Tugas Laporan Fundamental Database

Irfan Luthfiardi Anhar

24110300007

Advance Database

1. Tugas 1: Tampilkan 5 mahasiswa dengan IPK tertinggi, urutkan IPK descending, nama ascending

Input query:

```
--Tugas 1: Tampilkan 5 mahasiswa dengan IPK tertinggi, urutkan IPK descending, nama ascending
SELECT *
FROM mahasiswa
ORDER BY ipk DESC, nama ASC
LIMIT 5;
```

Disini untuk menampilkan 5 mahasiswa dengan ipk tertinggi dan mengurutkan IPK descending dan nama yang ascending maka saya menggunakan syntax berikut:

- SELECT * FROM mahasiswa, yaitu untuk memanggil apa saja yang ada di tabel mahasiswa.
- ORDER BY, yaitu kaya sama aja dengan sort by atau urutanya.
- Ipk DESC, yaitu artinya urutan ipk nya descendan atau dari yang terbesar hingga yang terkecil.
- Nama ASC, yaitu artinya urutan namanya ascendan atau dari yang terkecil hingga terbesar, untuk konteks ini maka Jika ada IPK yang sama, data akan diurutkan berdasarkan nama secara alfabet (A ke Z).
- LIMIT 5, itu artinya kita hanya maksimal menampilkan 5 data saja dari yang tertinggi.

Output Query:

	nim	nama	email	tanggal_lahir	ipk	semester
1	2021006	Abigail Rachel	rachel@gmail.com	2002-08-15	4	2
2	2021012	Andi Wijaya	andi@gmail.com	2002-04-18	3.95	4
3	2021004	Oline Manuel	oline@gmail.com	2002-11-03	3.9	3
4	2021010	Kevin Jonathan	kevin@gmail.com	2003-02-14	3.85	3
5	2021001	Ahmad Sutrisno	ahmad@email.com	2000-05-15	3.8	5

2. Tugas 2: Hitung berapa mahasiswa yang lahir di setiap tahun, tampilkan hanya tahun yang memiliki lebih dari 10 mahasiswa.

Input query:

```
--Tugas 2: Hitung berapa mahasiswa yang lahir di setiap tahun, tampilkan hanya tahun dengan > 10 mahasiswa
SELECT EXTRACT(YEAR FROM tanggal_lahir) AS tahun, COUNT(*) AS jumlah_mahasiswa
FROM mahasiswa
GROUP BY EXTRACT(YEAR FROM tanggal_lahir)
HAVING COUNT(*) > 10;
```

Disini untuk menghitung ada berapa mahasiswa yang lahir disetiap tahun yang sama maka saya menggunakan syntax berikut:

- SELECT EXTRACT(YEAR FROM tanggal_lahir) AS tahun, yaitu disini saya mengambil tahun nya saja dari kolom tanggal lahir untuk setiap mahasiswa.
- COUNT(*) AS jumlah_mahasiswa, yaitu untuk menghitung jumlah mahasiswa yang lahir pada tahun tersebut.
- FROM mahasiswa, yaitu mengambil data dari tabel mahasiswa.
- GROUP BY EXTRACT(YEAR FROM tanggal_lahir), yaitu untuk mengelompokkan atau mengurutkan data mahasiswa berdasarkan tahun lahir agar bisa dihitung jumlahnya per tahun.
- HAVING COUNT(*) > 10, yaitu untuk menampilkan hanya tahun-tahun yang memiliki lebih dari 10 mahasiswa.

Output Query:

	123 tahun	123 jumlah_mahasiswa
1	2,001	15

Artinya ada 15 mahasiswa yang lahir ditahun yang sama yaitu 2001.

