

**LAPORAN TUGAS TEKNOLOGI BASIS DATA
MATERIALIZED VIEWS & TRANSACTIONS**



**Disusun oleh :
Anasthashya Rachman
121450013
KELAS RB**

**PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA
TAHUN AJARAN 2023/2034**

MATERIALIZED VIEWS & TRANSACTIONS

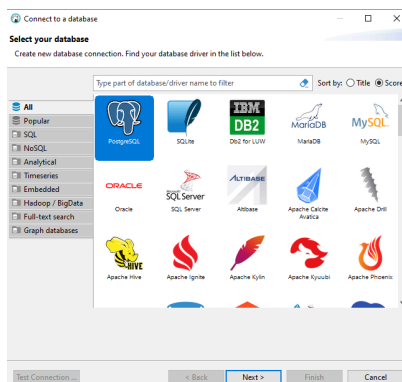
Tools:

1. Java Programming
2. DBMS Seperti Mysql, Postgress, MariaDB dll.

Tujuan: Mampu membuat material view dan transaction

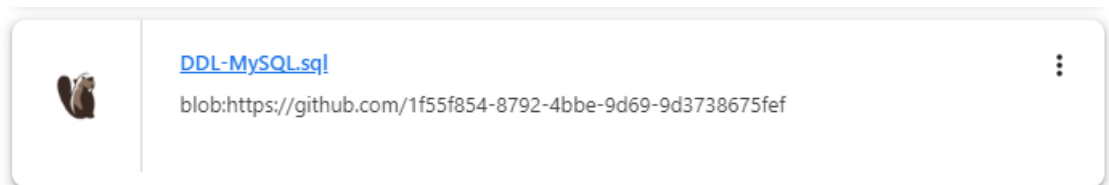
Deskripsi:

1. Pastikan DBMS sudah terinstall dan sedang dalam keadaan run



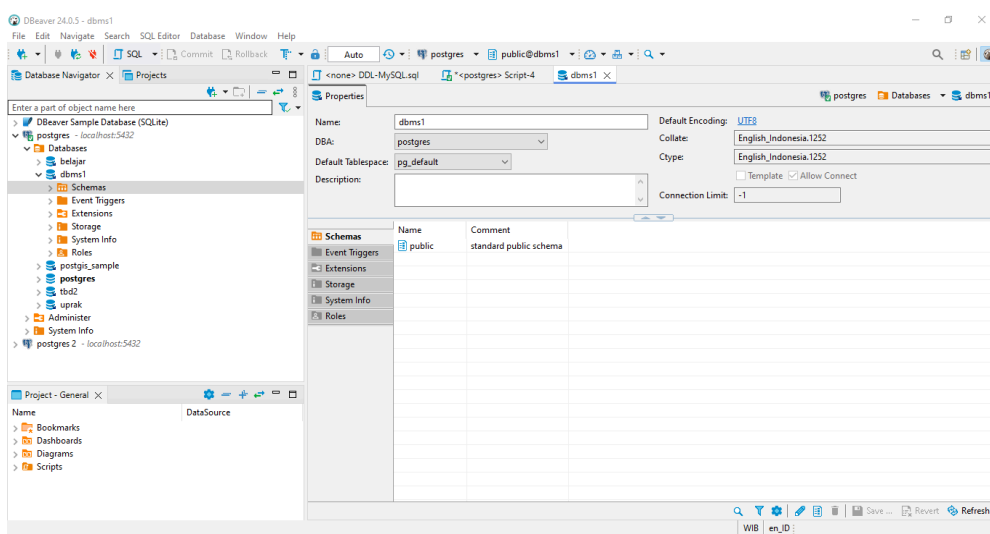
DBMS yang digunakan dalam tugas ini adalah postgresql dengan menggunakan dbeaver.

