto	otparpuS inaG
Rina Gunawan Astuti	itutsA nawanuG aniR
Nadine	enidaN
Sultan Hakim	mikaH natluS

j. SPACE (Memberikan spasi sebanyak 3 spasi)

```
SELECT SPACE (3) AS spasi;
```

spasi

PRAKTIKUM 4

1. **Buatlah perintah SQL yang menggunakan fungsi Aggregate (Masing-masing 1)!**
   a. **AVG (Menampilkan rata-rata IPK dari tabel Biodata)**

```
SELECT AVG(IPK) AS 'rata_IPK'
FROM biodata;
```

rata_IPK
3.466667

   b. **COUNT (Menampilkan jumlah seluruh mahasiswa yang ada pada tabel Biodata)**

```
SELECT COUNT(*) AS 'jumlah_mahasiswa'
FROM biodata;
```

jumlah_mahasiswa
6

   c. **SUM (Menjumlahkan total nilai IPK pada tabel Biodata)**

```
SELECT SUM(IPK) AS 'total_IPK'
FROM biodata;
```

total_IPK
20.80

   d. **MAX (Menampilkan nilai IPK paling besar dari tabel Biodata)**

```
SELECT MAX(IPK) AS 'IPK_tertinggi'
FROM biodata;
```

IPK_tertinggi
3.80

   e. **MIN (Menampilkan nilai IPK paling kecil dari tabel Biodata)**

```
SELECT MIN(IPK) AS 'IPK_terendah'
FROM biodata;
```

IPK_terendah
3.20

2. **Buatlah perintah SQL dengan klausa Order By, Group By, dan Having!**
   a. **Order By**
      - **ASC (Menampilkan data pada tabel Biodata dengan urutan ASC atau terurut secara abjad)**

-

```
SELECT *
FROM biodata
ORDER BY nama_mahasiswa ASC;
```

No_mahasiswa	Kode_jurusan	Nama_mahasiswa	Alamat	IPK
210012	KD02	Alexandra	Nusa dua	3.30
210090	KD05	Gani Suprapto	Singaraja	3.50
210099	KD02	Nadine	Gianyar	3.20
210098	KM01	Rina Gunawan Astuti	Denpasar	3.30
210002	KM01	Rizal Samurai	Denpasar	3.70
210100	KD04	Sultan Hakim	Jakarta	3.80
NULL	NULL	NULL	NULL	NULL

- **DESC (Menampilkan data pada tabel Biodata dengan urutan DESC atau terurut secara abjad terbalik)**

```
SELECT *
FROM biodata
ORDER BY nama_mahasiswa DESC;
```

No_mahasiswa	Kode_jurusan	Nama_mahasiswa	Alamat	IPK
210100	KD04	Sultan Hakim	Jakarta	3.80
210002	KM01	Rizal Samurai	Denpasar	3.70
210098	KM01	Rina Gunawan Astuti	Denpasar	3.30
210099	KD02	Nadine	Gianyar	3.20
210090	KD05	Gani Suprapto	Singaraja	3.50
210012	KD02	Alexandra	Nusa dua	3.30
NULL	NULL	NULL	NULL	NULL

b. Group By (Mengelompokkan data mahasiswa berdasarkan kode jurusan, dan menampilkan jumlah mahasiswa per jurusan)

```
SELECT kode_jurusan, COUNT(nama_mahasiswa) AS
jumlah_mahasiswa
FROM biodata
GROUP BY kode_jurusan;
```

kode_jurusan	jumlah_mahasiswa
KD02	2
KD04	1
KD05	1
KM01	2

c. Having (Menampilkan jurusan yang memiliki lebih dari 1 mahasiswa)

```
SELECT kode_jurusan, COUNT(nama_mahasiswa) AS
jumlah_mahasiswa
FROM biodata
```

```
    GROUP BY kode_jurusan
    HAVING COUNT(nama_mahasiswa) > 1;
```

kode_jurusan	jumlah_mahasiswa
KD02	2
KM01	2

PRAKTIKUM 5

1. **Buatlah perintah MySQL masing-masing pada basis data mahasiswa dengan menggunakan subqueries dan set operation!**
   a. **Subquery MAX (Menampilkan mahasiswa yang memiliki IPK tertinggi)**

```
SELECT No_mahasiswa, Nama_mahasiswa, IPK
FROM biodata
WHERE IPK = (
  SELECT MAX(IPK)
  FROM biodata);
```