3. Tugas 3: Update semester semua mahasiswa yang memiliki IPK >= 3.5 untuk naik 1 semester.

Input query:

```
--Tugas 3: Update semester semua mahasiswa yang memiliki IPK >= 3.5 untuk naik 1 semester
UPDATE mahasiswa
SET semester = semester + 1
WHERE ipk >= 3.5;
```

Disini untuk melakukan update semester bagi semua mahasiswa yang memiliki IPK >= 3.5 untuk naik 1 semester, maka saya menggunakan syntax berikut:

- UPDATE mahasiswa, yaitu disini saya menandakan bahwa kita akan melakukan perubahan data pada tabel mahasiswa.
- SET semester = semester + 1, yaitu kolom semester akan ditambah 1 jika setiap mahasiswa memenuhi kondisi yang kita tentukan.
- WHERE ipk >= 3.5, Yaitu disini kita membuat kondisi agar hanya mahasiswa dengan IPK ≥ 3.5 yang semesternya dinaikkan dengan menggunakan syntax tadi.

Output query:

Before execute:

	nim	nama	email	tanggal_lahir	ipk	semester
1	2021006	Abigail Rachel	rachel@gmail.com	2002-08-15	4	2
2	2021012	Andi Wijaya	andi@gmail.com	2002-04-18	3.95	4
3	2021004	Oline Manuel	oline@gmail.com	2002-11-03	3.9	3
4	2021010	Kevin Jonathan	kevin@gmail.com	2003-02-14	3.85	3
5	2021001	Ahmad Sutrisno	ahmad@email.com	2000-05-15	3.8	5
6	2021002	Siti Nurhaliza	siti@email.com	2001-03-20	3.75	4
7	2021021	Lia Nuraini	lia@gmail.com	2001-06-13	3.6	4
8	2021013	Nabila Putri	nabila@gmail.com	2001-12-29	3.6	5
9	2021019	Joko Santoso	joko@gmail.com	2001-04-17	3.55	4
10	2021003	Budi Santoso	budi@email.com	2000-12-10	3.5	4

After execute:

	nim	nama	email	tanggal_lahir	ipk	semester
1	2021006	Abigail Rachel	rachel@gmail.com	2002-08-15	4	3
2	2021012	Andi Wijaya	andi@gmail.com	2002-04-18	3.95	5
3	2021004	Oline Manuel	oline@gmail.com	2002-11-03	3.9	4
4	2021010	Kevin Jonathan	kevin@gmail.com	2003-02-14	3.85	4
5	2021001	Ahmad Sutrisno	ahmad@email.com	2000-05-15	3.8	6
6	2021002	Siti Nurhaliza	siti@email.com	2001-03-20	3.75	5
7	2021021	Lia Nuraini	lia@gmail.com	2001-06-13	3.6	5
8	2021013	Nabila Putri	nabila@gmail.com	2001-12-29	3.6	6
9	2021019	Joko Santoso	joko@gmail.com	2001-04-17	3.55	5
10	2021003	Budi Santoso	budi@email.com	2000-12-10	3.5	5

4. Tugas 4: Buat query untuk menampilkan semua jurusan beserta jumlah mahasiswanya (termasuk jurusan yang belum punya mahasiswa).

Input query:

```
--Tugas 4: Buat query untuk menampilkan semua jurusan beserta jumlah mahasiswanya (termasuk jurusan yang belum punya mahasiswa).
SELECT j.nama_jurusan, COUNT(m.nim) AS jumlah_mahasiswa
FROM jurusan j
LEFT JOIN mahasiswa m ON m.jurusan_id = j.id
GROUP BY j.nama_jurusan;
```

Disini untuk menampilkan semua jurusan beserta jumlah mahasiswanya dan tidak tekecuali untuk jurusan yang belum punya mahasiswanya, maka saya menggunakan syntax berikut:

- SELECT j.nama_jurusan, COUNT(m.nim) AS jumlah_mahasiswa, Yaitu Fungsinya untuk menampilkan nama jurusan dan jumlah mahasiswa per jurusan.
- FROM jurusan j, yaitu saya mengambil semua data jurusan dan "J" nya itu alias dari tabel jurusan.
- LEFT JOIN mahasiswa m ON m.jurusan_id = j.id , maksudnya adalah gabungkan dengan mahasiswa, tapi tetap menampilkan jurusan tak terkecuali bagi jurusan yang tidak memiliki mahasiswa.
- GROUP BY j.nama_jurusan , Yaitu disini saya mengelompokkan hasil per jurusan supaya COUNT bisa dihitung.