2. Download Query DDL. Download DDL



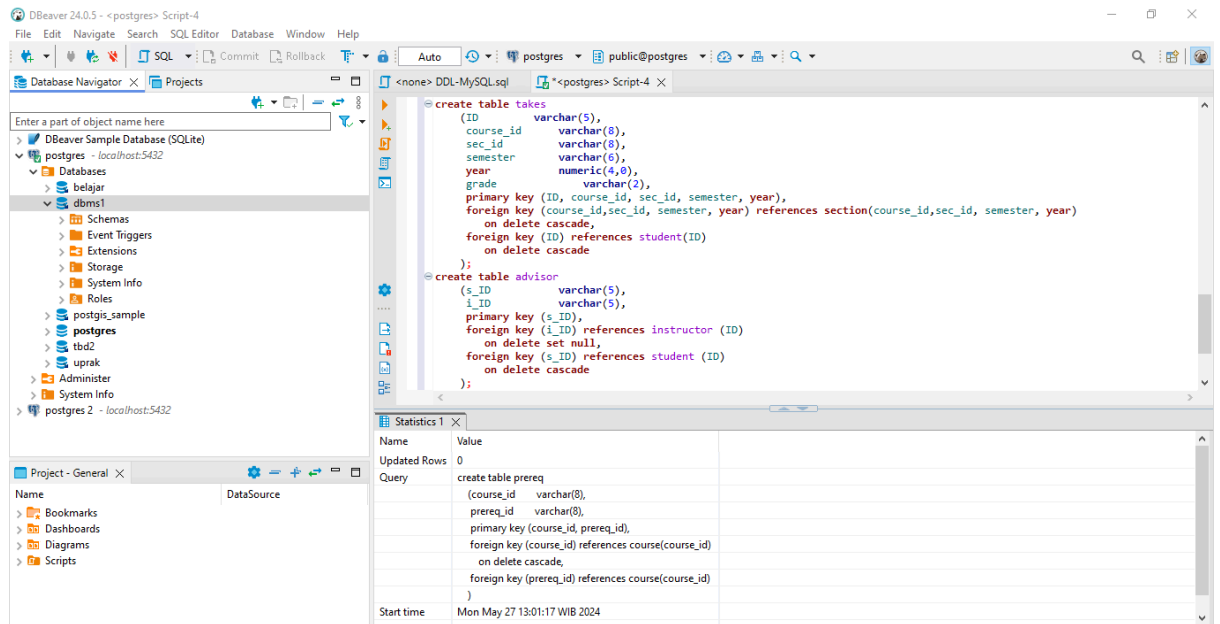
Melakukan download query untuk dapat menerapkan proses tuning indexing melalui query sql

3. Buat Schema Database. Contoh DBMS1



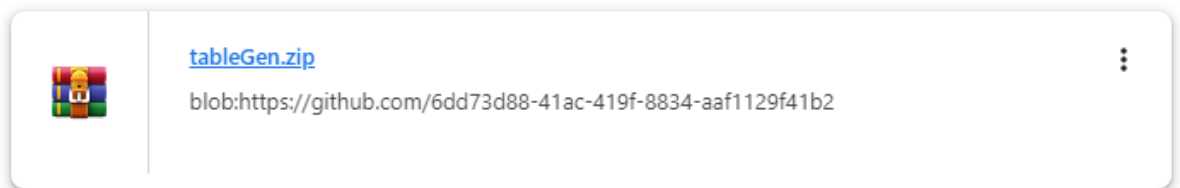
Proses ini dilakukan untuk membuat skema atau database dengan nama DBMS 1

4. Eksekusi SQL pada bagian (2)



Proses ini dilakukan dengan melakukan eksekusi query sql yang sebelumnya telah di download dan dihasilkan tabel untuk selanjutnya akan diinputkan valuenya

5. Download Code dalam bahasa pemrograman Java. Download Code



Tahapan ini dilakukan dengan mendownload code dalam bentuk java yang akan diubah dalam bentuk sql agar dapat dilakukan penginputan pada tabel yang telah dibuat sebelumnya

