Experiment 5

Student Name: Tanmay Singh   **UID: 23BCS10799**
Branch: BE-CSE   **Section/Group: KRG-3B**
Semester: 5   **Date of performance: 26-09-2025**
Subject name: ADBMS   **Subject code: 23CSP-333**

1. Problem Description/Aim:

Medium-Problem Title: Generate 1 million records per ID in 'transaction_data'

using generate_series() and random() ,create a normal view

and a materialized view 'sales_summary' with aggregated

metrics (total_quantity_sold , total_sales, total_orders) , and

compare their performance and execution time.

Procedure (Step-by-Step):

- Create a large dataset:- Create a table names transaction_data (id , value) with 1 million records.- take id 1 and 2, and for each id, generate 1 million records in value column- Use Generate_series () and random() to populate the data.
- Create a normal view and materialized view to for sales_summary, which includes total_quantity_sold, total_sales, and total_orders with aggregation.
- Compare the performance and execution time of both.

Sample Output Description:

The transaction_data table has 2 million rows (1 million per ID) with random values.

The normal view sales_summary computes aggregates on the fly, while the

materialized view sales_summary_mv stores precomputed results. Queries on the

materialized view are much faster, but it needs refreshing when data changes,

whereas the normal view always shows up-to-date results.

Hard-Problem Title: Create restricted views in the sales database to provide

summarized, non-sensitive data to the reporting team, and control

access using DCL commands(GRANT and REVOKE).

Procedure (Step-by-Step):

- Create restricted views-- Define views that show only aggregated sales data (e.g., total_sales, total_orders) without exposing sensitive columns like customer details or payment info.

2. Codes

------------------------------MEDIUM LEVEL PROBLEM ---

```
Create table TRANSACTION_DATA(id int,val decimal);

INSERT INTO TRANSACTION_DATA(ID,VAL)

SELECT 1,RANDOM()

FROM GENERATE_SERIES(1,1000000);


INSERT INTO TRANSACTION_DATA(ID,VAL)

SELECT 2,RANDOM()

FROM GENERATE_SERIES(1,1000000);

SELECT * FROM TRANSACTION_DATA;


CREATE or REPLACE VIEW SALES_SUMMARY AS

SELECT

ID,

COUNT(*) AS total_quantity_sold,

sum(val) AS total_sales,

count(distinct id) AS total_orders

FROM TRANSACTION_DATA

GROUP BY ID;


EXPLAIN ANALYZE

SELECT * FROM SALES_SUMMARY; /*Simple view */


CREATE MATERIALIZED VIEW SALES_SUMM_MV AS
```

SELECT

ID,

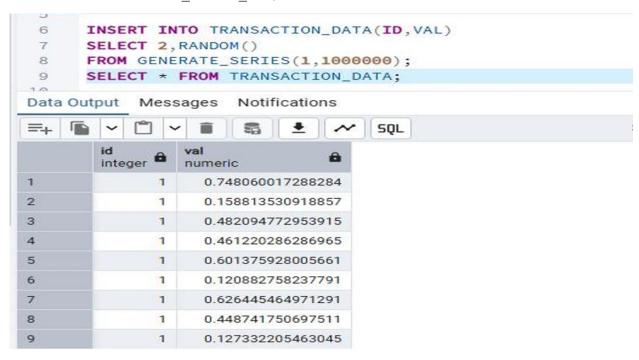COUNT(*) AS total_quantity_sold,

sum(val) AS total_sales,

count(distinct id) AS total_orders

FROM TRANSACTION_DATA

GROUP BY ID;

EXPLAIN ANALYZE

SELECT * FROM SALES_SUMM_MV; /*Materialized view*/

```
6    INSERT INTO TRANSACTION_DATA(ID,VAL)
7    SELECT 2,RANDOM()
8    FROM GENERATE_SERIES(1,1000000);
9    SELECT * FROM TRANSACTION_DATA;
```

Data Output   Messages   Notifications

id integer	val numeric
1	0.748060017288284
1	0.158813530918857
1	0.482094772953915
1	0.461220286286965
1	0.601375928005661
1	0.120882758237791
1	0.626445464971291
1	0.448741750697511
1	0.127332205463045

```
21   SELECT * FROM SALES_SUMMARY; /*Simple view */
```