Output query:

	A-Z nama_jurusan	123 jumlah_mahasiswa
1	Manajemen	11
2	Informatika	8
3	Sistem Informasi	7

5. Tugas 5: Tampilkan mahasiswa yang mengambil mata kuliah dengan SKS tertinggi di jurusannya.

Input query:

```sql
--Tugas 5: Tampilkan mahasiswa yang mengambil mata kuliah dengan SKS tertinggi di jurusannya.
SELECT m.nim, m.nama, mk.nama_mk, mk.sks, j.nama_jurusan
FROM mahasiswa m
INNER JOIN jurusan j ON m.jurusan_id = j.id
INNER JOIN enrollment e ON m.nim = e.nim
INNER JOIN mata_kuliah mk ON e.kode_mk = mk.kode
WHERE mk.sks = (
    SELECT MAX(mk2.sks)
    FROM mata_kuliah mk2
    WHERE mk2.jurusan_id = m.jurusan_id
);
```

Disini untuk menampilkan semua jurusan beserta jumlah mahasiswanya dan tidak tekecuali untuk jurusan yang belum punya mahasiswanya, maka saya menggunakan syntax berikut:

- INNER JOIN, Fungsinya untuk menggabungkan mahasiswa, jurusan, enrollment, dan mata kuliah agar semua info muncul dalam satu tabel.
- WHERE mk.sks = (...), Fungsinya buat memfilter mata kuliah yang memiliki SKS tertinggi di jurusan mahasiswa tersebut.
- Subquery SELECT MAX(mk2.sks) , Fungsinya untuk menentukan nilai SKS tertinggi di jurusan yang sama.

Output query:

	nim	nama	nama_mk	sks	nama_jurusan
1	2021001	Ahmad Sutrisno	Pemrograman Dasar	3	Informatika
2	2021002	Siti Nurhaliza	Basis Data	3	Informatika
3	2021003	Budi Santoso	Sistem Informasi	3	Sistem Informasi
4	2021004	Oline Manuel	Pemrograman Dasar	3	Informatika
5	2021005	Muhammad Ali	Sistem Informasi	3	Sistem Informasi
6	2021006	Abigail Rachel	Basis Data	3	Informatika
7	2021007	Daniel Prasetyo	Pemrograman Dasar	3	Informatika
8	2021008	Clara Angelina	Basis Data	3	Informatika
9	2021009	Rizky Ramadhan	Sistem Informasi	3	Sistem Informasi
10	2021010	Kevin Jonathan	Pemrograman Dasar	3	Informatika
11	2021011	Lestari Ayu	Basis Data	3	Informatika
12	2021012	Andi Wijaya	Sistem Informasi	3	Sistem Informasi
13	2021013	Nabila Putri	Pemrograman Dasar	3	Sistem Informasi
14	2021014	Jonathan Surya	Basis Data	3	Sistem Informasi
15	2021015	Maya Fitri	Sistem Informasi	3	Sistem Informasi

6. Tugas 6: Buat query self-join untuk menampilkan pasangan mahasiswa yang berasal dari jurusan yang sama.

Input query:

```
--Tugas 6: Buat query self-join untuk menampilkan pasangan mahasiswa yang berasal dari jurusan yang sama.
SELECT m1.nim AS nim1, m1.nama AS nama1,
       m2.nim AS nim2, m2.nama AS nama2,
       j.nama_jurusan
FROM mahasiswa m1
INNER JOIN mahasiswa m2
    ON m1.jurusan_id = m2.jurusan_id
    AND m1.nim < m2.nim
INNER JOIN jurusan j ON m1.jurusan_id = j.id
ORDER BY j.nama_jurusan, m1.nim, m2.nim;
```

Disini untuk menampilkan pasangan mahasiswa yang berasal dari jurusan yang sama, maka saya menggunakan syntax berikut:

- FROM mahasiswa m1, Saya menggunakan m1 sebagai “mahasiswa pertama” dalam pasangan.
- INNER JOIN mahasiswa m2 ON m1.jurusan_id = m2.jurusan_id AND m1.nim < m2.nim
    - m2 adalah “mahasiswa kedua” dalam pasangan.
    - m1.nim < m2.nim , Fungsinya buat mencegah duplikat pasangan dan agar mahasiswa tidak dipasangkan dengan dirinya sendiri.
- SELECT m1.nim AS nim1, m1.nama AS nama1, m2.nim AS nim2, m2.nama AS nama2
    - nim1 & nama1 ,yaitu informasi mahasiswa pertama.
    - nim2 & nama2 ,yaitu informasi mahasiswa kedua.
- INNER JOIN jurusan j ON m1.jurusan_id = j.id ,yang Fungsinya untuk menampilkan nama jurusan dari pasangan mahasiswa.
- ORDER BY j.nama_jurusan, m1.nim, m2.nim , yaitu untuk mengurutkan hasil supaya lebih rapih per jurusan dan per pasangan mahasiswa.

Output query:

	nim1	nama1	nim2	nama2	nama_jurusan
1	2021001	Ahmad Sutrisno	2021002	Siti Nurhaliza	Informatika
2	2021001	Ahmad Sutrisno	2021004	Oline Manuel	Informatika
3	2021001	Ahmad Sutrisno	2021006	Abigail Rachel	Informatika
4	2021001	Ahmad Sutrisno	2021007	Daniel Prasetyo	Informatika
5	2021001	Ahmad Sutrisno	2021008	Clara Angelina	Informatika
6	2021001	Ahmad Sutrisno	2021010	Kevin Jonathan	Informatika
7	2021001	Ahmad Sutrisno	2021011	Lestari Ayu	Informatika
8	2021002	Siti Nurhaliza	2021004	Oline Manuel	Informatika
9	2021002	Siti Nurhaliza	2021006	Abigail Rachel	Informatika
10	2021002	Siti Nurhaliza	2021007	Daniel Prasetyo	Informatika
11	2021002	Siti Nurhaliza	2021008	Clara Angelina	Informatika
12	2021002	Siti Nurhaliza	2021010	Kevin Jonathan	Informatika
13	2021002	Siti Nurhaliza	2021011	Lestari Ayu	Informatika
14	2021004	Oline Manuel	2021006	Abigail Rachel	Informatika
15	2021004	Oline Manuel	2021007	Daniel Prasetyo	Informatika
16	2021004	Oline Manuel	2021008	Clara Angelina	Informatika
17	2021004	Oline Manuel	2021010	Kevin Jonathan	Informatika
18	2021004	Oline Manuel	2021011	Lestari Ayu	Informatika
19	2021006	Abigail Rachel	2021007	Daniel Prasetyo	Informatika
20	2021006	Abigail Rachel	2021008	Clara Angelina	Informatika
21	2021006	Abigail Rachel	2021010	Kevin Jonathan	Informatika
22	2021006	Abigail Rachel	2021011	Lestari Ayu	Informatika
23	2021007	Daniel Prasetyo	2021008	Clara Angelina	Informatika
24	2021007	Daniel Prasetyo	2021010	Kevin Jonathan	Informatika
25	2021007	Daniel Prasetyo	2021011	Lestari Ayu	Informatika

	nim1	nama1	nim2	nama2	nama_jurusan
31	2021016	Fadli Ardi	2021019	Joko Santoso	Manajemen
32	2021016	Fadli Ardi	2021020	Kiki Ramadhani	Manajemen
33	2021016	Fadli Ardi	2021021	Lia Nuraini	Manajemen
34	2021016	Fadli Ardi	2021022	Mika Pratama	Manajemen
35	2021016	Fadli Ardi	2021023	Nina Savitri	Manajemen
36	2021016	Fadli Ardi	2021024	Oka Hidayat	Manajemen
37	2021016	Fadli Ardi	2021025	Putri Amalia	Manajemen
38	2021016	Fadli Ardi	2021026	Rian Saputra	Manajemen
39	2021017	Hani Putra	2021018	Indah Lestari	Manajemen
40	2021017	Hani Putra	2021019	Joko Santoso	Manajemen
41	2021017	Hani Putra	2021020	Kiki Ramadhani	Manajemen
42	2021017	Hani Putra	2021021	Lia Nuraini	Manajemen
43	2021017	Hani Putra	2021022	Mika Pratama	Manajemen
44	2021017	Hani Putra	2021023	Nina Savitri	Manajemen
45	2021017	Hani Putra	2021024	Oka Hidayat	Manajemen
46	2021017	Hani Putra	2021025	Putri Amalia	Manajemen
47	2021017	Hani Putra	2021026	Rian Saputra	Manajemen
48	2021018	Indah Lestari	2021019	Joko Santoso	Manajemen
49	2021018	Indah Lestari	2021020	Kiki Ramadhani	Manajemen
50	2021018	Indah Lestari	2021021	Lia Nuraini	Manajemen
51	2021018	Indah Lestari	2021022	Mika Pratama	Manajemen
52	2021018	Indah Lestari	2021023	Nina Savitri	Manajemen
53	2021018	Indah Lestari	2021024	Oka Hidayat	Manajemen
54	2021018	Indah Lestari	2021025	Putri Amalia	Manajemen
55	2021018	Indah Lestari	2021026	Rian Saputra	Manajemen
56	2021019	Joko Santoso	2021020	Kiki Ramadhani	Manajemen