6. Compile dan jalan code tersebut pada komputer anda.

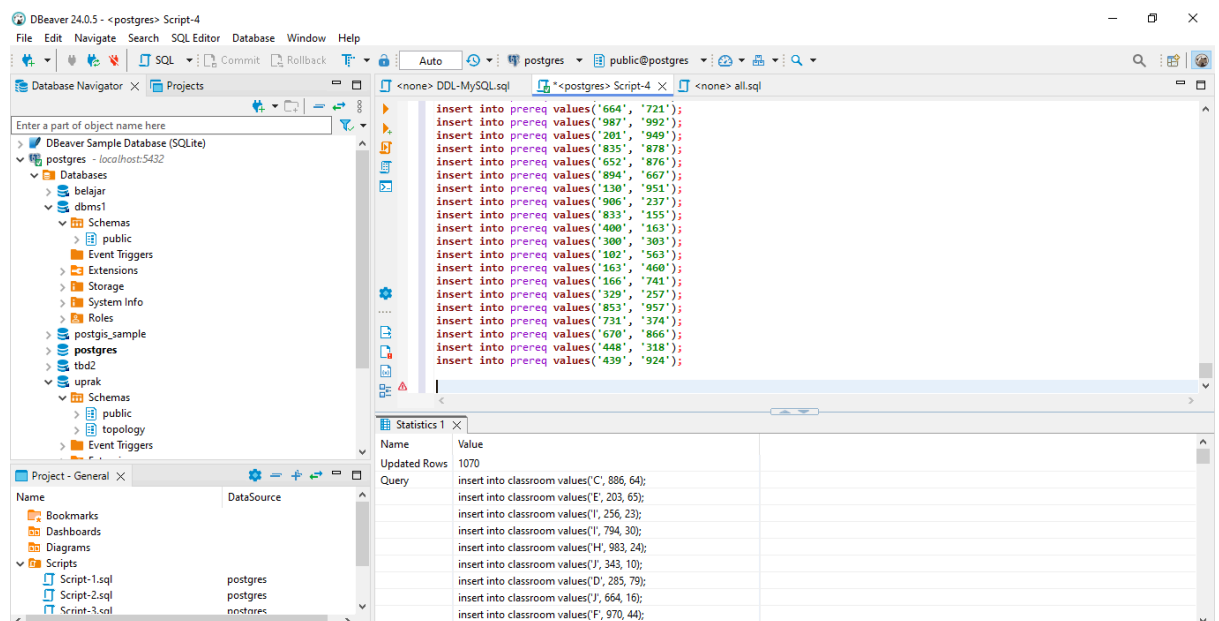
\$javac tableGen.java

\$java tableGen

```

C:\Windows\system32>cd /d H:\tbd\uprak
H:\tbd\uprak>javac tableGen.java
H:\tbd\uprak>java tableGen
'F', 694, 26
'G', 816, 10
'E', 118, 111
'G', 424, 10
'G', 56, 107
'E', 696, 128
'A', 581, 12
'G', 507, 23
'I', 583, 164
'B', 523, 29
'RQ', 'H', 615416.27
RQ/RQ
HI/RQ
'HI', 'C', 818115.73
DK/RQ
DK/HI
'DK', 'G', 584723.90
HH/RQ
HH/HI
HH/DK
'HH', 'D', 154477.62
EL/RQ
EL/HI
EL/DK
EL/HH

```



Proses ini dilakukan dengan pengubahan file java menjadi sql dan dilanjutkan dengan input data sql dalam tabel

7. Menampilkan proses query

- Menampilkan tabel classrom

SELECT * FROM student

student 1 X

SELECT * FROM student | Enter a SQL expression to filter results (use Ctrl+Space)

