Kanaya Dea Thalita Akhmad
121450001
RB

Materialized views & Transactions

Tools : Java Programming, DBMS Seperti Mysql, Postgress, MariaDB dll.

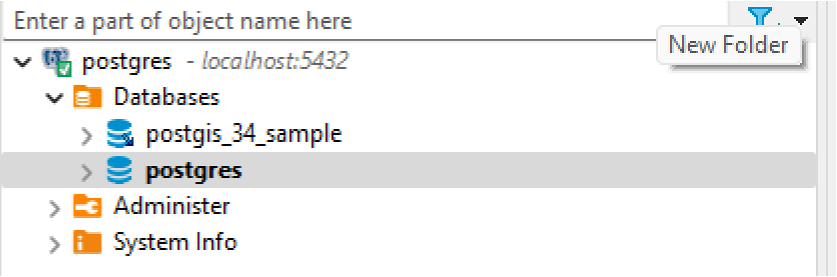
Tujuan: Mampu membuat material view dan transaction

Deskripsi

1. Pastikan DBMS sudah terinstall dan sedang dalam keadaan run
2. Download Query DDL. Download DDL
3. Buat Schema Database. Contoh DBMS1
   Eksekusi SQL pada bagian (2)
4. Download Code dalam bahasa pemrograman Java. Download Code
5. Compile dan jalan code tersebut pada komputer anda.
   $javactableGen.java$java tableGen
6. Buat contoh Materialized view dan transactions (masing-masing 5)
   Thank, Selamat mengerjakan 😊

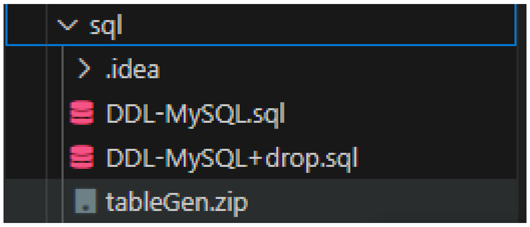
Jawab :

1. Pastikan DBMS sudah terinstall dan sedang dalam keadaan run



tanda centang hijau menandakan sudah terkoenksinya antara postgre dengan local server yang saya punya

2. Download Query DDL. Download DDL



sql
.idea
DDL-MySQL.sql
DDL-MySQL+drop.sql
tableGen.zip

3. Buat Schema Database. Contoh DBMS1

dan Eksekusi SQL pada bagian (2)

```sql
create table classroom
        (building               varchar(15),
         room_number            varchar(7),
         capacity               numeric(4,0),
         primary key (building, room_number)
        );
create table department
        (dept_name              varchar(20),
         building               varchar(15),
         budget                 numeric(12,2) check (budget > 0),
         primary key (dept_name)
        );
create table course
        (course_id              varchar(8),
         title                  varchar(50),
         dept_name              varchar(20),
         credits                numeric(2,0) check (credits > 0),
         primary key (course_id),
         foreign key (dept_name) references department(dept_name)
                on delete set null
        );
create table instructor
        (ID                     varchar(5),
         name                   varchar(20) not null,
         dept_name              varchar(20),
         salary                 numeric(8,2) check (salary > 29000),
         primary key (ID),
         foreign key (dept_name) references department(dept_name)
                on delete set null
        );
create table section
        (course_id              varchar(8),
         sec_id                 varchar(8),
         semester               varchar(6)
                check (semester in ('Fall', 'Winter', 'Spring', 'Summer')),
         year                   numeric(4,0) check (year > 1701 and year < 2100),
         building               varchar(15),
         room_number            varchar(7),
         time_slot_id           varchar(4),
         primary key (course_id, sec_id, semester, year),
         foreign key (course_id) references course(course_id)
                on delete cascade,
         foreign key (building, room_number) references classroom(building, room_number)
                on delete set null
        );
create table teaches
        (ID                     varchar(5),
         course_id              varchar(8),
         sec_id                 varchar(8),
         semester               varchar(6),
         year                   numeric(4,0),
         primary key (ID, course_id, sec_id, semester, year),
         foreign key (course_id,sec_id, semester, year) references section(course_id,sec_id, ser
                on delete cascade,
         foreign key (ID) references instructor(ID)
                on delete cascade
        );
create table student
        (ID                     varchar(5),
         name                   varchar(20) not null,
         dept_name              varchar(20),
         tot_cred               numeric(3,0) check (tot_cred >= 0),
         primary key (ID),
         foreign key (dept_name) references department(dept_name)
                on delete set null
        );
create table takes
        (ID                     varchar(5),
         course_id              varchar(8),
         sec_id                 varchar(8),
         semester               varchar(6),
         year                   numeric(4,0),
         grade                  varchar(2),
         primary key (ID, course_id, sec_id, semester, year),
         foreign key (course_id,sec_id, semester, year) references section(course_id,sec_id, ser
                on delete cascade,
         foreign key (ID) references student(ID)
                on delete cascade
        );
create table advisor
        (s_ID                   varchar(5),
         i_ID                   varchar(5),
         primary key (s_ID),
         foreign key (i_ID) references instructor (ID)
                on delete set null,
         foreign key (s_ID) references student (ID)
                on delete cascade
        );
create table prereq
        (course_id              varchar(8),
         prereq_id              varchar(8),
         primary key (course_id, prereq_id),
         foreign key (course_id) references course(course_id)
                on delete cascade,
         foreign key (prereq_id) references course(course_id)
        )
```

Menampilkan seluruh tabel yang telah ditambahkan melalui query diatas.

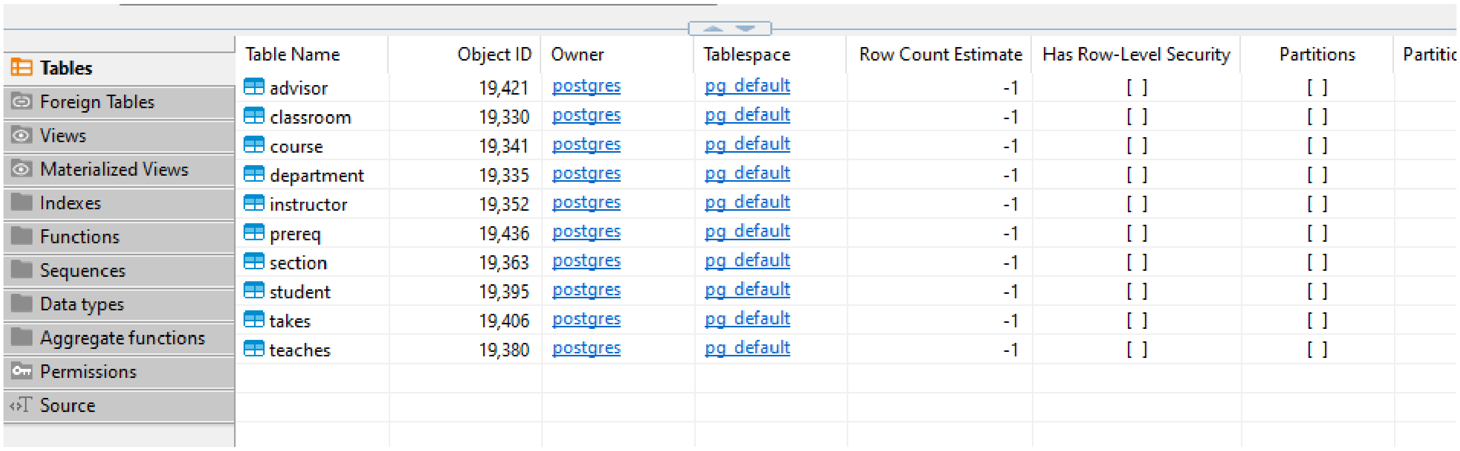
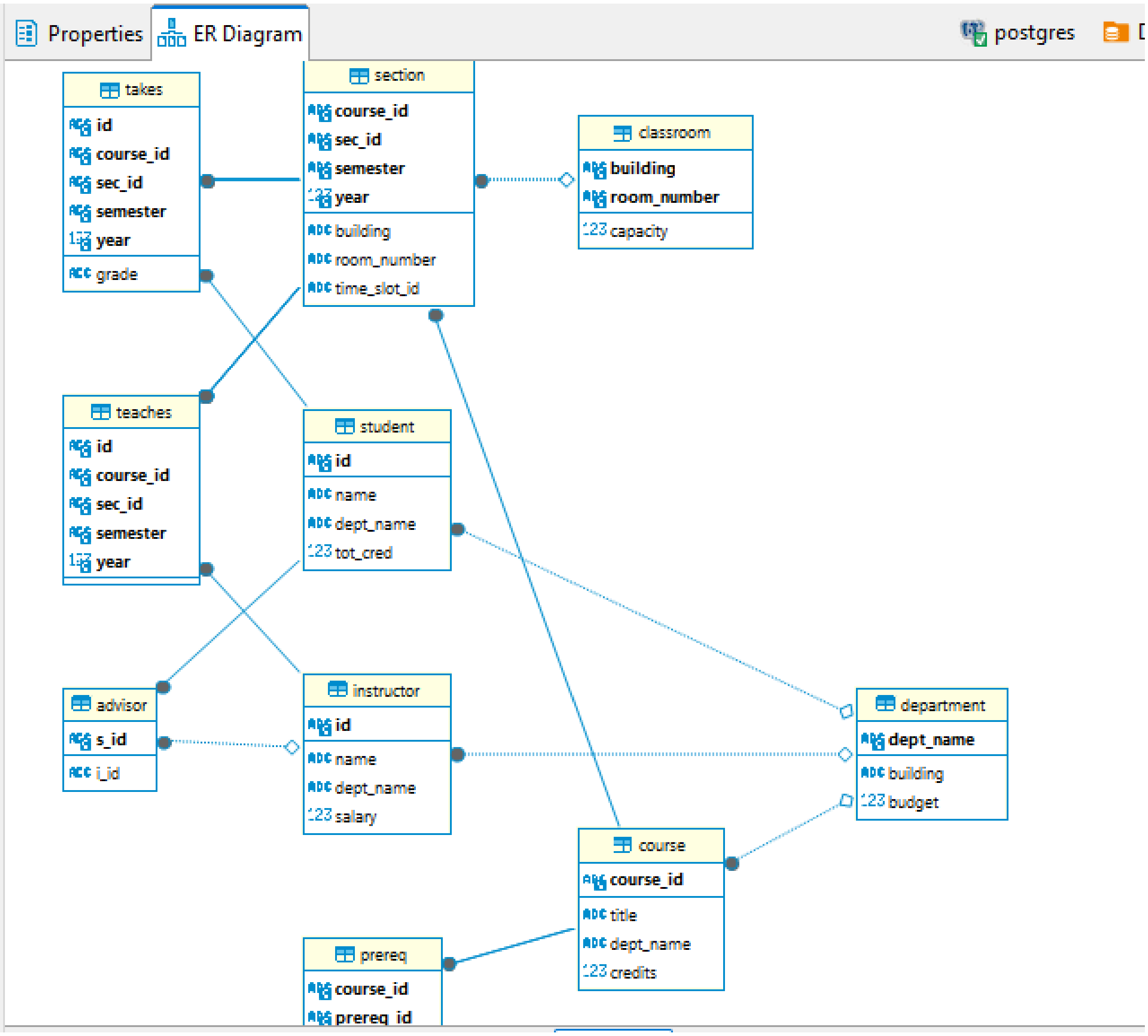


