

## **Experiment 1.5**

**Student Name:** Vaibhav Kumar

**UID:** 23BCS13386

**Branch:** B.E-C.S.E

**Section/Group:** 23BCS\_KRG-2B

**Semester:** 5<sup>th</sup>

**Date of Performance:** 24 Sept, 2025

**Subject Name:** ADBMS

**Subject Code:** 23CSP-333

|                |
|----------------|
| MEDIUM - LEVEL |
|----------------|

1. **Problem Title:** Views: Performance Benchmarking : Normal View vs. Materialized View

2. **Procedure (Step-by-Step):**

1. 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.


2. Create a normal view and materialized view to for sales\_summary, which includes total\_quantity\_sold, total\_sales, and total\_orders with aggregation.


3. Compare the performance and execution time of both.

3. **SQL Commands:**

```
1 CREATE TABLE transaction_data (  
2     id INT,  
3     value INT  
4 );  
5  
6  
7 -- For id = 1  
8 INSERT INTO transaction_data (id, value)  
9 SELECT 1, random() * 100 -- simulate transaction amounts 0-1000  
10 FROM generate_series(1, 100);  
11  
12 -- For id = 2  
13 INSERT INTO transaction_data (id, value)  
14 SELECT 2, random() * 1000  
15 FROM generate_series(1, 100);  
16  
17 SELECT * FROM transaction_data;  
18  
19  
20  
21 --WITH NORMAL VIEW  
22 CREATE VIEW sales_summary_view AS  
23 SELECT  
24     id,  
25     COUNT(*) AS total_orders,  
26     SUM(value) AS total_sales,  
27     AVG(value) AS avg_transaction  
28 FROM transaction_data  
29 GROUP BY id;  
30  
31  
32 EXPLAIN ANALYZE  
33 SELECT * FROM sales_summary_view;  
34  
35  
36  
37 --WITH MATERIALIZED VIEW  
38 CREATE MATERIALIZED VIEW sales_summary_mv AS  
39 SELECT  
40     id,  
41     COUNT(*) AS total_orders,  
42     SUM(value) AS total_sales,  
43     AVG(value) AS avg_transaction  
44 FROM transaction_data  
45 GROUP BY id;  
46  
47  
48  
49 EXPLAIN ANALYZE  
50 SELECT * FROM sales_summary_mv;  
51  
52  
53 REFRESH MATERIALIZED VIEW sales_summary_mv;  
54
```

## 5. Output:

EXPERIMENT-5(MEDIUM) 


AI NEW PostgreSQL RUN 


STDIN

Input for the program ( Optional )

Output:

CREATE TABLE  
INSERT 0 100  
INSERT 0 100  
id | value  
-----+-----  
1 | 16  
1 | 23  
1 | 97  
1 | 44  
1 | 20  
1 | 45  
1 | 16  
1 | 15  
1 | 39  
1 | 60  
1 | 10  
1 | 91  
1 | 31  
1 | 21  
1 | 16  
1 | 14  
1 | 88  
1 | 98  
1 | 21

EXPERIMENT-5(MEDIUM) 

AI NEW PostgreSQL RUN 

STDIN

Input for the program ( Optional )

2 | 38  
2 | 332  
2 | 81  
2 | 729  
2 | 89  
2 | 829  
2 | 869  
2 | 190  
2 | 560  
2 | 519  
2 | 548  
2 | 129  
2 | 348  
2 | 572  
2 | 862  
2 | 390  
2 | 909  
2 | 42  
2 | 895  
2 | 400  
2 | 768  
2 | 256  
2 | 869  
2 | 801  
2 | 74  
(200 rows)

CREATE VIEW

QUERY PLAN

```
-----  
HashAggregate (cost=55.20..57.70 rows=200 width=52) (actual time=0.266..0.270 rows=2 loops=1)  
  Group Key: transaction_data.id  
  Batches: 1  Memory Usage: 40kB  
  -> Seq Scan on transaction_data (cost=0.00..32.60 rows=2260 width=8) (actual time=0.005..0.122 rows=200 loops=1)  
Planning Time: 0.368 ms  
Execution Time: 0.355 ms  
(6 rows)
```

SELECT 2

QUERY PLAN

```
-----  
Seq Scan on sales_summary_mv (cost=0.00..20.20 rows=1020 width=52) (actual time=0.003..0.005 rows=2 loops=1)  
Planning Time: 0.035 ms  
Execution Time: 0.013 ms  
(3 rows)
```

REFRESH MATERIALIZED VIEW

## HARD - LEVEL

### 1. Problem Title: Views: Securing Data Access with Views and Role-Based Permissions

### 2. Procedure (Step-by-Step):

The company **TechMart Solutions** stores all sales transactions in a central database.

A new reporting team has been formed to analyze sales but **they should not have direct access to the base tables** for security reasons. The database administrator has decided to:

1. Create **restricted views** to display only summarized, non-sensitive data.
2. Assign access to these views to specific users using **DCL commands** (GRANT, REVOKE).