	ABC id	ABC name	ABC dept_name	123 tot_cred
1	20720	Kiki	RR	127
2	65757	Yohan	HH	3
3	54626	Yohan	EL	8
4	36324	Ande	EL	78
5	18110	rahmat	DF	110
6	97060	Budi	MT	90
7	79428	Yohan	SS	94
8	54408	Ande	MT	109
9	56600	Kiki	HH	21
10	61885	Ande	EL	20
11	89280	Budi	DF	56
12	33329	Johan	DF	82

- Menampilkan tabel departement

SELECT * FROM department

department 1 X

SELECT * FROM department | Enter a SQL expression to filter results (usi

	ABC dept_name	ABC building	123 budget
1	DE	G	485,966.67
2	RR	D	275,956.77
3	SS	B	725,764.38
4	RQ	G	861,369.52
5	ED	J	5,628.76
6	MT	J	93,401.1
7	EL	C	523,219.18
8	DK	G	167,259.17
9	HH	I	561,924.85
10	DF	J	796,521.17

- Menampilkan tabel course

SELECT * FROM course

course 1 X

SELECT * FROM course | Enter a SQL expression to filter results (use Ctrl+Space)

	ABC course_id	ABC title	ABC dept_name	123 credits
1	953	IF0230	RR	3
2	781	IF3211	HH	3
3	552	IF4321	RR	3
4	125	IF3030	DK	3
5	961	IF3022	EL	4
6	713	IF3022	DF	3
7	667	IF3120	RR	4
8	830	IF3120	DE	4
9	496	IF3022	DK	3
10	231	IF5555	HH	4
11	957	IF5555	EL	3
12	300	IF4041	EL	4
13	453	IF3120	RQ	3

- Menampilkan tabel instructor

SELECT * FROM instructor

instructor 1 X

SELECT * FROM instructor | Enter a SQL expression to filter results (use Ctrl+Sp

	ABC id	ABC name	ABC dept_name	123 salary
1	12116	Josu	HH	37,007.95
2	76863	yuyun	ED	127,361.41
3	41377	Ahmad	SS	33,165.29
4	94237	Josu	MT	86,818.99
5	8312	Ande	RR	95,843.78
6	69914	rahmat	MT	106,132.6
7	39025	Kiki	SS	126,861.04
8	51284	Ande	DK	115,406.2
9	32005	Johan	RQ	123,526.82
10	85648	Yohan	HH	58,117.23
11	93840	Kiki	RR	64,671.29
12	18961	Johan	DE	85,606.53
13	16400	Yohan	DF	123,042.57

- Menampilkan tabel section

SELECT * FROM section

section 1

SELECT * FROM section Enter a SQL expression to filter results (use Ctrl+Space)

	id course_id	id sec_id	id semester	id year	id building	id room
1	690	1	Spring	2,006	E	203
2	428	1	Fall	2,006	D	285
3	749	1	Fall	2,007	F	970
4	545	1	Spring	2,002	I	794
5	150	1	Spring	2,001	I	794
6	830	1	Fall	2,005	E	203
7	506	1	Spring	2,008	J	343
8	303	1	Fall	2,003	I	794
9	195	1	Spring	2,006	I	256
10	302	1	Spring	2,006	J	343
11	626	1	Fall	2,001	E	203
12	987	1	Fall	2,003	I	256

- Menampilkan tabel teaches

SELECT * FROM teaches

teaches 1

SELECT * FROM teaches Enter a SQL expression to filter results (use Ctrl+Space)

	id id	id course_id	id sec_id	id semester	id year
1	90161	489	1	Fall	2,006
2	75342	329	2	Spring	2,009
3	26737	424	3	Fall	2,009
4	15437	102	2	Spring	2,001
5	40047	362	1	Fall	2,008
6	32470	366	1	Spring	2,005
7	75342	144	1	Spring	2,004
8	22215	428	1	Fall	2,006
9	63646	563	1	Fall	2,001
10	23063	670	1	Fall	2,008
11	23063	331	1	Fall	2,006
12	30097	913	1	Fall	2,010
13	51394	132	1	Fall	2,003