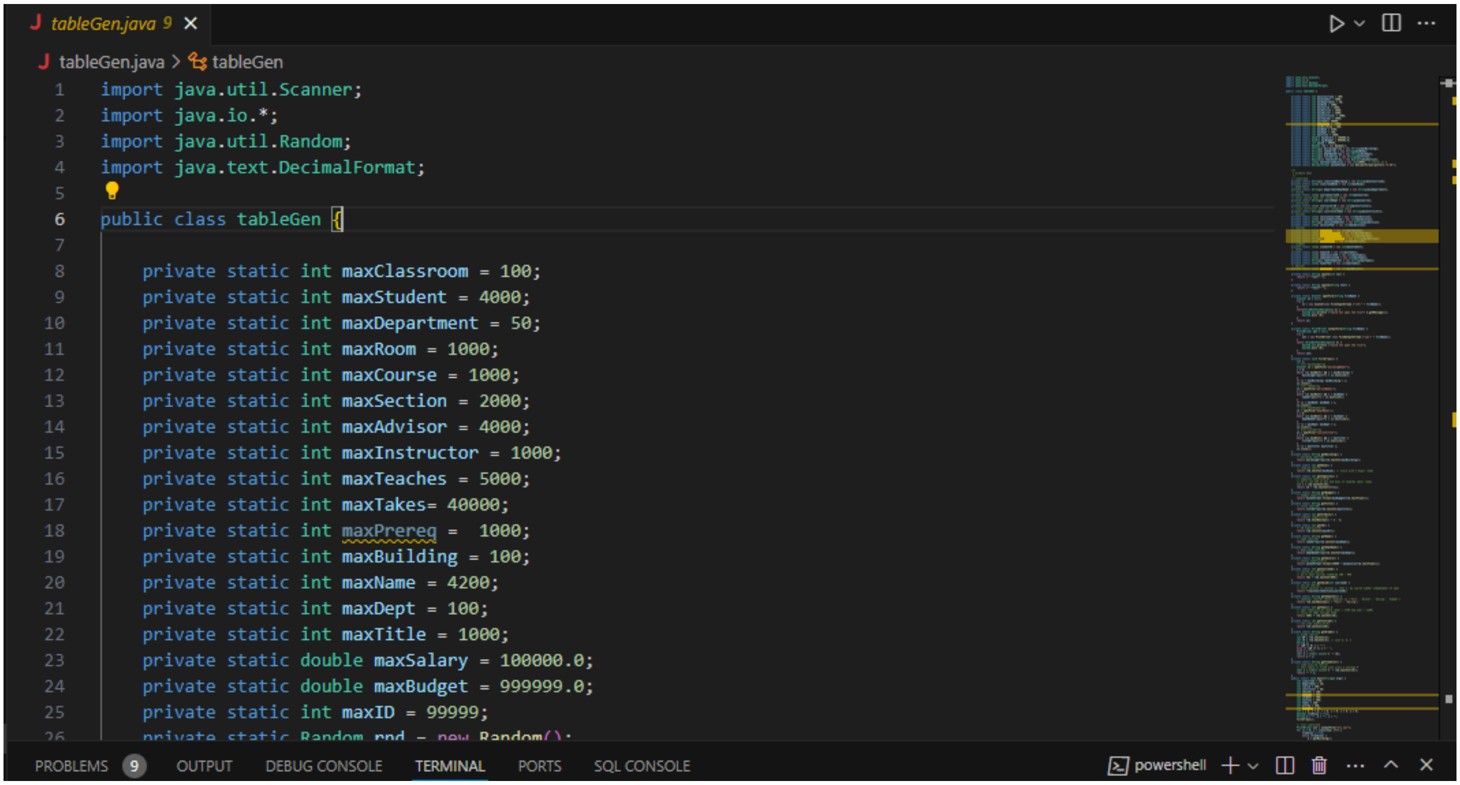
Table Name	Object ID	Owner	Tablespace	Row Count Estimate	Has Row-Level Security	Partitions	Partitic
advisor	19,421	postgres	pg_default	-1	[ ]	[ ]	
classroom	19,330	postgres	pg_default	-1	[ ]	[ ]	
course	19,341	postgres	pg_default	-1	[ ]	[ ]	
department	19,335	postgres	pg_default	-1	[ ]	[ ]	
instructor	19,352	postgres	pg_default	-1	[ ]	[ ]	
prereq	19,436	postgres	pg_default	-1	[ ]	[ ]	
section	19,363	postgres	pg_default	-1	[ ]	[ ]	
student	19,395	postgres	pg_default	-1	[ ]	[ ]	
takes	19,406	postgres	pg_default	-1	[ ]	[ ]	
teaches	19,380	postgres	pg_default	-1	[ ]	[ ]	

tampilan database relationalnya yang terhubung satu sama lain.



4. Download Code dalam bahasa pemrograman Java. Download Code

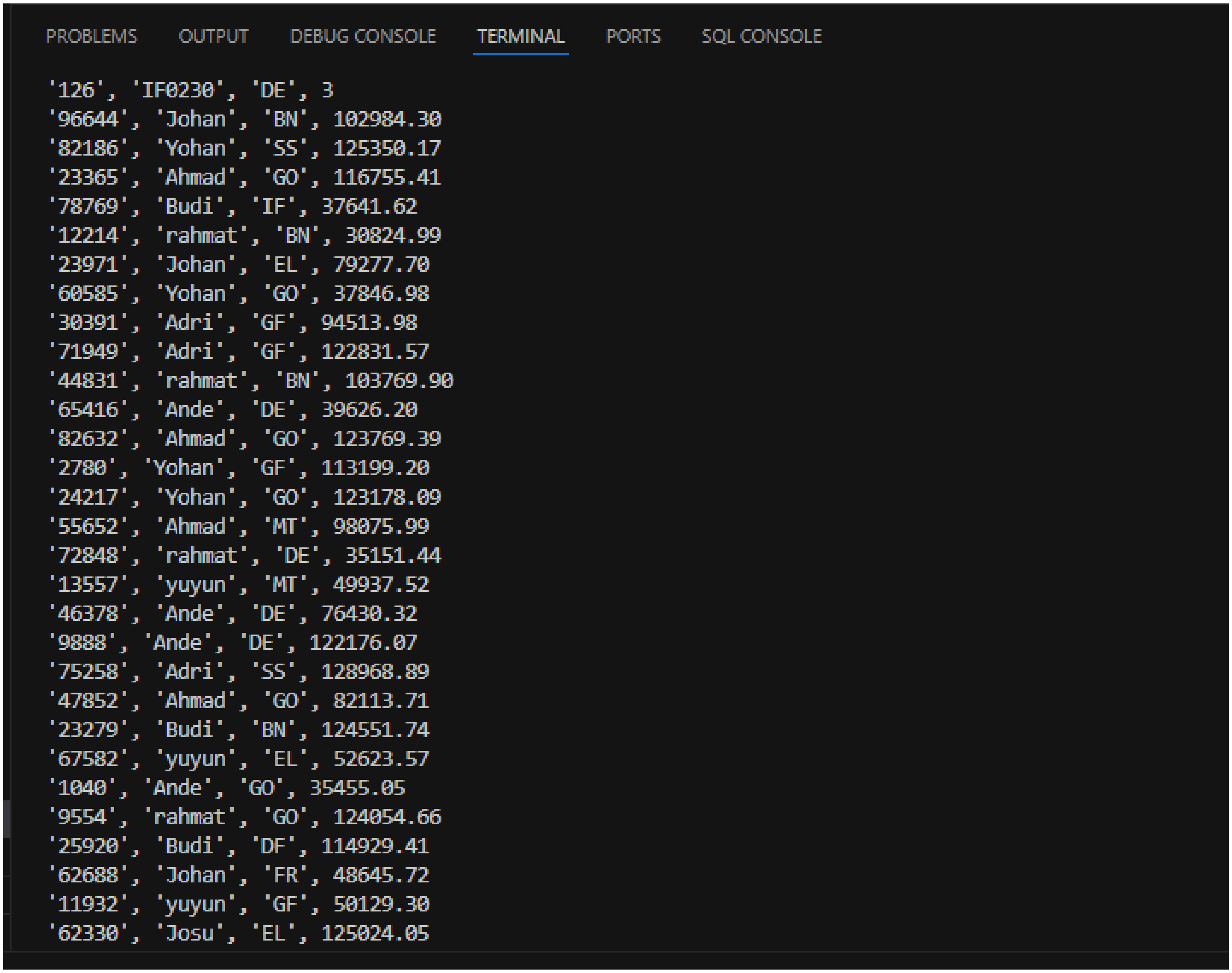
code java :



```java
import java.util.Scanner;
import java.io.*;
import java.util.Random;
import java.text.DecimalFormat;

public class tableGen {

    private static int maxClassroom = 100;
    private static int maxStudent = 4000;
    private static int maxDepartment = 50;
    private static int maxRoom = 1000;
    private static int maxCourse = 1000;
    private static int maxSection = 2000;
    private static int maxAdvisor = 4000;
    private static int maxInstructor = 1000;
    private static int maxTeaches = 5000;
    private static int maxTakes= 40000;
    private static int maxPrereq =  1000;
    private static int maxBuilding = 100;
    private static int maxName = 4200;
    private static int maxDept = 100;
    private static int maxTitle = 1000;
    private static double maxSalary = 100000.0;
    private static double maxBudget = 999999.0;
    private static int maxID = 99999;
    private static Random rnd = new Random();
```

5. Compile dan jalan code tersebut pada komputer anda.

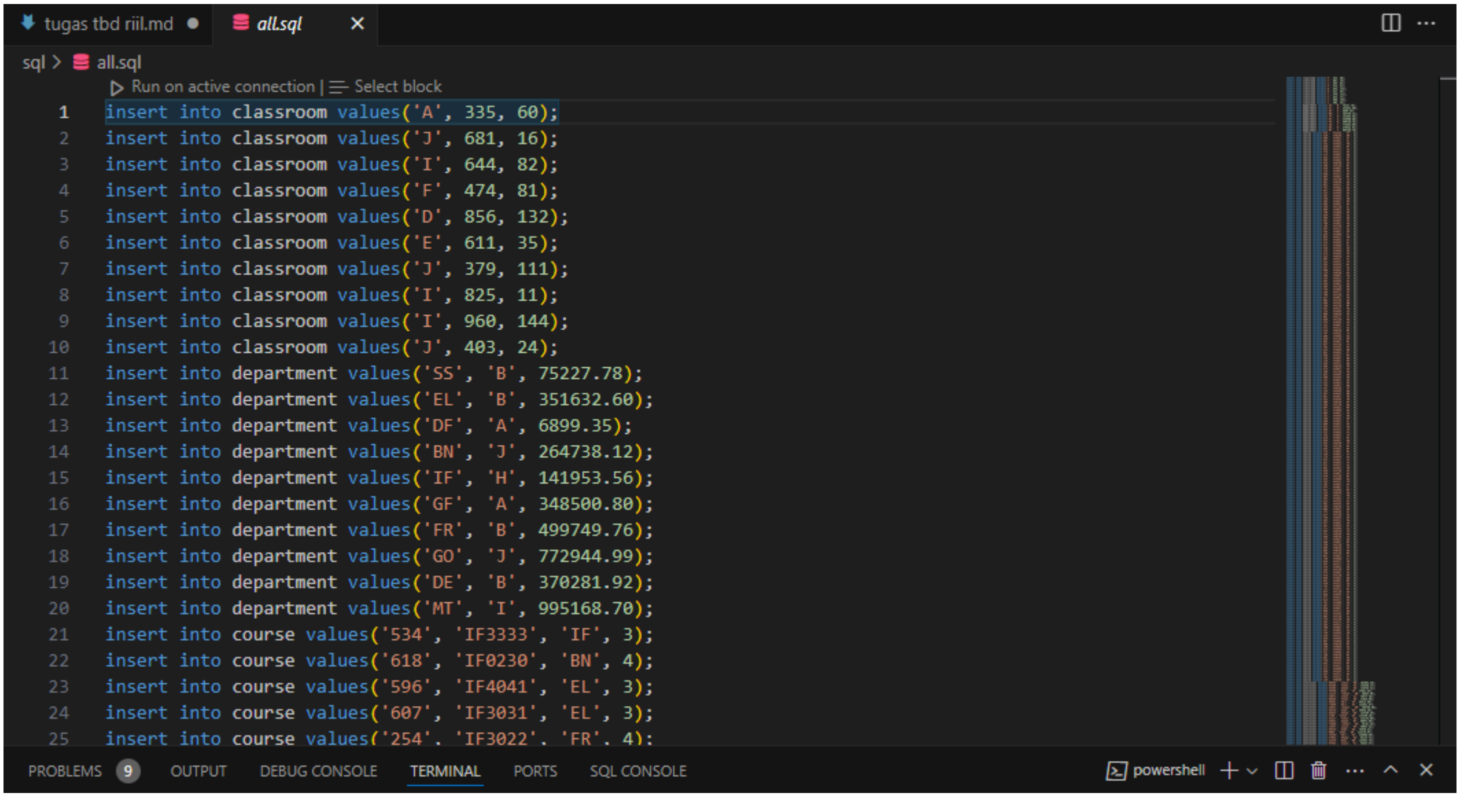
maka menghasilkan output pada terminal seperti berikut



```
'126', 'IF0230', 'DE', 3
'96644', 'Johan', 'BN', 102984.30
'82186', 'Yohan', 'SS', 125350.17
'23365', 'Ahmad', 'GO', 116755.41
'78769', 'Budi', 'IF', 37641.62
'12214', 'rahmat', 'BN', 30824.99
'23971', 'Johan', 'EL', 79277.70
'60585', 'Yohan', 'GO', 37846.98
'30391', 'Adri', 'GF', 94513.98
'71949', 'Adri', 'GF', 122831.57
'44831', 'rahmat', 'BN', 103769.90
'65416', 'Ande', 'DE', 39626.20
'82632', 'Ahmad', 'GO', 123769.39
'2780', 'Yohan', 'GF', 113199.20
'24217', 'Yohan', 'GO', 123178.09
'55652', 'Ahmad', 'MT', 98075.99
'72848', 'rahmat', 'DE', 35151.44
'13557', 'yuyun', 'MT', 49937.52
'46378', 'Ande', 'DE', 76430.32
'9888', 'Ande', 'DE', 122176.07
'75258', 'Adri', 'SS', 128968.89
'47852', 'Ahmad', 'GO', 82113.71
'23279', 'Budi', 'BN', 124551.74
'67582', 'yuyun', 'EL', 52623.57
'1040', 'Ande', 'GO', 35455.05
'9554', 'rahmat', 'GO', 124054.66
'25920', 'Budi', 'DF', 114929.41
'62688', 'Johan', 'FR', 48645.72
'11932', 'yuyun', 'GF', 50129.30
'62330', 'Josu', 'EL', 125024.05
```