No_mahasiswa	Nama_mahasiswa	IPK
210100	Sultan Hakim	3.80
NULL	NULL	NULL

   b. **Subquery MIN (Menampilkan mahasiswa yang memiliki IPK terkecil)**

```
SELECT No_mahasiswa, Nama_mahasiswa, IPK
FROM biodata
WHERE IPK = (
  SELECT MIN(IPK)
  FROM biodata);
```

No_mahasiswa	Nama_mahasiswa	IPK
210099	Nadine	3.20
NULL	NULL	NULL

   c. **Subquery IN (Menampilkan mahasiswa yang terdaftar pada jurusan yang diketuai oleh "`Dr. Ir. Budi Santoso`")**

```
SELECT Nama_mahasiswa, Kode_jurusan
FROM biodata
WHERE kode_jurusan IN (
  SELECT kode_jurusan
  FROM jurusan
  WHERE ketua_jurusan = 'Dr. Ir. Budi Santoso');
```

Nama_mahasiswa	Kode_jurusan
Sultan Hakim	KD04

   d. **UNION (Menampilkan nama mahasiswa dari tabel biodata dan nama jurusan dari tabel jurusan)**

```
SELECT nama_mahasiswa AS nama
FROM biodata
UNION
SELECT nama_jurusan AS nama
FROM jurusan;
```

nama
Rizal Samurai
Alexandra
Gani Suprapto
Rina Gunawan Astuti
Nadine
Sultan Hakim
Teknik Informatika
Teknik Elektro
Tekhnik Komputer
Sistem Informasi

e. UNION ALL (Menampilkan semua, termasuk yang duplikat)

```
SELECT nama_mahasiswa AS nama
FROM biodata
UNION ALL
SELECT nama_jurusan AS nama
FROM jurusan;
```

nama
Rizal Samurai
Alexandra
Gani Suprapto
Rina Gunawan Astuti
Nadine
Sultan Hakim
Teknik Informatika
Teknik Elektro
Tekhnik Komputer
Sistem Informasi

2. Buatlah perintah MySQL yang menghasilkan output INTERSECT dan EXCEPT dengan menggunakan fungsi UNION!

a. INTERSECT dengan UNION (Menampilkan nama dan kode jurusan dari tabel biodata yang juga muncul pada tabel jurusan)

```
SELECT nama_mahasiswa, kode_jurusan FROM biodata
WHERE kode_jurusan IN (
  SELECT kode_jurusan FROM jurusan )
```

```
UNION ALL
SELECT nama_jurusan, kode_jurusan
FROM jurusan
WHERE kode_jurusan IN (
  SELECT kode_jurusan FROM biodata);
```

nama_mahasiswa	kode_jurusan
Alexandra	KD02
Nadine	KD02
Sultan Hakim	KD04
Gani Suprapto	KD05
Rizal Samurai	KM01
Rina Gunawan Astuti	KM01
Teknik Informatika	KD02
Teknik Elektro	KD04
Tekhnik Komputer	KD05
Sistem Informasi	KM01

b. EXCEPT dengan UNION (Menampilkan nama mahasiswa dan kode jurusan dari tabel biodata yang tidak ada di tabel jurusan)

```
SELECT nama_mahasiswa, kode_jurusan FROM biodata
WHERE kode_jurusan NOT IN (
  SELECT kode_jurusan FROM jurusan )
UNION ALL
SELECT nama_jurusan, kode_jurusan
FROM jurusan
WHERE kode_jurusan NOT IN (
  SELECT kode_jurusan FROM biodata);
```