- Menampilkan tabel students

SELECT * FROM student

student 1

SELECT * FROM student Enter a SQL expression to filter results (use Ctrl+Space)

	id id	id name	id dept_name	id tot_cred
1	20720	Kiki	RR	127
2	65757	Yohan	HH	3
3	54626	Yohan	EL	8
4	36324	Ande	EL	78
5	18110	rahmat	DF	110
6	97060	Budi	MT	90
7	79428	Yohan	SS	94
8	54408	Ande	MT	109
9	56600	Kiki	HH	21
10	61885	Ande	EL	20
11	89280	Budi	DF	56
12	33329	Johan	DF	82
13	59556	Johan	MT	67

- Menampilkan tabel takes

SELECT * FROM takes

takes 1

SELECT * FROM takes Enter a SQL expression to filter results (use Ctrl+Space)

	id id	id course_id	id sec_id	id semester	id year	id grade
1	89280	489	2	Fall	2,001	B-
2	46982	512	1	Fall	2,006	A
3	39194	171	1	Spring	2,006	A-
4	50212	741	3	Spring	2,010	C
5	36324	217	3	Fall	2,010	C-
6	40766	918	1	Spring	2,008	B-
7	14943	329	3	Fall	2,010	B+
8	34862	457	1	Fall	2,001	A
9	80203	428	1	Fall	2,006	A+
10	54285	102	2	Spring	2,001	A-
11	59556	621	2	Fall	2,007	A
12	6846	690	2	Spring	2,004	A+

- Menampilkan tabel advisor

SELECT * FROM advisor

advisor 1

SELECT * FROM advisor

	asc s_id	asc i_id
1	20720	40374
2	65757	27085
3	54626	38292
4	36324	78270
5	18110	39025
6	97060	78071
7	79428	97912
8	54408	27085
9	56600	32470
10	61885	78270
11	89280	51284
12	33329	38292
13	59556	96993

- Menampilkan tabel prereq

SELECT * FROM prereq

prereq 1

SELECT * FROM prereq

	asc course_id	asc prereq_id
1	626	123
2	396	785
3	681	951
4	598	496
5	918	785
6	237	143
7	453	199
8	407	637
9	257	372
10	396	907
11	835	794
12	220	962
13	505	801

- Menampilkan student yang memiliki total credit besar dari 30

SELECT * FROM student WHERE tot_cred > 30;

student 1

SELECT * FROM student WHERE tot_cred > 30;

	asc id	asc name	asc dept_name	123 tot_cred
2	36324	Ande	EL	78
3	18110	rahmat	DF	110
4	97060	Budi	MT	90
5	79428	Yohan	SS	94
6	54408	Ande	MT	109
7	89280	Budi	DF	56
8	33329	Johan	DF	82
9	59556	Johan	MT	67
10	80885	Yohan	ED	91
11	54444	Josu	DE	42
12	6780	Budi	SS	103
13	84389	Yohan	HH	31
14	46681	rahmat	SS	66
15	27452	Kiki	DE	82
16	94276	Adri	ED	98
17	53760	Ahmad	SS	67
18	92021	Ahmad	MT	108
19	61347	Kiki	DE	111
20	33607	Adri	DK	113
21	39194	yuyun	EL	81
22	54902	Budi	DK	78
23	9400	Kiki	MT	81
24	12161	Josu	DE	74