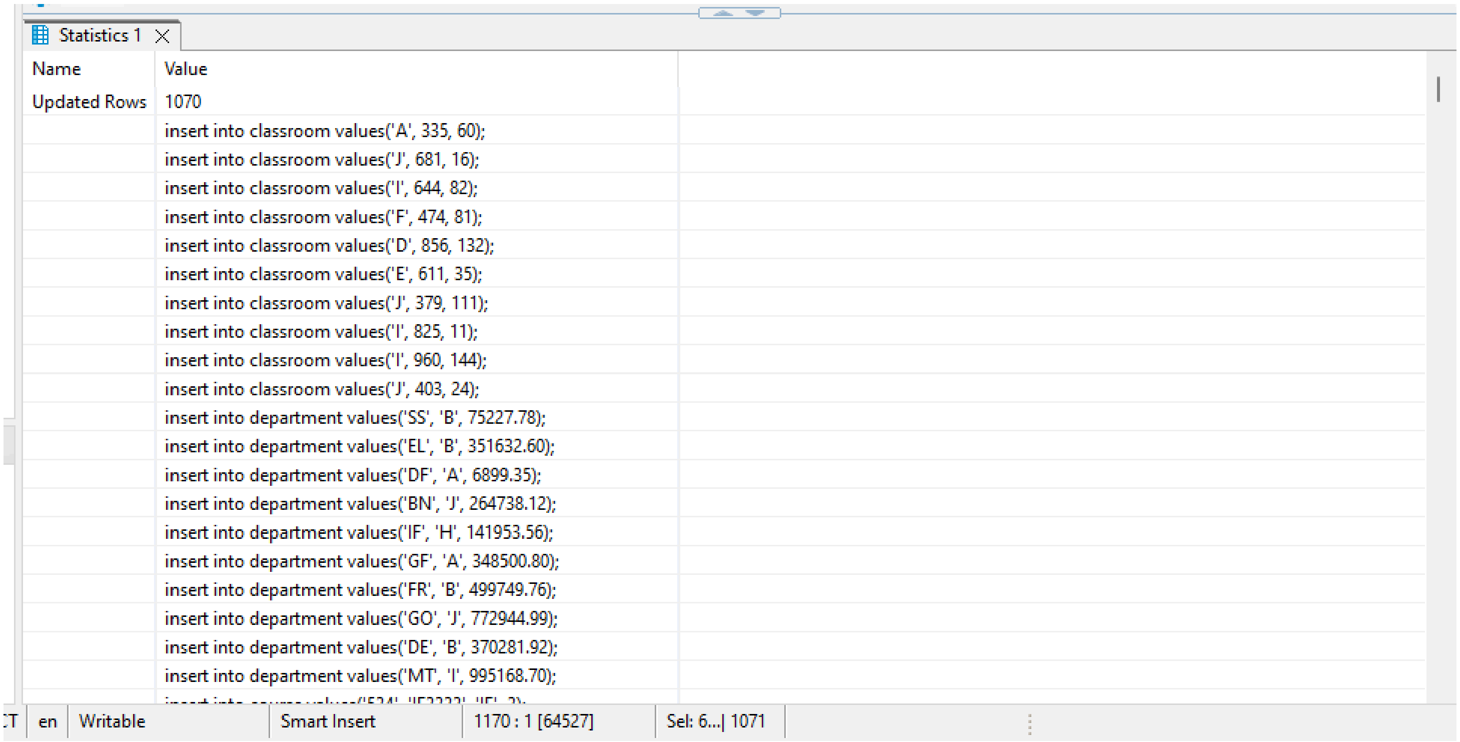
File tableGen.class adalah file bytecode yang dihasilkan setelah kompilasi dari kode sumber Java (tableGen.java). File ini tidak dapat dibaca dan diedit secara langsung menggunakan editor teks karena berisi bytecode yang dirancang untuk dijalankan oleh Java Virtual Machine (JVM).

selain itu akan dihasilkan output sql yang menampilkan



```sql
insert into classroom values('A', 335, 60);
insert into classroom values('J', 681, 16);
insert into classroom values('I', 644, 82);
insert into classroom values('F', 474, 81);
insert into classroom values('D', 856, 132);
insert into classroom values('E', 611, 35);
insert into classroom values('J', 379, 111);
insert into classroom values('I', 825, 11);
insert into classroom values('I', 960, 144);
insert into classroom values('J', 403, 24);
insert into department values('SS', 'B', 75227.78);
insert into department values('EL', 'B', 351632.60);
insert into department values('DF', 'A', 6899.35);
insert into department values('BN', 'J', 264738.12);
insert into department values('IF', 'H', 141953.56);
insert into department values('GF', 'A', 348500.80);
insert into department values('FR', 'B', 499749.76);
insert into department values('GO', 'J', 772944.99);
insert into department values('DE', 'B', 370281.92);
insert into department values('MT', 'I', 995168.70);
insert into course values('534', 'IF3333', 'IF', 3);
insert into course values('618', 'IF0230', 'BN', 4);
insert into course values('596', 'IF4041', 'EL', 3);
insert into course values('607', 'IF3031', 'EL', 3);
insert into course values('254', 'IF3022', 'FR', 4);
```

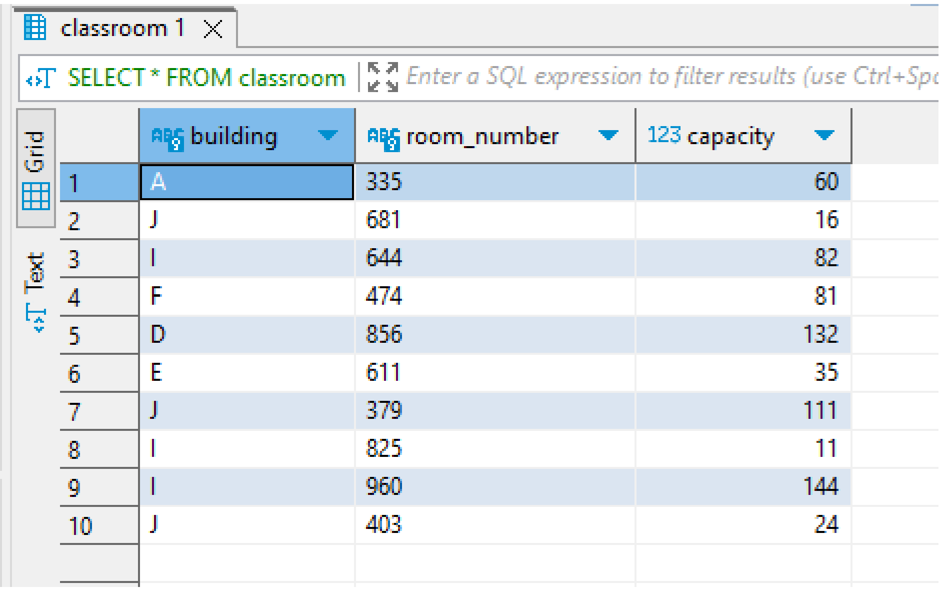
versi yang ada di server lokal dengan dbeaver (sama saja)



Name	Value
Updated Rows	1070
	insert into classroom values('A', 335, 60);
	insert into classroom values('J', 681, 16);
	insert into classroom values('I', 644, 82);
	insert into classroom values('F', 474, 81);
	insert into classroom values('D', 856, 132);
	insert into classroom values('E', 611, 35);
	insert into classroom values('J', 379, 111);
	insert into classroom values('I', 825, 11);
	insert into classroom values('I', 960, 144);
	insert into classroom values('J', 403, 24);
	insert into department values('SS', 'B', 75227.78);
	insert into department values('EL', 'B', 351632.60);
	insert into department values('DF', 'A', 6899.35);
	insert into department values('BN', 'J', 264738.12);
	insert into department values('IF', 'H', 141953.56);
	insert into department values('GF', 'A', 348500.80);
	insert into department values('FR', 'B', 499749.76);
	insert into department values('GO', 'J', 772944.99);
	insert into department values('DE', 'B', 370281.92);
	insert into department values('MT', 'I', 995168.70);
	insert into course values('534', 'IF3333', 'IF', 3);

```
T | en | Writable | Smart Insert | 1170 : 1 [64527] | Sel: 6...| 1071
```

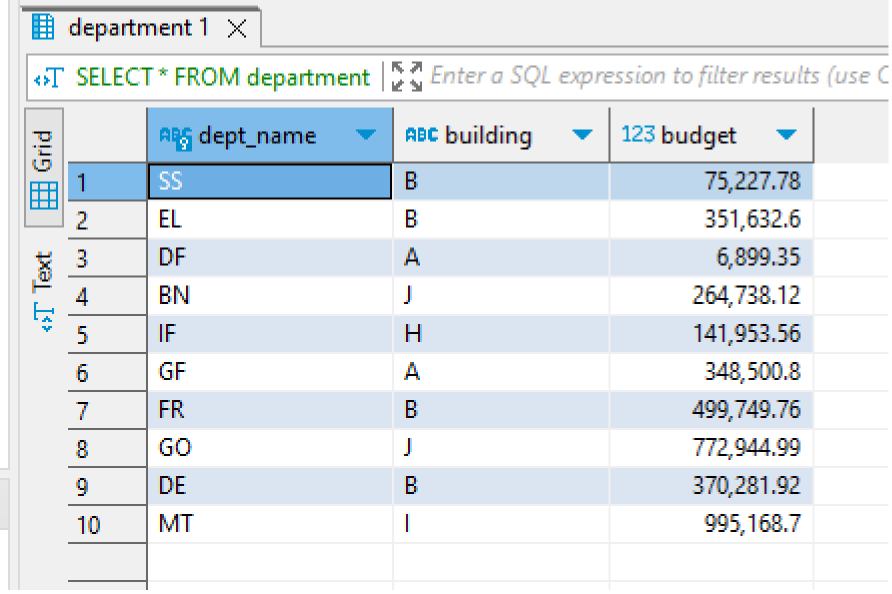
1. tampilan tabel classroom setelah diinput



SELECT * FROM classroom

	building	room_number	capacity
1	A	335	60
2	J	681	16
3	I	644	82
4	F	474	81
5	D	856	132
6	E	611	35
7	J	379	111
8	I	825	11
9	I	960	144
10	J	403	24

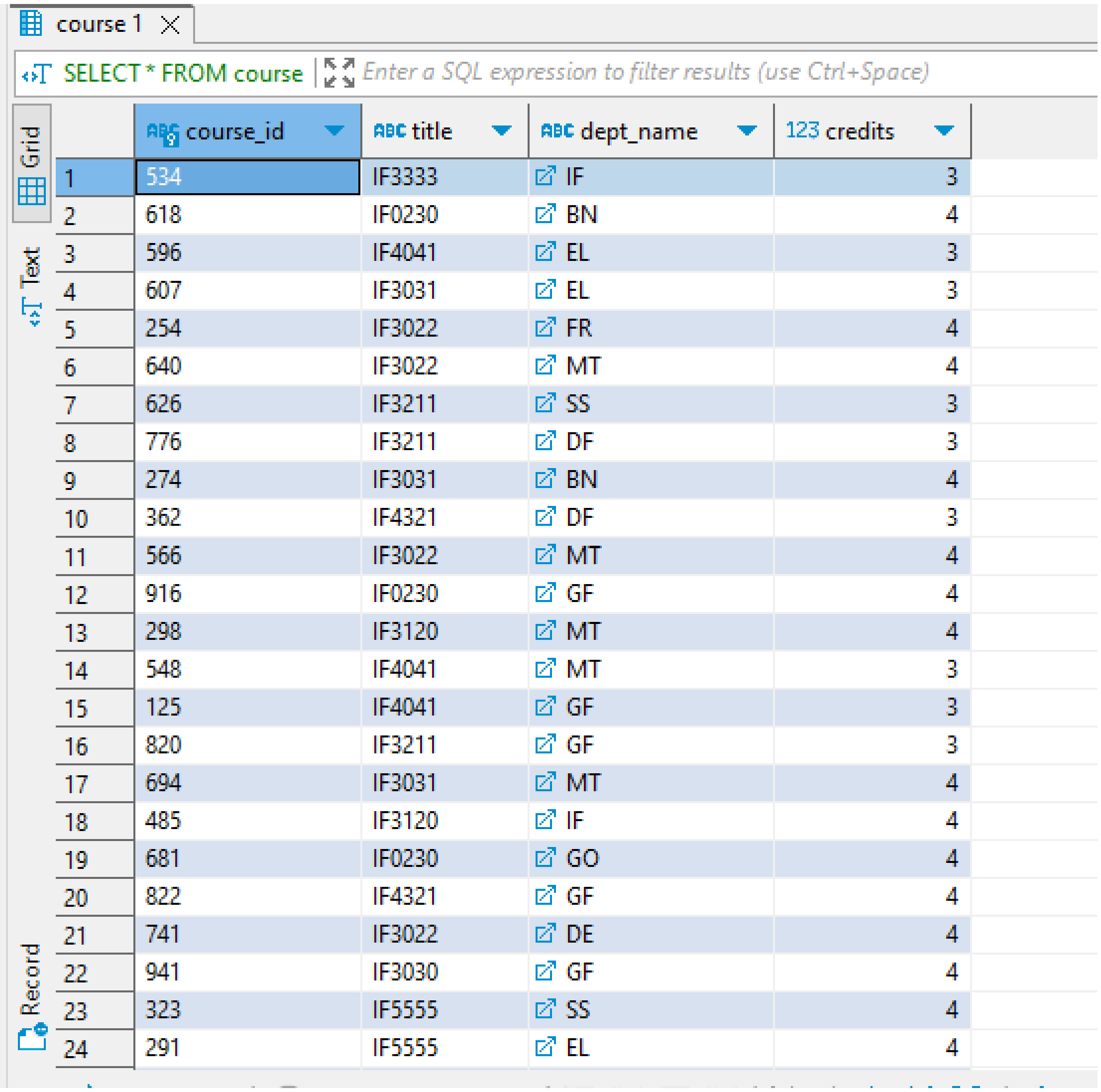
2. tampilan tabel departemen setelah diinput