84	2021003	Budi Santoso	2021005	Muhammad Ali	Sistem Informasi
85	2021003	Budi Santoso	2021009	Rizky Ramadhan	Sistem Informasi
86	2021003	Budi Santoso	2021012	Andi Wijaya	Sistem Informasi
87	2021003	Budi Santoso	2021013	Nabila Putri	Sistem Informasi
88	2021003	Budi Santoso	2021014	Jonathan Surya	Sistem Informasi
89	2021003	Budi Santoso	2021015	Maya Fitri	Sistem Informasi
90	2021005	Muhammad Ali	2021009	Rizky Ramadhan	Sistem Informasi
91	2021005	Muhammad Ali	2021012	Andi Wijaya	Sistem Informasi
92	2021005	Muhammad Ali	2021013	Nabila Putri	Sistem Informasi
93	2021005	Muhammad Ali	2021014	Jonathan Surya	Sistem Informasi
94	2021005	Muhammad Ali	2021015	Maya Fitri	Sistem Informasi
95	2021009	Rizky Ramadhan	2021012	Andi Wijaya	Sistem Informasi
96	2021009	Rizky Ramadhan	2021013	Nabila Putri	Sistem Informasi
97	2021009	Rizky Ramadhan	2021014	Jonathan Surya	Sistem Informasi
98	2021009	Rizky Ramadhan	2021015	Maya Fitri	Sistem Informasi
99	2021012	Andi Wijaya	2021013	Nabila Putri	Sistem Informasi
100	2021012	Andi Wijaya	2021014	Jonathan Surya	Sistem Informasi
101	2021012	Andi Wijaya	2021015	Maya Fitri	Sistem Informasi
102	2021013	Nabila Putri	2021014	Jonathan Surya	Sistem Informasi
103	2021013	Nabila Putri	2021015	Maya Fitri	Sistem Informasi
104	2021014	Jonathan Surya	2021015	Maya Fitri	Sistem Informasi

Kesimpulan Pembelajaran

Selama mengerjakan tugas pertama dari minggu pertama mata kuliah Advance Database ini, saya mengingat kembali tentang bagaimana cara mengambil, memfilter, dan mengurutkan data menggunakan query SQL dasar seperti SELECT, WHERE, ORDER BY, dan LIMIT. Selain itu, saya memahami penggunaan UPDATE untuk memodifikasi data dan COUNT serta GROUP BY untuk menghitung data berdasarkan kategori tertentu.

Dalam latihan JOIN, saya belajar berbagai jenis penggabungan tabel (INNER JOIN, LEFT JOIN, FULL OUTER JOIN) serta self-join untuk menemukan relasi dalam tabel yang sama. Penggunaan alias seperti j.nama_jurusan, m1-m2, dan nim1-nim2 membantu saya untuk membuat query yang lebih terlihat rapih dan jelas. Dengan adanya tugas ini saya jadi dapat mengingat,dan mempelajari hal yang blm masuk ke kepala saya sejak semester 2 tentang system database, kemudian juga menekankan pentingnya struktur tabel yang baik, relasi antar tabel, dan pengambilan data yang efisien.