Terdapat 76 siswa yang memiliki total kredit lebih dari 30

- Menampilkan nama departemen dari siswa yang memiliki total kredit lebih dari 30

```
SELECT dept_name FROM student WHERE tot_cred > 30;
```

dept_name
RR
EL
DF
MT
SS
MT
DF
DF
MT
ED
DE
SS
HH
SS

- Menampilkan data dari tabel takes yang dilakukan operasi join berdasarkan id students

```
SELECT * FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course_id = section.course_id
```

id	course_id	sec_id	semester	year	grade
89280	489	2	Fall	2,001	B-
89280	489	2	Fall	2,001	B-
89280	489	2	Fall	2,001	B-
46982	512	1	Fall	2,006	A
39194	171	1	Spring	2,006	A-
39194	171	1	Spring	2,006	A-
50212	741	3	Spring	2,010	C
50212	741	3	Spring	2,010	C
50212	741	3	Spring	2,010	C
36324	217	3	Fall	2,010	C-
36324	217	3	Fall	2,010	C-
36324	217	3	Fall	2,010	C-

Terdapat 401 data hasil dari tabel takes yang dilakukan operasi join berdasarkan id students

- Menampilkan informasi yang menggabungkan detail mahasiswa, mata kuliah yang diambil, serta informasi ruang dan gedung tempat mata kuliah

```
SELECT
    student.name,
    student.dept_name,
    takes.sec_id AS pengambilan,
    takes.semester,
    section.room_number,
    section.building,
    course.course_id,
    course.dept_name AS course_dept_name
FROM
    takes
JOIN
    student ON takes.ID = student.ID
JOIN
    section ON takes.course_id = section.course_id AND takes.sec_id = section.sec_id AND takes.semester = section.semester
JOIN
    course ON section.course_id = course.course_id;
```

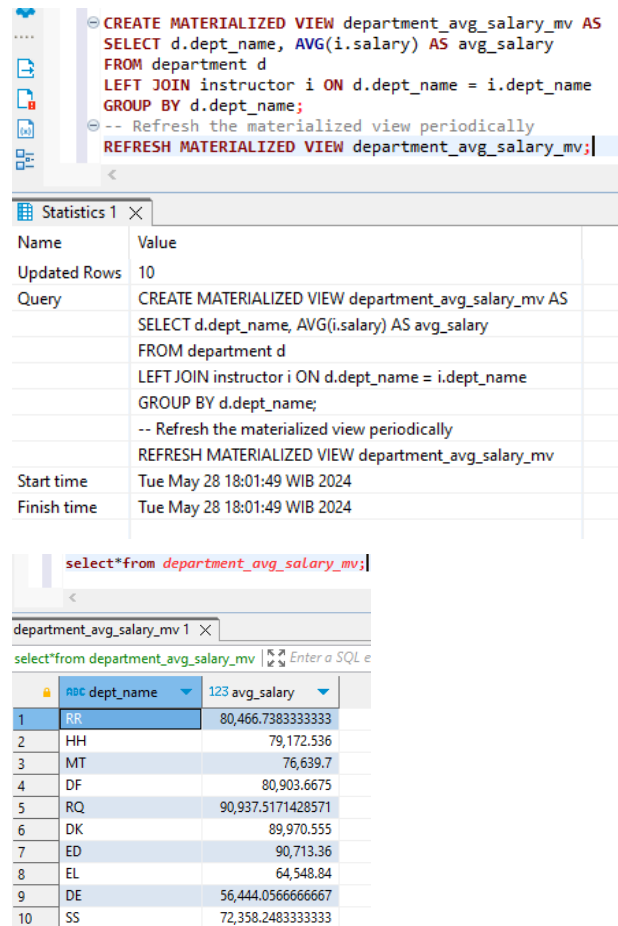
name	dept_name	pengambilan	semester	room_number
Budi	DF	2	Fall	983
Kiki	DE	1	Fall	983
yuyun	EL	1	Spring	983
Adri	DE	3	Spring	256
Ande	EL	3	Fall	836
yuyun	RQ	1	Spring	970
Johan	RQ	3	Fall	836
yuyun	FI	1	Fall	704

Terdapat 200 data

Pada tahapan ini dilakukan proses query yang terdapat dalam laporan dan query untuk menampilkan hasil dari input data dalam tabel.