SELECT * FROM department

	dept_name	building	budget
1	SS	B	75,227.78
2	EL	B	351,632.6
3	DF	A	6,899.35
4	BN	J	264,738.12
5	IF	H	141,953.56
6	GF	A	348,500.8
7	FR	B	499,749.76
8	GO	J	772,944.99
9	DE	B	370,281.92
10	MT	I	995,168.7

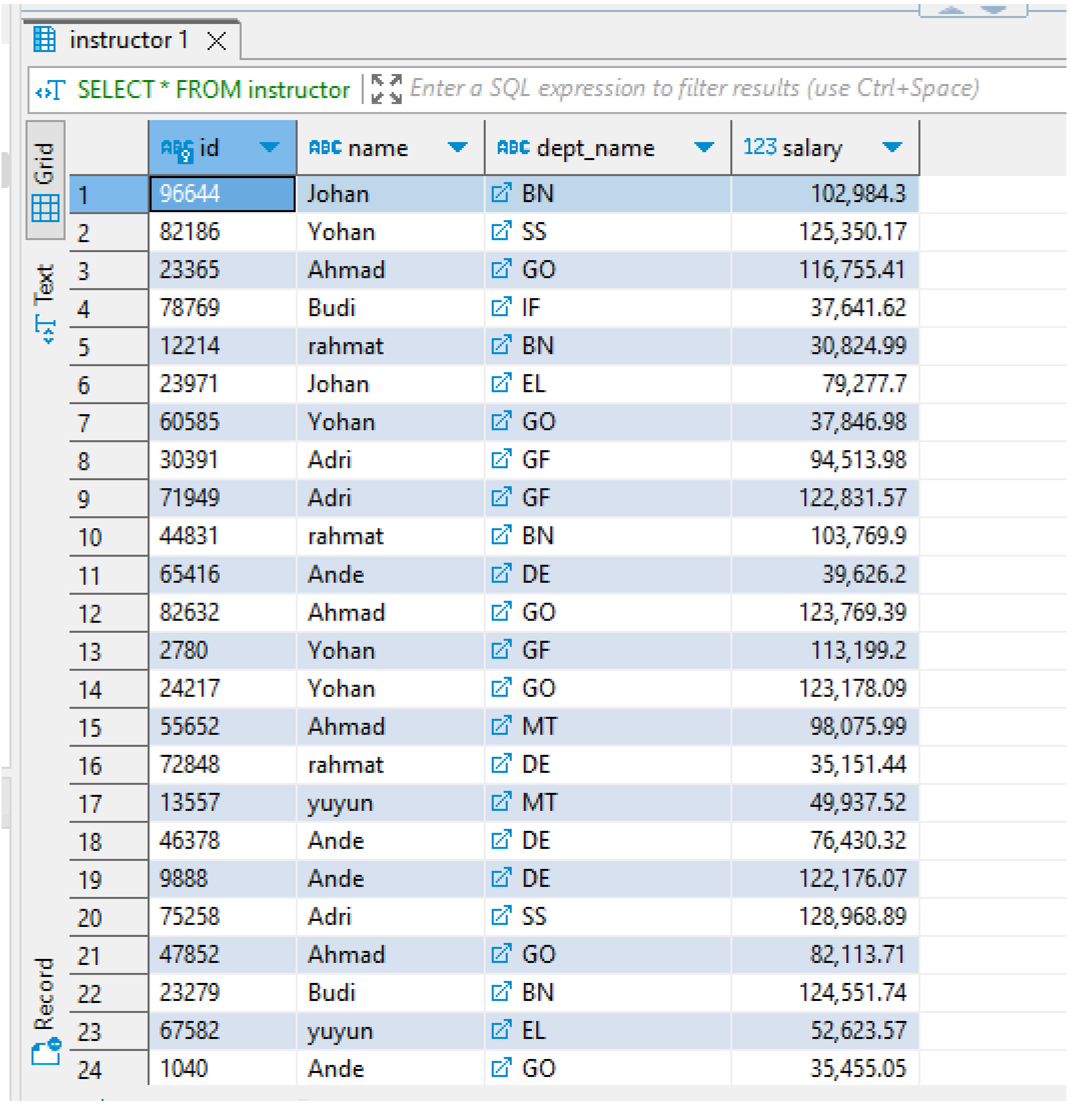
3. tampilan tabel course setelah diinput



SELECT * FROM course

	course_id	title	dept_name	credits
1	534	IF3333	IF	3
2	618	IF0230	BN	4
3	596	IF4041	EL	3
4	607	IF3031	EL	3
5	254	IF3022	FR	4
6	640	IF3022	MT	4
7	626	IF3211	SS	3
8	776	IF3211	DF	3
9	274	IF3031	BN	4
10	362	IF4321	DF	3
11	566	IF3022	MT	4
12	916	IF0230	GF	4
13	298	IF3120	MT	4
14	548	IF4041	MT	3
15	125	IF4041	GF	3
16	820	IF3211	GF	3
17	694	IF3031	MT	4
18	485	IF3120	IF	4
19	681	IF0230	GO	4
20	822	IF4321	GF	4
21	741	IF3022	DE	4
22	941	IF3030	GF	4
23	323	IF5555	SS	4
24	291	IF5555	EL	4

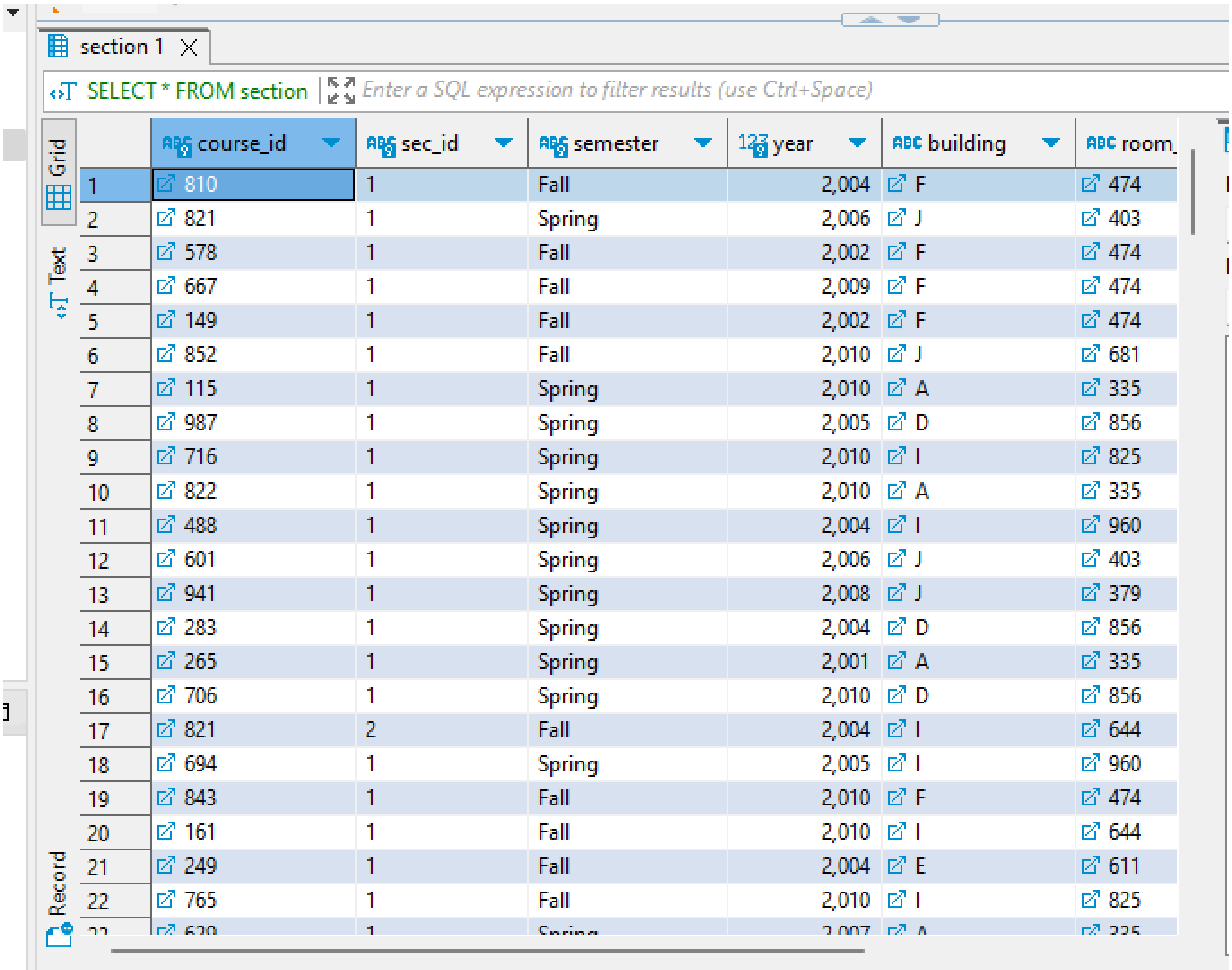
4. tampilan tabel instructor setelah diinput



SELECT * FROM instructor

	id	name	dept_name	salary
1	96644	Johan	BN	102,984.3
2	82186	Yohan	SS	125,350.17
3	23365	Ahmad	GO	116,755.41
4	78769	Budi	IF	37,641.62
5	12214	rahmat	BN	30,824.99
6	23971	Johan	EL	79,277.7
7	60585	Yohan	GO	37,846.98
8	30391	Adri	GF	94,513.98
9	71949	Adri	GF	122,831.57
10	44831	rahmat	BN	103,769.9
11	65416	Ande	DE	39,626.2
12	82632	Ahmad	GO	123,769.39
13	2780	Yohan	GF	113,199.2
14	24217	Yohan	GO	123,178.09
15	55652	Ahmad	MT	98,075.99
16	72848	rahmat	DE	35,151.44
17	13557	yuyun	MT	49,937.52
18	46378	Ande	DE	76,430.32
19	9888	Ande	DE	122,176.07
20	75258	Adri	SS	128,968.89
21	47852	Ahmad	GO	82,113.71
22	23279	Budi	BN	124,551.74
23	67582	yuyun	EL	52,623.57
24	1040	Ande	GO	35,455.05