Data Output   Messages   Notifications

Showir

id integer	total_quantity_sold bigint	total_sales numeric	total_orders bigint
1	1	2000000	1000226.201610874170319933640	1
2	2	1000000	499473.47586932728250459408	1

```
20   EXPLAIN ANALYZE
21   SELECT * FROM SALES_SUMMARY; /*Simple view *
```

Data Output   Messages   Notifications

	QUERY PLAN text
1	GroupAggregate (cost=471514.97..509014.99 rows=2 width=52) (a
2	Group Key: transaction_data.id
3	-> Sort (cost=471514.97..479014.97 rows=3000000 width=15) (ac
4	Sort Key: transaction_data.id
5	Sort Method: external merge Disk: 73504kB
6	-> Seq Scan on transaction_data (cost=0.00..46224.00 rows=3
7	Planning Time: 0.135 ms
8	Execution Time: 4396.880 ms

```
33    SELECT * FROM SALES_SUMM; /*Materialized view*/
```

Data Output   Messages   Notifications

Show

id integer	total_quantity_sold bigint	total_sales numeric	total_orders bigint
1	1	1000000	500106.667545326356598143529	1
2	2	1000000	499473.475869327728250459408	1

Data Output   Messages   Notifications

Showing rows: 1

QUERY PLAN text
1	Seq Scan on sales_summ (cost=0.00..20.20 rows=1020 width=52) (actual time=0.017..0.018 rows=2 loops=...
2	Planning Time: 0.063 ms
3	Execution Time: 0.032 ms

-----------------------------------HARD PROBLEM-----------------------------------

CREATE TABLE customer_data (

    transaction_id SERIAL PRIMARY KEY,

    customer_name VARCHAR(100),

    email VARCHAR(100),

    phone VARCHAR(15),

    payment_info VARCHAR(50), -- sensitive

    order_value DECIMAL,

    order_date DATE DEFAULT CURRENT_DATE

);

-- Insert sample data

INSERT INTO customer_data (customer_name, email, phone, payment_info, order_value)

VALUES

```
('Tanmay Singh', 'tanmay@example.com', '9040122324', '1234-5678-9012-3456', 500),

('Aniket Chugh', 'aniket@example.com', '9040122324', '1234-5678-9012-3456', 1000),

('Jyoti Kumari', 'jyoti@example.com', '9876543210', '9876-5432-1098-7654', 700),

('Rohan', 'rohan@example.com', '9876543210', '9876-5432-1098-7654', 300);


CREATE OR REPLACE VIEW RESTRICTED_SALES_DATA AS
SELECT
CUSTOMER_NAME,
COUNT(*) AS total_orders,
SUM(order_value) as total_sales
from customer_data
group by customer_name;


select * from restricted_sales_data;


CREATE USER CLIENT1 WITH PASSWORD 'REPORT1234';
GRANT SELECT ON RESTRICTED_SALES_DATA TO CLIENT1;
REVOKE SELECT ON RESTRICTED_SALES_DATA FROM CLIENT1;
```

Query Query History

```
62    group by customer_name;
63
64    select * from restricted_sales_data;
65
```

Data Output Messages Notifications

```
ERROR:  permission denied for view restricted_sales_data

SQL state: 42501
```

Query   Query History

```
65
66      CREATE USER CLIENT1 WITH PASSWORD 'REPORT1234';
67      GRANT SELECT ON RESTRICTED_SALES_DATA TO CLIENT1;
68      REVOKE SELECT ON RESTRICTED_SALES_DATA FROM CLIENT;
```

Data Output   Messages   Notifications

GRANT

Query returned successfully in 154 msec.

Query   Query History

```
64      select * from restricted_sales_data;
65
66      CREATE USER CLIENT1 WITH PASSWORD 'REPORT1234';
67      GRANT SELECT ON RESTRICTED_SALES_DATA TO CLIENT1;
68      REVOKE SELECT ON RESTRICTED_SALES_DATA FROM CLIENT1;
```

Data Output   Messages   Notifications

REVOKE

Query returned successfully in 163 msec.

Query   Query History

```
63
64      select * from restricted_sales_data;
65      |
66      CREATE USER CLIENT1 WITH PASSWORD 'REPORT1234';
67      GRANT SELECT ON RESTRICTED_SALES_DATA TO CLIENT1;
```

Data Output   Messages   Notifications

ERROR: permission denied for view restricted_sales_data

SQL state: 42501