8. Buat contoh Materialized view dan transactions

- Menampilkan gaji rata-rata instruktur tiap departemen dengan materialized view



```

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;

```

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; -- Refresh the materialized view periodically REFRESH MATERIALIZED VIEW department_avg_salary_mv
Start time	Tue May 28 18:01:49 WIB 2024
Finish time	Tue May 28 18:01:49 WIB 2024

```

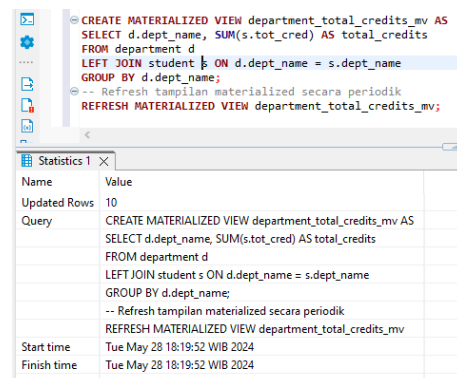
select * from department_avg_salary_mv;

```

dept_name	avg_salary
RR	80,466.7383333333
HH	79,172.536
MT	76,639.7
DF	80,903.6675
RQ	90,937.5171428571
DK	89,970.555
ED	90,713.36
EL	64,548.84
DE	56,444.0566666667
SS	72,358.2483333333

Didapatkan hasil 10 departement dengan rata-rata gaji tertinggi pada departemen RQ dan terendah pada departemen EL

- Menampilkan total kredit dari departemen



```

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;

```

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; -- Refresh tampilan materialized secara periodik REFRESH MATERIALIZED VIEW department_total_credits_mv
Start time	Tue May 28 18:19:52 WIB 2024
Finish time	Tue May 28 18:19:52 WIB 2024

select*from department_total_credits_mv;

department_total_credits_mv 1 X

select*from department_total_credits_mv; Enter a SQL expression

	dept_name	total_credits
1	RR	349
2	HH	424
3	MT	1,025
4	DF	772
5	RQ	965
6	DK	391
7	ED	526
8	EL	591
9	DE	860
10	SS	693

Didapatkan hasil 10 departemen dengan total kredit terbanyak pada departemen MT dan tersedikit pada RR

- Menampilkan jumlah kursus

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;

Statistics 1 X

Name	Value
Updated Rows	125
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; -- Refresh the materialized view periodically REFRESH MATERIALIZED VIEW course_enrollment_count_mv
Start time	Tue May 28 18:23:03 WIB 2024
Finish time	Tue May 28 18:23:03 WIB 2024

select*from course_enrollment_count_mv;

course_enrollment_count_mv 1 X

select*from course_enrollment_count_mv; Enter a SQL expression to filter results (use Ctrl+Space)

	course_id	sec_id	semester	year	enrollment_count
1	240	3	Spring	2,004	2
2	568	1	Spring	2,001	1
3	263	1	Fall	2,007	1
4	428	1	Fall	2,006	1
5	866	3	Spring	2,002	1
6	102	2	Spring	2,001	3
7	278	1	Spring	2,009	2
8	366	1	Spring	2,005	2
9	318	1	Fall	2,006	3
10	150	1	Spring	2,001	2
11	248	1	Spring	2,007	1
12	545	1	Spring	2,002	2

Didapatkan hasil jumlah enrollment terbanyak adalah 3 dari jumlah kursus dan tersedikit adalah 1

- Menampilkan rata-rata kapasitas ruangan

<pre>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</pre>	
Statistics 1	
Name	Value
Updated Rows	8
Query	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
Start time	Tue May 28 18:26:31 WIB 2024
Finish time	Tue May 28 18:26:31 WIB 2024

select*from building_avg_capacity_mv;	
building_avg_capacity_mv 1	
select*from building_avg_capacity_mv Enter a SQL expres	
ABC building	123 avg_capacity
1 C	64
2 J	13
3 H	24
4 D	79
5 I	26.5
6 E	65
7 F	44
8 A	14