5. tampilan tabel section setelah diinput

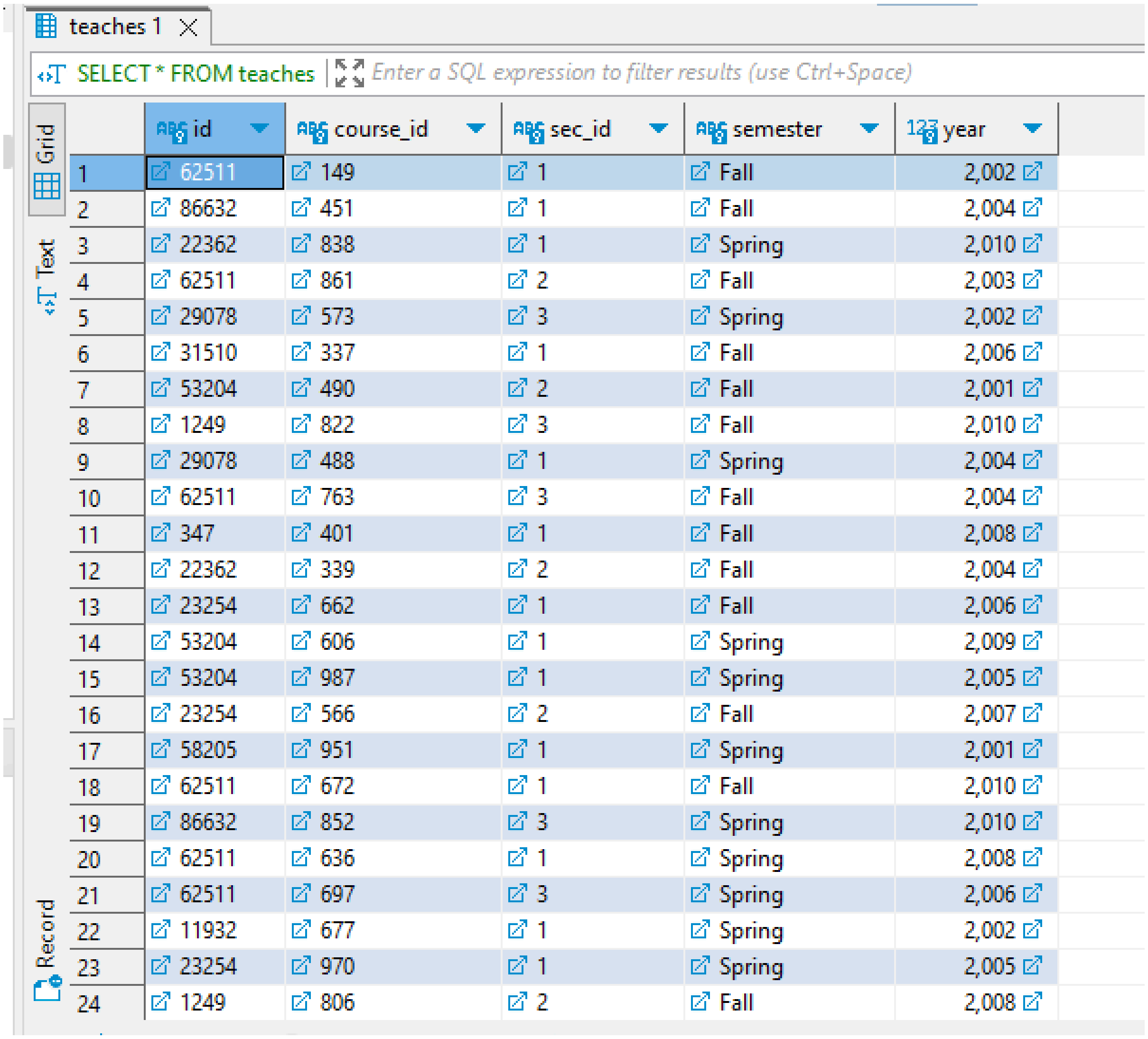


SELECT * FROM section

Enter a SQL expression to filter results (use Ctrl+Space)

	course_id	sec_id	semester	year	building	room_
1	810	1	Fall	2,004	F	474
2	821	1	Spring	2,006	J	403
3	578	1	Fall	2,002	F	474
4	667	1	Fall	2,009	F	474
5	149	1	Fall	2,002	F	474
6	852	1	Fall	2,010	J	681
7	115	1	Spring	2,010	A	335
8	987	1	Spring	2,005	D	856
9	716	1	Spring	2,010	I	825
10	822	1	Spring	2,010	A	335
11	488	1	Spring	2,004	I	960
12	601	1	Spring	2,006	J	403
13	941	1	Spring	2,008	J	379
14	283	1	Spring	2,004	D	856
15	265	1	Spring	2,001	A	335
16	706	1	Spring	2,010	D	856
17	821	2	Fall	2,004	I	644
18	694	1	Spring	2,005	I	960
19	843	1	Fall	2,010	F	474
20	161	1	Fall	2,010	I	644
21	249	1	Fall	2,004	E	611
22	765	1	Fall	2,010	I	825
23	620	1	Spring	2,007	A	335

6. tampilan tabel teaches setelah diinput

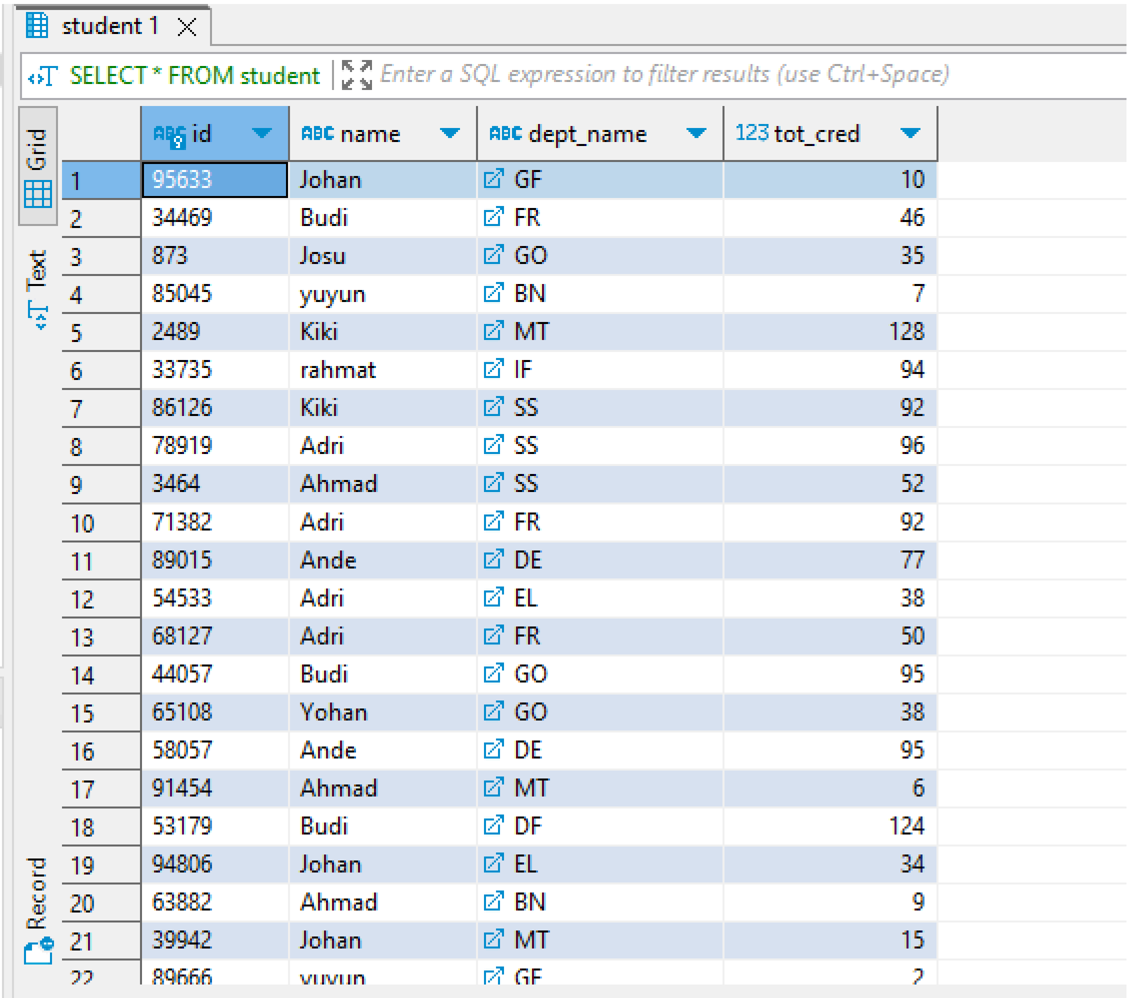


SELECT * FROM teaches

Enter a SQL expression to filter results (use Ctrl+Space)

	id	course_id	sec_id	semester	year
1	62511	149	1	Fall	2,002
2	86632	451	1	Fall	2,004
3	22362	838	1	Spring	2,010
4	62511	861	2	Fall	2,003
5	29078	573	3	Spring	2,002
6	31510	337	1	Fall	2,006
7	53204	490	2	Fall	2,001
8	1249	822	3	Fall	2,010
9	29078	488	1	Spring	2,004
10	62511	763	3	Fall	2,004
11	347	401	1	Fall	2,008
12	22362	339	2	Fall	2,004
13	23254	662	1	Fall	2,006
14	53204	606	1	Spring	2,009
15	53204	987	1	Spring	2,005
16	23254	566	2	Fall	2,007
17	58205	951	1	Spring	2,001
18	62511	672	1	Fall	2,010
19	86632	852	3	Spring	2,010
20	62511	636	1	Spring	2,008
21	62511	697	3	Spring	2,006
22	11932	677	1	Spring	2,002
23	23254	970	1	Spring	2,005
24	1249	806	2	Fall	2,008

7. tampilan tabel student setelah diinput

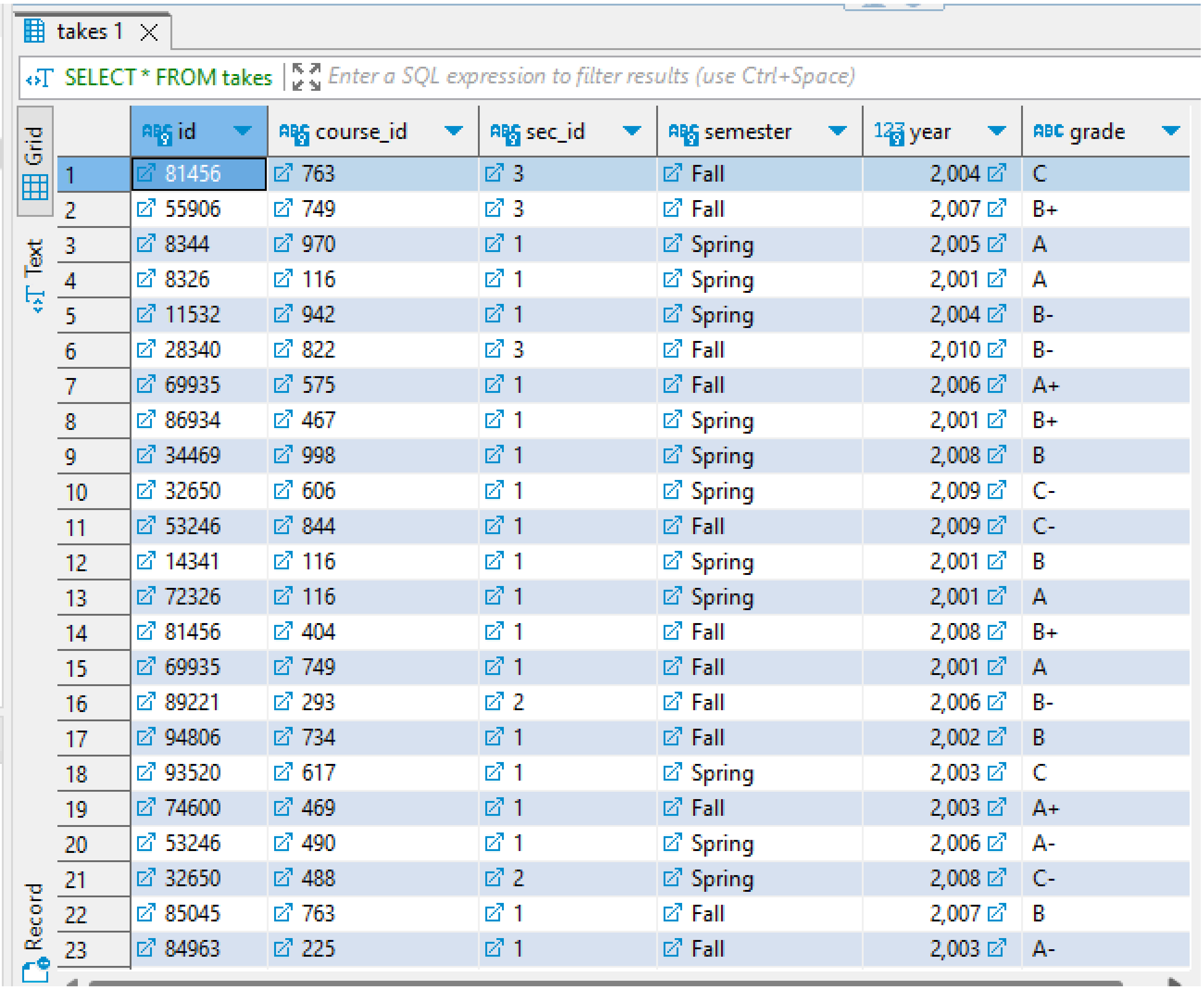


SELECT * FROM student

Enter a SQL expression to filter results (use Ctrl+Space)

	id	name	dept_name	tot_cred
1	95633	Johan	GF	10
2	34469	Budi	FR	46
3	873	Josu	GO	35
4	85045	yuyun	BN	7
5	2489	Kiki	MT	128
6	33735	rahmat	IF	94
7	86126	Kiki	SS	92
8	78919	Adri	SS	96
9	3464	Ahmad	SS	52
10	71382	Adri	FR	92
11	89015	Ande	DE	77
12	54533	Adri	EL	38
13	68127	Adri	FR	50
14	44057	Budi	GO	95
15	65108	Yohan	GO	38
16	58057	Ande	DE	95
17	91454	Ahmad	MT	6
18	53179	Budi	DF	124
19	94806	Johan	EL	34
20	63882	Ahmad	BN	9
21	39942	Johan	MT	15
22	89666	yuyun	GF	2