nama_mahasiswa	kode_jurusan

3. Perhatikan output-nya dan buatlah kesimpulan mengenai perbedaan antara perintah masing-masing!

- Subquery adalah query di dalam query lain. Biasanya dipakai untuk mencari data berdasarkan hasil dari query lain, seperti mencari nilai tertinggi, terendah, atau data yang memenuhi kondisi tertentu.
- UNION digunakan untuk menggabungkan dua hasil query dengan jumlah kolom dan tipe data yang sama hasilnya tidak menampilkan data duplikat.
- UNION ALL mirip dengan UNION, tapi menampilkan semua data termasuk duplikat.
- INTERSECT menampilkan data yang sama di kedua query, sedangkan EXCEPT menampilkan data yang hanya ada di query pertama tapi tidak di query kedua.

PRAKTIKUM 6

1. **Sebutkan fungsi dari View!**
   View bisa dibilang sebagai table virtual yang berisi hasil dari suatu query yang fungsinya untuk menyederhanakan query yang kompleks, membatasi akses data karena hanya kolom tertentu yang ditampilkan bukan semua, dapat digunakan seperti table biasa (bisa di SELECT, JOIN, dll), otomatis menyesuaikan jika ada data yang berubah pada table aslinya.

2. **Buatlah view pada basis data "Mahasiswa" yang anda buat!**

```
CREATE VIEW laporan_mahasiswa AS
SELECT biodata.No_mahasiswa,
       biodata.Nama_mahasiswa,
       biodata.kode_jurusan,
       jurusan.nama_jurusan
FROM biodata, jurusan
WHERE biodata.kode_jurusan = jurusan.kode_jurusan;
SELECT * FROM laporan_mahasiswa;
```

No_mahasiswa	Nama_mahasiswa	kode_jurusan	nama_jurusan
210012	Alexandra	KD02	Teknik Informatika
210099	Nadine	KD02	Teknik Informatika
210100	Sultan Hakim	KD04	Teknik Elektro
210090	Gani Suprapto	KD05	Tekhnik Komputer
210002	Rizal Samurai	KM01	Sistem Informasi
210098	Rina Gunawan Astuti	KM01	Sistem Informasi

3. **Buatlah control flow functiom dan tabel temporary pada basis data "Mahasiswa" dengan menggunakan ekspresi berbentuk perintah DML SELECT!**

   CONTROL FLOW FUNCTION

   a. **IF**

```
SELECT IF(24 > 20, 'Beban studi normal', 'Beban studi
ringan') AS hasil;
```

hasil
Beban studi normal

   b. **IFNULL**

```
SELECT IFNULL(
  (SELECT nama_mahasiswa FROM biodata WHERE
No_mahasiswa = 'M999'),
  'Tidak Ditemukan') AS hasil;
```

hasil
Tidak Ditemukan

c. NULLIF

```
SELECT NULLIF(
  (SELECT nama_jurusan FROM jurusan WHERE
kode_jurusan = 'KM01'),
  'Sistem Informasi') AS hasil;
```

hasil
NULL

d. CASE

```
SELECT No_mahasiswa, Nama_mahasiswa,
Kode_jurusan,
CASE
  WHEN Kode_jurusan = 'KD02' THEN 'Teknik
Informatika'
  WHEN Kode_jurusan = 'KM01' THEN 'Sistem
Informasi'
  WHEN Kode_jurusan = 'KD06' THEN 'Desain
Komunikasi Visual'
  ELSE 'Jurusan Lain'
END AS Keterangan_Jurusan
FROM biodata;
```

No_mahasiswa	Nama_mahasiswa	Kode_jurusan	Keterangan_Jurusan
210002	Rizal Samurai	KM01	Sistem Informasi
210012	Alexandra	KD02	Teknik Informatika
210090	Gani Suprapto	KD05	Jurusan Lain
210098	Rina Gunawan Astuti	KM01	Sistem Informasi
210099	Nadine	KD02	Teknik Informatika
210100	Sultan Hakim	KD04	Jurusan Lain

e. Table Temporary

```
CREATE TEMPORARY TABLE temp_mahasiswa (
  No INT PRIMARY KEY,
  nama VARCHAR(80),
  alamat VARCHAR(100),
  jurusan VARCHAR(50)
);

SELECT * FROM temp_mahasiswa;
```