### 3. SQL Commands:



commands.sql

EXPERIMENT-5(HARD)

```
1 CREATE TABLE customer_master (  
2     customer_id VARCHAR(5) PRIMARY KEY,  
3     full_name VARCHAR(50) NOT NULL,  
4     phone VARCHAR(15),  
5     email VARCHAR(50),  
6     city VARCHAR(30)  
7 );  
8  
9 CREATE TABLE product_catalog (  
10     product_id VARCHAR(5) PRIMARY KEY,  
11     product_name VARCHAR(50) NOT NULL,  
12     brand VARCHAR(30),  
13     unit_price NUMERIC(10,2) NOT NULL  
14 );  
15  
16 CREATE TABLE sales_orders (  
17     order_id SERIAL PRIMARY KEY,  
18     product_id VARCHAR(5) REFERENCES product_catalog(product_id),  
19     quantity INT NOT NULL,  
20     customer_id VARCHAR(5) REFERENCES customer_master(customer_id),  
21     discount_percent NUMERIC(5,2),  
22     order_date DATE NOT NULL  
23 );  
24  
25  
26  
27 INSERT INTO customer_master (customer_id, full_name, phone, email, city) VALUES  
28 ('C1', 'Amit Sharma', '9876543210', 'amit.sharma@example.com', 'Delhi'),  
29 ('C2', 'Priya Verma', '9876501234', 'priya.verma@example.com', 'Mumbai'),  
30 ('C3', 'Ravi Kumar', '9988776655', 'ravi.kumar@example.com', 'Bangalore'),  
31 ('C4', 'Neha Singh', '9123456789', 'neha.singh@example.com', 'Kolkata'),  
32 ('C5', 'Arjun Mehta', '9812345678', 'arjun.mehta@example.com', 'Hyderabad'),  
33 ('C6', 'Sneha Reddy', '9090909090', 'sneha.reddy@example.com', 'Chennai'),  
34 ('C7', 'Vikram Das', '9123412345', 'vikram.das@example.com', 'Pune'),  
35 ('C8', 'Rohit Gupta', '9000000001', 'rohit.gupta@example.com', 'Lucknow'),  
36 ('C9', 'Pooja Nair', '9898989898', 'pooja.nair@example.com', 'Kochi'),  
37 ('C10', 'Ankit Yadav', '9345678901', 'ankit.yadav@example.com', 'Ahmedabad');  
38  
39  
40  
41 INSERT INTO product_catalog (product_id, product_name, brand, unit_price) VALUES  
42 ('P1', 'Smartphone X100', 'Samsung', 25000.00),  
43 ('P2', 'Laptop Pro 15', 'Dell', 65000.00),  
44 ('P3', 'Wireless Earbuds', 'Sony', 5000.00),  
45 ('P4', 'Smartwatch Fit', 'Apple', 30000.00),  
46 ('P5', 'Tablet 10.5', 'Lenovo', 22000.00),  
47 ('P6', 'Gaming Console', 'Sony', 45000.00),  
48 ('P7', 'Bluetooth Speaker', 'JBL', 7000.00),  
49 ('P8', 'Digital Camera', 'Canon', 55000.00),  
50 ('P9', 'LED TV 55 inch', 'LG', 60000.00),  
51 ('P10', 'Power Bank 20000mAh', 'Mi', 2500.00);  
52  
53  
54  
55 INSERT INTO sales_orders (product_id, quantity, customer_id, discount_percent, order_date) VALUES  
56 ('P1', 2, 'C1', 5.00, '2025-09-01'),  
57 ('P2', 1, 'C2', 10.00, '2025-09-02'),  
58 ('P3', 3, 'C3', 0.00, '2025-09-03'),  
59 ('P4', 1, 'C4', 8.00, '2025-09-04'),  
60 ('P5', 2, 'C5', 5.00, '2025-09-05'),  
61 ('P6', 1, 'C1', 12.00, '2025-09-06'),  
62 ('P7', 2, 'C2', 0.00, '2025-09-07'),  
63 ('P8', 1, 'C3', 10.00, '2025-09-08'),  
64 ('P9', 1, 'C6', 15.00, '2025-09-09'),  
65 ('P10', 4, 'C7', 0.00, '2025-09-10'),  
66 ('P1', 1, 'C8', 5.00, '2025-09-11'),  
67 ('P2', 2, 'C9', 10.00, '2025-09-12'),  
68 ('P3', 2, 'C10', 0.00, '2025-09-13'),  
69 ('P4', 1, 'C5', 8.00, '2025-09-14'),  
70 ('P5', 3, 'C6', 5.00, '2025-09-15'),  
71 ('P6', 1, 'C7', 12.00, '2025-09-16'),  
72 ('P7', 2, 'C8', 0.00, '2025-09-17'),  
73 ('P8', 1, 'C9', 10.00, '2025-09-18'),  
74 ('P9', 1, 'C10', 15.00, '2025-09-19'),  
75 ('P10', 5, 'C4', 0.00, '2025-09-20');  
76  
77
```

```

76
77
78 SELECT * FROM customer_master;
79 SELECT * FROM product_catalog;
80 SELECT * FROM sales_orders;
81
82
83
84 CREATE VIEW vw_ORDER_SUMMARY
85 AS
86 SELECT
87     O.order_id,
88     O.order_date,
89     P.product_name,
90     C.full_name,
91     (P.unit_price * O.quantity) - ((P.unit_price * O.quantity) * O.discount_percent / 100) AS final_cost
92 FROM customer_master AS C
93 JOIN sales_orders AS O
94     ON O.customer_id = C.customer_id
95 JOIN product_catalog AS P
96     ON P.product_id = O.product_id;
97
98
99
100 CREATE ROLE CLIENT_ABC
101 LOGIN
102 PASSWORD '1234';
103
104 GRANT SELECT ON vw_ORDER_SUMMARY TO CLIENT_ABC;
105
106 REVOKE SELECT ON vw ORDER SUMMARY FROM CLIENT ABC;