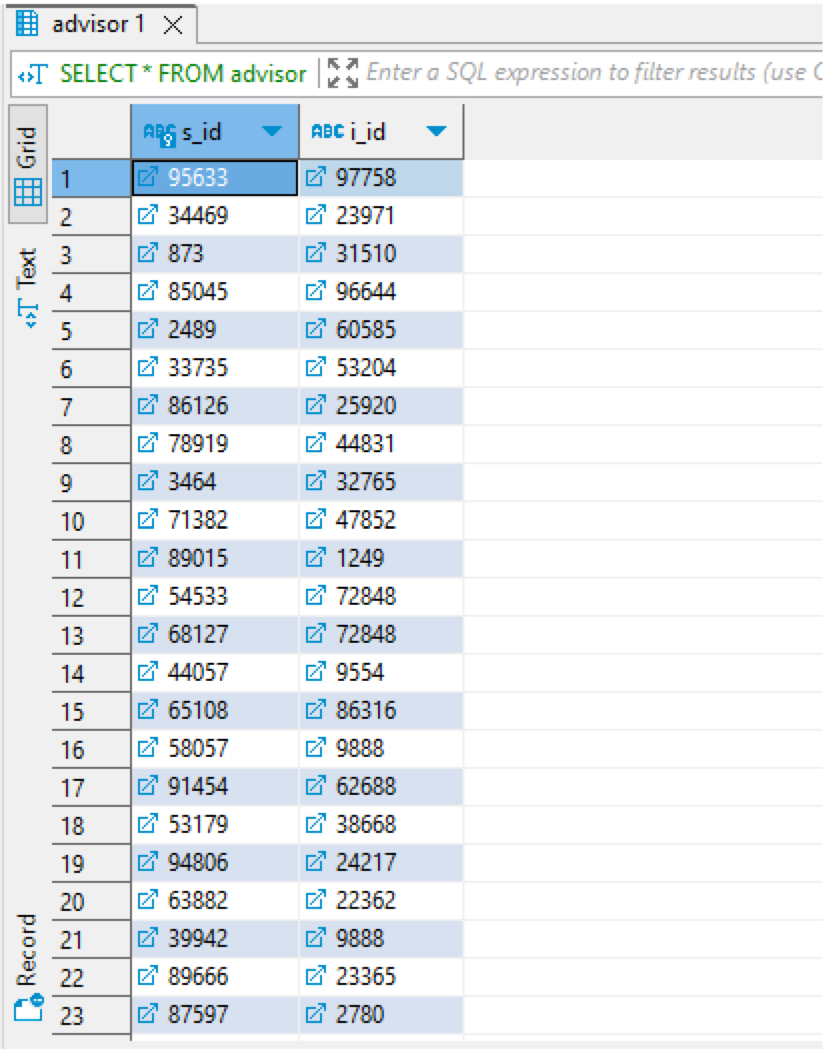
8. tampilan tabel takes setelah diinput



```
SELECT * FROM takes
```

	id	course_id	sec_id	semester	year	grade
1	81456	763	3	Fall	2,004	C
2	55906	749	3	Fall	2,007	B+
3	8344	970	1	Spring	2,005	A
4	8326	116	1	Spring	2,001	A
5	11532	942	1	Spring	2,004	B-
6	28340	822	3	Fall	2,010	B-
7	69935	575	1	Fall	2,006	A+
8	86934	467	1	Spring	2,001	B+
9	34469	998	1	Spring	2,008	B
10	32650	606	1	Spring	2,009	C-
11	53246	844	1	Fall	2,009	C-
12	14341	116	1	Spring	2,001	B
13	72326	116	1	Spring	2,001	A
14	81456	404	1	Fall	2,008	B+
15	69935	749	1	Fall	2,001	A
16	89221	293	2	Fall	2,006	B-
17	94806	734	1	Fall	2,002	B
18	93520	617	1	Spring	2,003	C
19	74600	469	1	Fall	2,003	A+
20	53246	490	1	Spring	2,006	A-
21	32650	488	2	Spring	2,008	C-
22	85045	763	1	Fall	2,007	B
23	84963	225	1	Fall	2,003	A-

9. tampilan tabel advisor setelah diinput

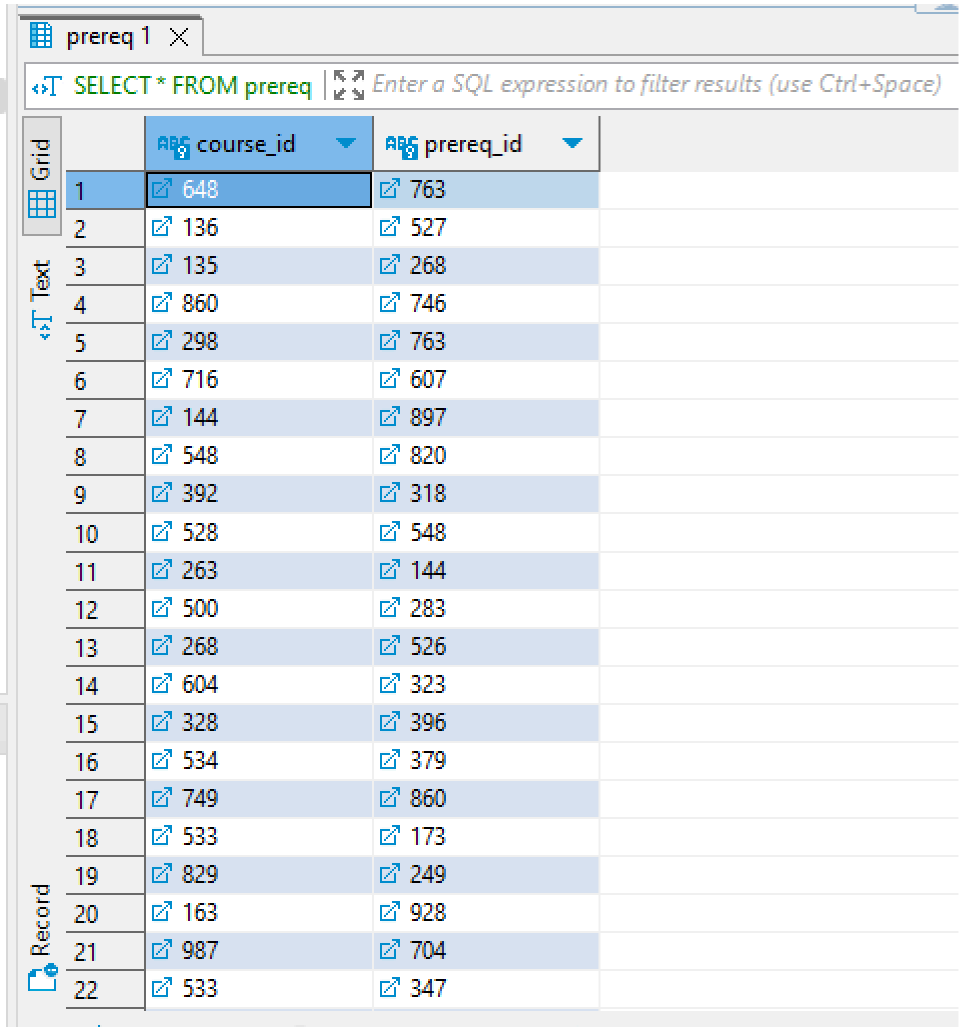


```
SELECT * FROM advisor
```

Enter a SQL expression to filter results (use C

	s_id	i_id
1	95633	97758
2	34469	23971
3	873	31510
4	85045	96644
5	2489	60585
6	33735	53204
7	86126	25920
8	78919	44831
9	3464	32765
10	71382	47852
11	89015	1249
12	54533	72848
13	68127	72848
14	44057	9554
15	65108	86316
16	58057	9888
17	91454	62688
18	53179	38668
19	94806	24217
20	63882	22362
21	39942	9888
22	89666	23365
23	87597	2780

10. tampilan tabel prereq setelah diinput



```
SELECT * FROM prereq
```

	course_id	prereq_id
1	648	763
2	136	527
3	135	268
4	860	746
5	298	763
6	716	607
7	144	897
8	548	820
9	392	318
10	528	548
11	263	144
12	500	283
13	268	526
14	604	323
15	328	396
16	534	379
17	749	860
18	533	173
19	829	249
20	163	928
21	987	704
22	533	347

program ini adalah sebuah program untuk menghasilkan data acak dan menyimpannya dalam file SQL yang akan kita lakukan langkah berikutnya untuk aterialized view dan transactions (masing-masing 5)

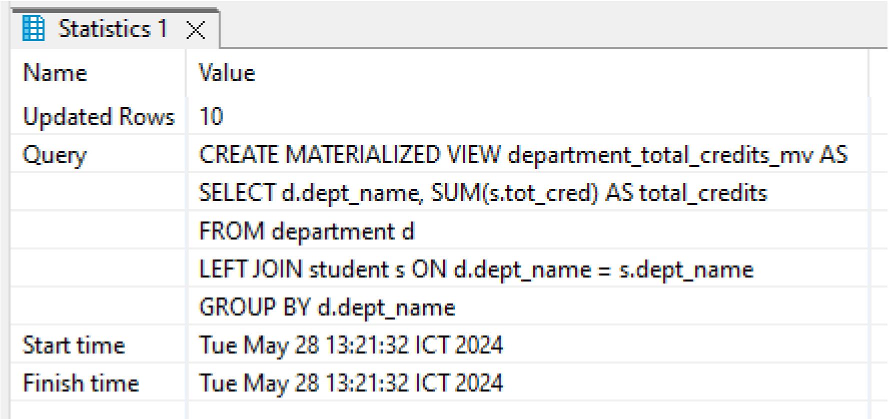
6. Buat contoh Materialized view dan transactions (masing-masing 5)

contoh Materialized view

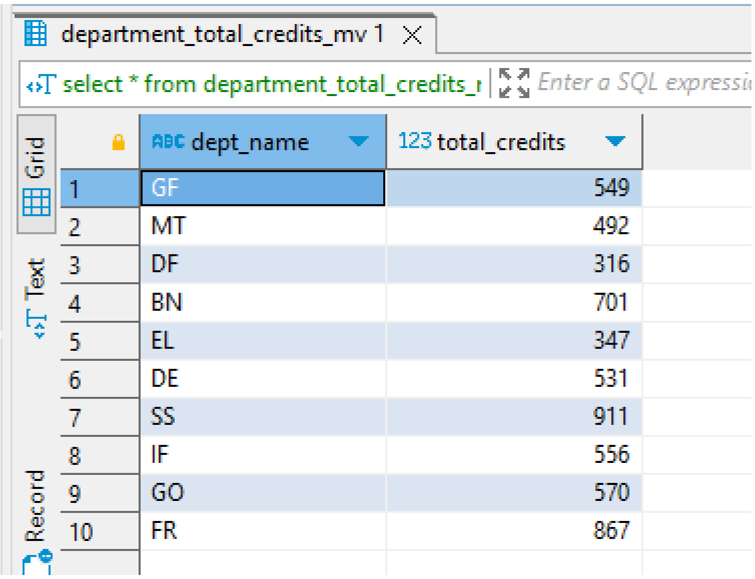
1. Total SKS yang Diperoleh Mahasiswa di Tiap Jurusan
   codenya :

```
CREATE MATERIALIZED VIEW department_total_credits_mv AS
SELECT d.dept_name, SUM(s.tot_cred) AS total_credits
FROM department d
LEFT JOIN student s ON d.dept_name = s.dept_name
GROUP BY d.dept_name;

-- Refresh tampilan materialized secara periodik
REFRESH MATERIALIZED VIEW department_total_credits_mv;
```

output :