No	nama	alamat	jurusan

PRAKTIKUM 7

1. **Buatlah sebuah stored procedure pada basis data "Mahasiswa" yang anda buat untuk menampilkan semua data mahsiswa berdasarkan ketua jurusannya!**

```
CREATE PROCEDURE tampil_mahasiswa_ketua (IN nama_ketua
VARCHAR(50))
BEGIN
  SELECT biodata.No_mahasiswa, biodata.Nama_mahasiswa,
jurusan.Nama_jurusan, jurusan.Ketua_jurusan
  FROM biodata
  INNER JOIN jurusan ON biodata.Kode_jurusan =
jurusan.Kode_jurusan
  WHERE jurusan.Ketua_jurusan = nama_ketua;
END //
DELIMITER ;

CALL tampil_mahasiswa_ketua('Harnaningrum, S.Si');
```

No_mahasiswa	Nama_mahasiswa	Nama_jurusan	Ketua_jurusan
210002	Rizal Samurai	Sistem Informasi	Harnaningrum, S.Si
210098	Rina Gunawan Astuti	Sistem Informasi	Harnaningrum, S.Si

2. **Buatlah stored procedure yang akan mencari nama ketua jurusan dengan nama tertentu!**

```
DELIMITER //
CREATE PROCEDURE cari_ketua_jurusan (IN nama_ketua
VARCHAR(50))
BEGIN
  SELECT Kode_jurusan, Nama_jurusan, Ketua_jurusan
  FROM jurusan
  WHERE Ketua_jurusan LIKE CONCAT('%', nama_ketua,
'%');
END //
DELIMITER ;

CALL cari_ketua_jurusan('Budi');
```

Kode_jurusan	Nama_jurusan	Ketua_jurusan
KD04	Teknik Elektro	Dr. Ir. Budi Santoso

3. **Buatlah sebuah trigger pada basis data "Mahasiswa"!**

```
CREATE TABLE log_mahasiswa (
  id_log INT AUTO_INCREMENT PRIMARY KEY,
  no_mahasiswa VARCHAR(10),
```

```
    nama_mahasiswa VARCHAR(100),
    created_at DATETIME
);
```

```
DELIMITER //
CREATE TRIGGER after_insert_biodata
AFTER INSERT ON biodata
FOR EACH ROW
BEGIN
INSERT INTO log_mahasiswa (no_mahasiswa,
nama_mahasiswa, created_at)
VALUES (NEW.no_mahasiswa, NEW.nama_mahasiswa, NOW());
END //
DELIMITER ;
```

```
INSERT INTO biodata (no_mahasiswa, kode_jurusan,
nama_mahasiswa, alamat, ipk)
VALUES ('202407', 'KM01', 'Anisa', 'Tangsel', 4.0);
```

```
SHOW TRIGGERS;
```

Trigger	Event	Table	Statement	Timing	Created	sql_mode	Definer
after_insert_biodata	INSERT	biodata	BEGIN INSERT INTO log_mahasiswa (no_maha...	AFTER	2025-11-12 22:45:44.37	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@localhost

4. **Apakah perbedaan antara stored procedure dengan fungsi?**
   Intinya kalua fungsi itu harus mengembalikan nilai, kayak fungsi di matematika atau di bahasa pemrograman biasa. Dia dirancang buat menghitung sesuatu dan ngasih hasilnya balik. Sementara Stored Procedure (sering dipanggil SP aja) itu tidak harus mengembalikan nilai, dia lebih fokus ke melakukan suatu aksi atau serangkaian tugas di database. Fungsi bisa dipanggil dengan SELECT, WHERE, HAVING, kalua Store Procedure biasanya dipanggil dengan EXECUTE dan EXEC. Fungsi tidak bisa melakukan perubahan pada database (alias INSERT, UPDATE, DELETE). Dia cuma boleh baca data (SELECT). Tapi, kalau Store procedure bisa melakukan INSERT, UPDATE, DELETE, dan juga SELECT.