Didapatkan ruangan D adalah ruangan dengan kapasitas terbesar dan kapasitas terkecil adalah ruangan J

- Menampilkan alokasi anggaran gedung

Statistics 1	
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

building_total_budget_mv 1	
select * from building_total_budget_mv Enter a SQL expres	
ABC building	123 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

Didapatkan total anggaran terbesar pada gedung B dan anggaran termurah pada gedung A

Pada tahapan ini dilakukan dengan penerapan dari implementasi Materialized view dan transactions yang berguna dalam menyimpan hasil dari query secara fisik di dalam disk.

9. Tunning Query

Proses ini dilakukan tunning indexing dimana tunning indexing adalah proses pengoptimalan kinerja basis data dengan cara merancang dan mengelola indeks secara efektif. Indeks merupakan struktur data tambahan yang dibuat di atas tabel untuk mempercepat pencarian, pengurutan, dan pengelompokan data. Dengan menggunakan indeks yang tepat, query dapat dieksekusi lebih cepat dan beban kerja basis data dapat dikurangi. Berikut merupakan hasil dari tunning indexing:

Query	Waktu (second)
SELECT * FROM student	3,55
SELECT * FROM student WHERE tot_cred > 30;	2,42
SELECT `name`, department FROM student WHERE tot_cred > 30;	1,45
SELECT * FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course_id = section.course_id	1,41
SELECT student.`name`, student.dept_name, takes. sec_id AS pengambilan, takes.semester, section.roo m_number, section.building, course.cours e_id, course.dept_name FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course_id = section.course_id JOIN course ON section.course_id = course.course_id	1,3

Pada tabel dapat terlihat jelas bahwa kolom query pertama adalah yang belum dilakukan indexing sehingga, menggunakan waktu yang lebih lama dalam mengeksekusinya. Sedangkan, query yang telah dilakukan tunning indexing seperti pada kolom query 2-5 menggunakan waktu yang relatif lebih singkat.

KESIMPULAN

Dalam tugas ini, dilakukan menerapkan materialized views dan transactions menggunakan PostgreSQL sebagai DBMS dengan bantuan DBeaver untuk pengelolaan basis data. Penerapan materialized views memungkinkan untuk menyimpan hasil query yang kompleks secara fisik di dalam disk, yang meningkatkan performa query berulang yang membutuhkan data agregat atau hasil perhitungan yang sama. Sebagai contoh, dibuat penerapan materialized view untuk menghitung total anggaran per gedung, yang dapat diperbarui secara manual atau periodik untuk memastikan data tetap up-to-date.

Selain itu, dilakukan pemanfaatan transactions untuk menjaga integritas dan konsistensi data selama operasi basis data yang kompleks. Transactions memungkinkan untuk mengelompokkan beberapa operasi SQL menjadi satu unit kerja atomik, sehingga semua operasi tersebut dieksekusi secara keseluruhan atau dibatalkan seluruhnya jika terjadi kesalahan. Hal ini memastikan bahwa basis data tetap dalam keadaan konsisten meskipun terjadi kegagalan selama proses eksekusi.

Penggunaan materialized views dan transactions dalam pengelolaan basis data memberikan manfaat signifikan dalam hal performa dan integritas data. Materialized views mengurangi beban komputasi untuk query yang sering dijalankan dengan hasil yang relatif statis, sementara transactions memberikan jaminan bahwa data di dalam basis data tetap konsisten dan akurat, bahkan dalam situasi kegagalan sistem. Implementasi ini menunjukkan bagaimana teknologi basis data dapat dioptimalkan untuk mendukung kebutuhan pengolahan data yang efisien dan andal. Penggunaan indexing juga digunakan untuk efisiensi query sehingga, memungkinkan lebih cepat dan tepat dalam penggunaan query.