Name	Value
Updated Rows	10
Query	CREATE MATERIALIZED VIEW department_total_credits_mv AS
	SELECT d.dept_name, SUM(s.tot_cred) AS total_credits
	FROM department d
	LEFT JOIN student s ON d.dept_name = s.dept_name
	GROUP BY d.dept_name
Start time	Tue May 28 13:21:32 ICT 2024
Finish time	Tue May 28 13:21:32 ICT 2024



select * from department_total_credits_r

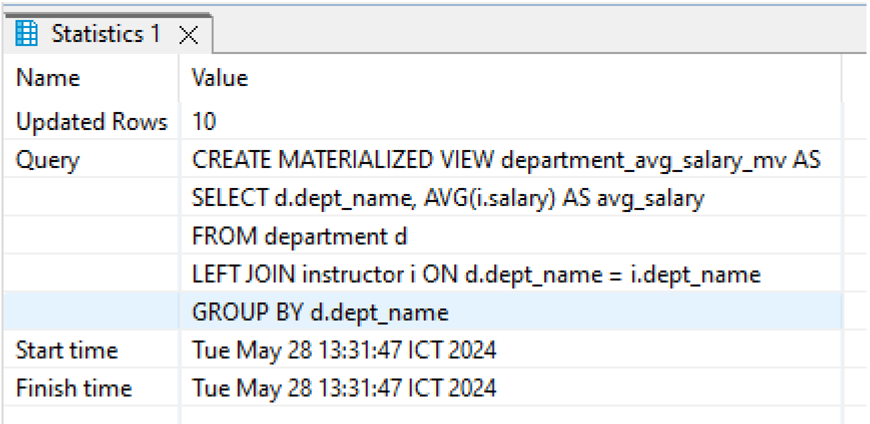
	dept_name	total_credits
1	GF	549
2	MT	492
3	DF	316
4	BN	701
5	EL	347
6	DE	531
7	SS	911
8	IF	556
9	GO	570
10	FR	867

disini akan terus di update Total SKS yang Diperoleh Mahasiswa di Tiap Jurusan

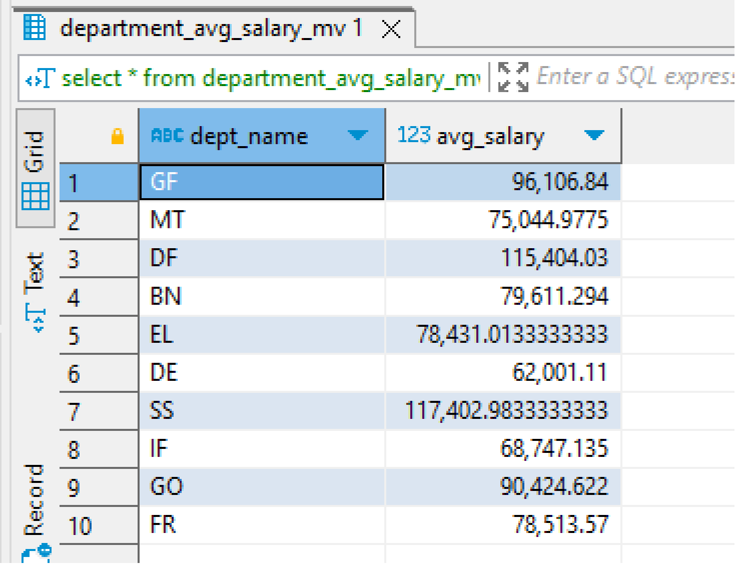
2. Gaji Rata-Rata Instruktur di Setiap Departemen
   codenya :

```
CREATE MATERIALIZED VIEW department_avg_salary_mv AS
SELECT d.dept_name, AVG(i.salary) AS avg_salary
FROM department d
LEFT JOIN instructor i ON d.dept_name = i.dept_name
GROUP BY d.dept_name;

-- Refresh the materialized view periodically
REFRESH MATERIALIZED VIEW department_avg_salary_mv;
```

outputnya :



Name	Value
Updated Rows	10
Query	CREATE MATERIALIZED VIEW department_avg_salary_mv AS
	SELECT d.dept_name, AVG(i.salary) AS avg_salary
	FROM department d
	LEFT JOIN instructor i ON d.dept_name = i.dept_name
	GROUP BY d.dept_name
Start time	Tue May 28 13:31:47 ICT 2024
Finish time	Tue May 28 13:31:47 ICT 2024



```
select * from department_avg_salary_mv
```

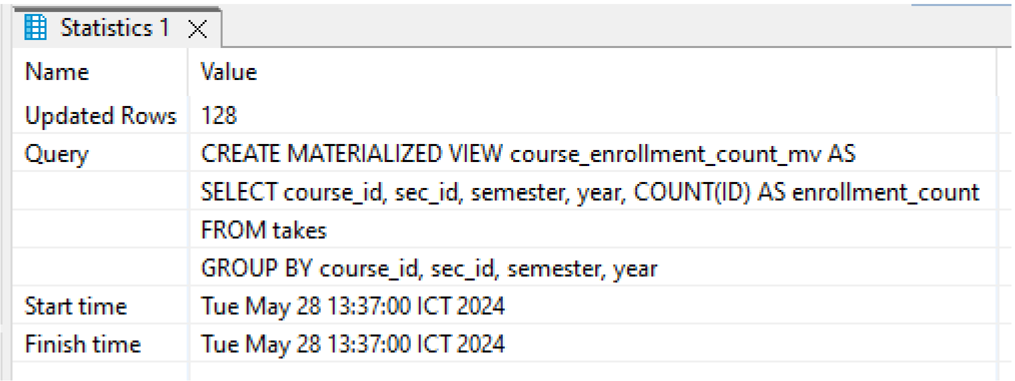
	dept_name	avg_salary
1	GF	96,106.84
2	MT	75,044.9775
3	DF	115,404.03
4	BN	79,611.294
5	EL	78,431.0133333333
6	DE	62,001.11
7	SS	117,402.9833333333
8	IF	68,747.135
9	GO	90,424.622
10	FR	78,513.57

disini dapat kita lihat untuk Gaji Rata-Rata Instruktur di Setiap Departemen yang dapat kita lihat secara periodik akan terus terupdate apabila transaksi terus dilakukan.

3. Jumlah Pendaftaran di Setiap Bagian Kursus
   codenya :

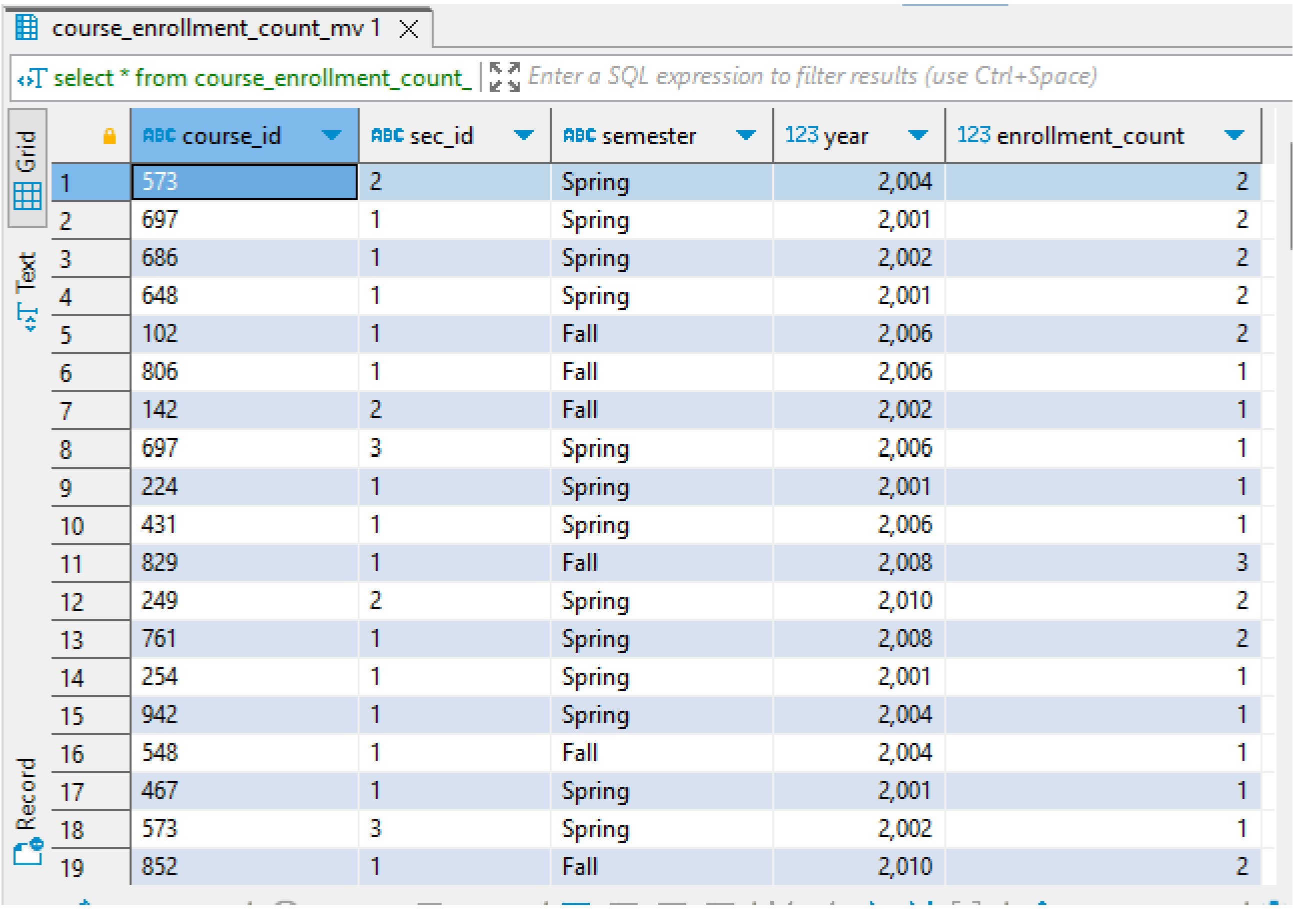
```
CREATE MATERIALIZED VIEW course_enrollment_count_mv AS
SELECT course_id, sec_id, semester, year, COUNT(ID) AS enrollment_count
FROM takes
GROUP BY course_id, sec_id, semester, year;

-- Refresh the materialized view periodically
REFRESH MATERIALIZED VIEW course_enrollment_count_mv;
```

outputnya :



Statistics 1

Name	Value
Updated Rows	128
Query	CREATE MATERIALIZED VIEW course_enrollment_count_mv AS
	SELECT course_id, sec_id, semester, year, COUNT(ID) AS enrollment_count
	FROM takes
	GROUP BY course_id, sec_id, semester, year
Start time	Tue May 28 13:37:00 ICT 2024
Finish time	Tue May 28 13:37:00 ICT 2024



course_enrollment_count_mv 1

select * from course_enrollment_count_

	course_id	sec_id	semester	year	enrollment_count
1	573	2	Spring	2,004	2
2	697	1	Spring	2,001	2
3	686	1	Spring	2,002	2
4	648	1	Spring	2,001	2
5	102	1	Fall	2,006	2
6	806	1	Fall	2,006	1
7	142	2	Fall	2,002	1
8	697	3	Spring	2,006	1
9	224	1	Spring	2,001	1
10	431	1	Spring	2,006	1
11	829	1	Fall	2,008	3
12	249	2	Spring	2,010	2
13	761	1	Spring	2,008	2
14	254	1	Spring	2,001	1
15	942	1	Spring	2,004	1
16	548	1	Fall	2,004	1
17	467	1	Spring	2,001	1
18	573	3	Spring	2,002	1
19	852	1	Fall	2,010	2

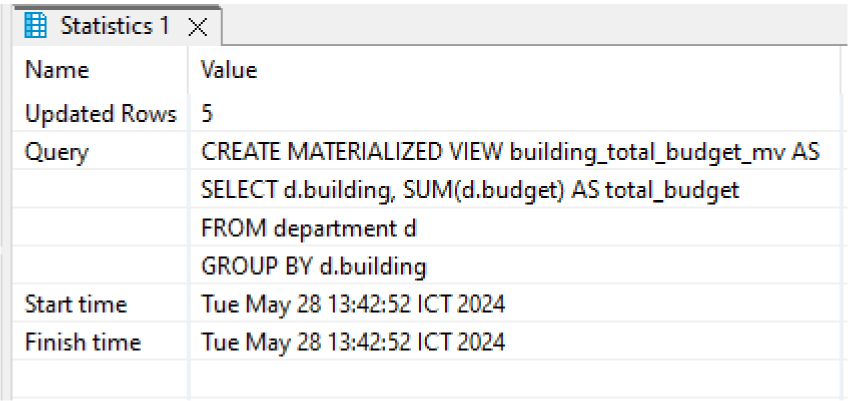
dari query diatas kita bisa nemampilkan data pendaftaran di setiap bagian kursus yang nantinya bisa kita lakukan terus secara update untuk menampilkan data yang terbaru dan sesuai dengan data transaksionalnya.