```

## 4. Output:

AI

NEW

POSTGRESQL

RUN

STDIN

Input for the program ( Optional )

Output:

```

CREATE TABLE
CREATE TABLE
CREATE TABLE
INSERT 0 10
INSERT 0 10
INSERT 0 20

```

| customer_id | full_name   | phone      | email                   | city      |
|-------------|-------------|------------|-------------------------|-----------|
| C1          | Amit Sharma | 9876543210 | amit.sharma@example.com | Delhi     |
| C2          | Priya Verma | 9876501234 | priya.verma@example.com | Mumbai    |
| C3          | Ravi Kumar  | 9988776655 | ravi.kumar@example.com  | Bangalore |
| C4          | Neha Singh  | 9123456789 | neha.singh@example.com  | Kolkata   |
| C5          | Arjun Mehta | 9812345678 | arjun.mehta@example.com | Hyderabad |
| C6          | Sneha Reddy | 9090909090 | sneha.reddy@example.com | Chennai   |
| C7          | Vikram Das  | 9123412345 | vikram.das@example.com  | Pune      |
| C8          | Rohit Gupta | 9000000001 | rohit.gupta@example.com | Lucknow   |
| C9          | Pooja Nair  | 9898989898 | pooja.nair@example.com  | Kochi     |
| C10         | Ankit Yadav | 9345678901 | ankit.yadav@example.com | Ahmedabad |

(10 rows)

| product_id | product_name        | brand   | unit_price |
|------------|---------------------|---------|------------|
| P1         | Smartphone X100     | Samsung | 25000.00   |
| P2         | Laptop Pro 15       | Dell    | 65000.00   |
| P3         | Wireless Earbuds    | Sony    | 5000.00    |
| P4         | Smartwatch Fit      | Apple   | 30000.00   |
| P5         | Tablet 10.5         | Lenovo  | 22000.00   |
| P6         | Gaming Console      | Sony    | 45000.00   |
| P7         | Bluetooth Speaker   | JBL     | 7000.00    |
| P8         | Digital Camera      | Canon   | 55000.00   |
| P9         | LED TV 55 inch      | LG      | 60000.00   |
| P10        | Power Bank 20000mAh | Mi      | 2500.00    |

(10 rows)

| order_id | product_id | quantity | customer_id | discount_percent | order_date |
|----------|------------|----------|-------------|------------------|------------|
| 1        | P1         | 2        | C1          | 5.00             | 2025-09-01 |
| 2        | P2         | 1        | C2          | 10.00            | 2025-09-02 |
| 3        | P3         | 3        | C3          | 0.00             | 2025-09-03 |
| 4        | P4         | 1        | C4          | 8.00             | 2025-09-04 |
| 5        | P5         | 2        | C5          | 5.00             | 2025-09-05 |
| 6        | P6         | 1        | C1          | 12.00            | 2025-09-06 |
| 7        | P7         | 2        | C2          | 0.00             | 2025-09-07 |
| 8        | P8         | 1        | C3          | 10.00            | 2025-09-08 |
| 9        | P9         | 1        | C6          | 15.00            | 2025-09-09 |
| 10       | P10        | 4        | C7          | 0.00             | 2025-09-10 |
| 11       | P1         | 1        | C8          | 5.00             | 2025-09-11 |
| 12       | P2         | 2        | C9          | 10.00            | 2025-09-12 |
| 13       | P3         | 2        | C10         | 0.00             | 2025-09-13 |
| 14       | P4         | 1        | C5          | 8.00             | 2025-09-14 |
| 15       | P5         | 3        | C6          | 5.00             | 2025-09-15 |
| 16       | P6         | 1        | C7          | 12.00            | 2025-09-16 |
| 17       | P7         | 2        | C8          | 0.00             | 2025-09-17 |
| 18       | P8         | 1        | C9          | 10.00            | 2025-09-18 |
| 19       | P9         | 1        | C10         | 15.00            | 2025-09-19 |
| 20       | P10        | 5        | C4          | 0.00             | 2025-09-20 |

(20 rows)

CREATE VIEW