4. Total Alokasi Anggaran Tiap Gedung

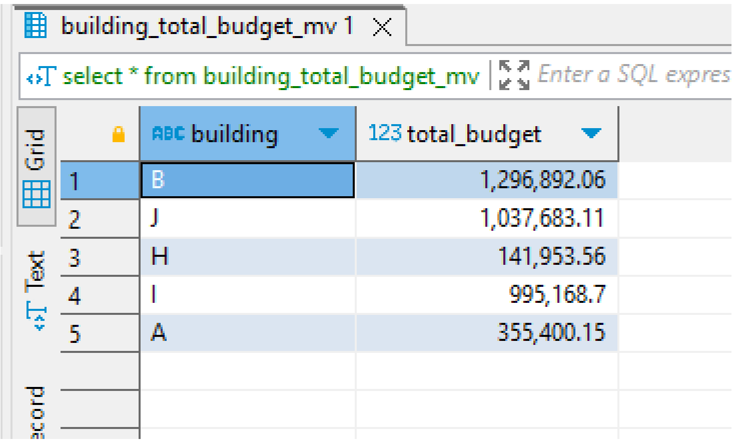
   codenya :

```
CREATE MATERIALIZED VIEW building_total_budget_mv AS
SELECT c.building, SUM(d.budget) AS total_budget
FROM department d
JOIN course c ON d.dept_name = c.dept_name
GROUP BY c.building;

-- Refresh the materialized view periodically
REFRESH MATERIALIZED VIEW building_total_budget_mv;
```

outputnya :



Name	Value
Updated Rows	5
Query	CREATE MATERIALIZED VIEW building_total_budget_mv AS
	SELECT d.building, SUM(d.budget) AS total_budget
	FROM department d
	GROUP BY d.building
Start time	Tue May 28 13:42:52 ICT 2024
Finish time	Tue May 28 13:42:52 ICT 2024



select * from building_total_budget_mv

	building	total_budget
1	B	1,296,892.06
2	J	1,037,683.11
3	H	141,953.56
4	I	995,168.7
5	A	355,400.15

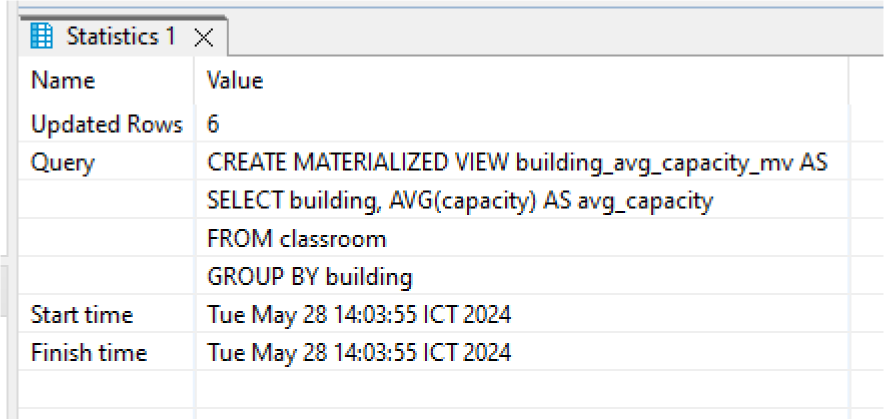
disini dapat kita lihat untuk dana yang dikeluarkan untuk anggaran dari setiap gedung, disini tentunya bisa akan terus terupdate apabila ada perubahan pada data yang inputan terbaru untuk anggaran bangunan tersebut

5. Rata-rata Kapasitas Ruangan Berdasarkan Bangunan

   codenya :

```
CREATE MATERIALIZED VIEW building_avg_capacity_mv AS
SELECT building, AVG(capacity) AS avg_capacity
FROM classroom
GROUP BY building;

-- Refresh the materialized view periodically
REFRESH MATERIALIZED VIEW building_avg_capacity_mv;
```

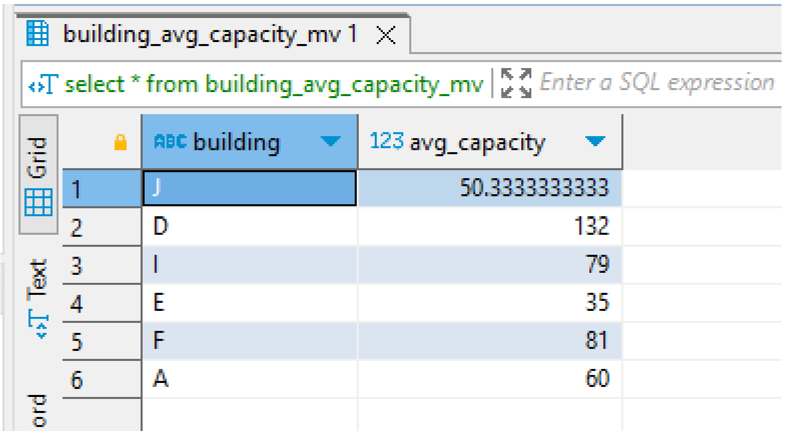
outputnya :



Name	Value
Updated Rows	6

Query	CREATE MATERIALIZED VIEW building_avg_capacity_mv AS
	SELECT building, AVG(capacity) AS avg_capacity
	FROM classroom
	GROUP BY building

Start time	Tue May 28 14:03:55 ICT 2024
Finish time	Tue May 28 14:03:55 ICT 2024


building_avg_capacity_mv 1
select * from building_avg_capacity_mv
Enter a SQL expression
Grid
Text
ord

	building	avg_capacity
1	J	50.3333333333
2	D	132
3	I	79
4	E	35
5	F	81
6	A	60

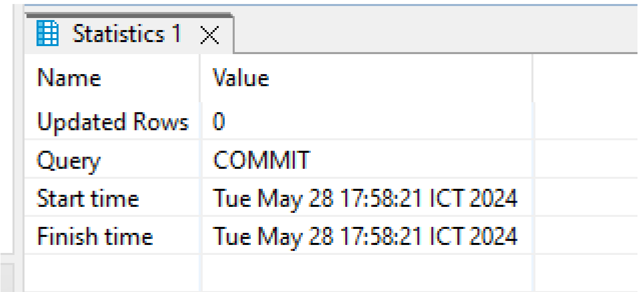
dari query tersebut kita bisa melihat rata rata untuk kapasitas di setiap ruangan berdasarkan gedungnya yang dapat dilihat dari tampilan tabel tersebut.

Contoh Transactions

Dalam melakukan transaksional, artinya mengubah dari isi entry tabel baik itu menambahkan data dan sebagainya, biasanya bentuknya berupa CRUT Create Read Update Delete.

1. Mendaftarkan Siswa Baru dan Mendaftar untuk Kursus
   codenya :

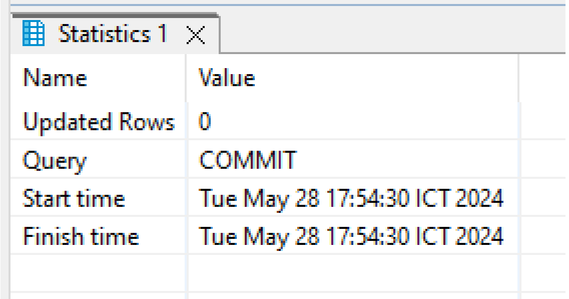
```
START TRANSACTION;

-- Insert a new student
INSERT INTO student (ID, name, dept_name, tot_cred)
VALUES ('S1001', 'Alice Smith', 'Computer Science', 0);

-- Enroll the new student in a course section
INSERT INTO takes (ID, course_id, sec_id, semester, year, grade)
VALUES ('S1001', 'CS101', '001', 'Fall', 2024, NULL);

COMMIT;
```



Name	Value
Updated Rows	0
Query	COMMIT
Start time	Tue May 28 17:58:21 ICT 2024
Finish time	Tue May 28 17:58:21 ICT 2024

2. Assign an Advisor to a Student

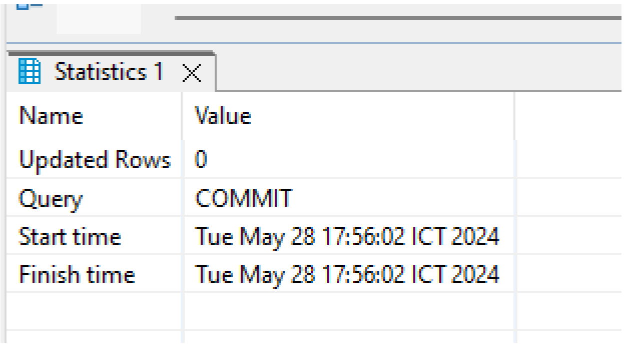
```
START TRANSACTION;

-- Check if the student exists
SELECT * FROM student WHERE ID = 'S1001' FOR UPDATE;

-- Assign an advisor to the student
INSERT INTO advisor (s_ID, i_ID)
VALUES ('S1001', 'I2001');

COMMIT;
```



Name	Value
Updated Rows	0
Query	COMMIT
Start time	Tue May 28 17:54:30 ICT 2024
Finish time	Tue May 28 17:54:30 ICT 2024

3. Update Instructor Salary and Budget
   codenya :

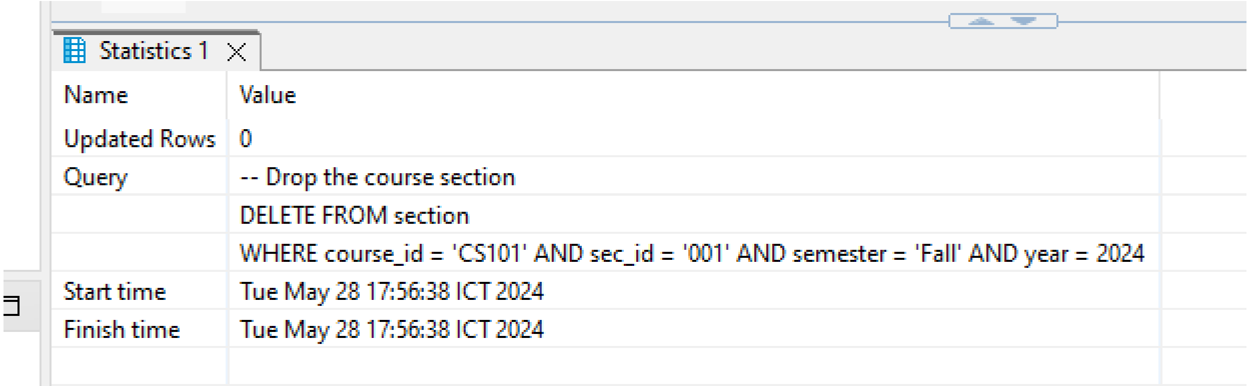
```
START TRANSACTION;

-- Update instructor salary
UPDATE instructor
SET salary = salary + 5000
WHERE ID = 'I2001';

-- Update department budget accordingly
UPDATE department
SET budget = budget - 5000
WHERE dept_name = (SELECT dept_name FROM instructor WHERE ID = 'I2001');

-- Ensure budget does not go negative
SELECT budget FROM department WHERE dept_name = (SELECT dept_name FROM instructor WHERE ID = 'I2

-- Rollback if budget becomes negative
SAVEPOINT budget_check;
IF (SELECT budget FROM department WHERE dept_name = (SELECT dept_name FROM instructor WHERE ID =
    ROLLBACK TO budget_check;
    RAISE EXCEPTION 'Budget cannot be negative';
END IF;

COMMIT;
```



Name	Value
Updated Rows	0
Query	COMMIT
Start time	Tue May 28 17:56:02 ICT 2024
Finish time	Tue May 28 17:56:02 ICT 2024

4. Drop a Course Section
   codenya :

```
START TRANSACTION;

-- Check if students are enrolled in the course section
SELECT * FROM takes WHERE course_id = 'CS101' AND sec_id = '001' AND semester = 'Fall' AND year

-- Drop the course section
DELETE FROM section
WHERE course_id = 'CS101' AND sec_id = '001' AND semester = 'Fall' AND year = 2024;

-- Ensure all related records in 'takes' are deleted as well (on delete cascade)

COMMIT;
```



Name	Value
Updated Rows	0

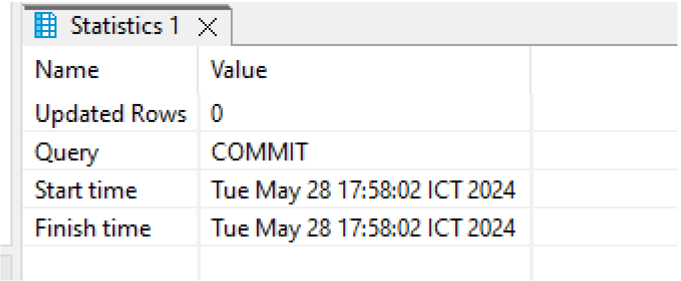
Query	-- Drop the course section
	DELETE FROM section
	WHERE course_id = 'CS101' AND sec_id = '001' AND semester = 'Fall' AND year = 2024

Start time	Tue May 28 17:56:38 ICT 2024
Finish time	Tue May 28 17:56:38 ICT 2024

5. Transfer a Student to a Different Department

   codenya :

```
START TRANSACTION;

-- Check if the student exists and current department
SELECT * FROM student WHERE ID = 'S1001' FOR UPDATE;

-- Update student's department
UPDATE student
SET dept_name = 'Electrical Engineering'
WHERE ID = 'S1001';

-- Adjust department budgets accordingly
UPDATE department
SET budget = budget + 5000
WHERE dept_name = 'Electrical Engineering';

UPDATE department
SET budget = budget - 5000
WHERE dept_name = 'Computer Science';

-- Ensure no department budget goes negative
SAVEPOINT budget_check;
IF (SELECT budget FROM department WHERE dept_name = 'Computer Science') < 0 THEN
    ROLLBACK TO budget_check;
    RAISE EXCEPTION 'Budget for Computer Science cannot be negative';
END IF;

IF (SELECT budget FROM department WHERE dept_name = 'Electrical Engineering') < 0 THEN
    ROLLBACK TO budget_check;
    RAISE EXCEPTION 'Budget for Electrical Engineering cannot be negative';
END IF;

COMMIT;
```



Name	Value
Updated Rows	0
Query	COMMIT
Start time	Tue May 28 17:58:02 ICT 2024
Finish time	Tue May 28 17:58:02 ICT 2024