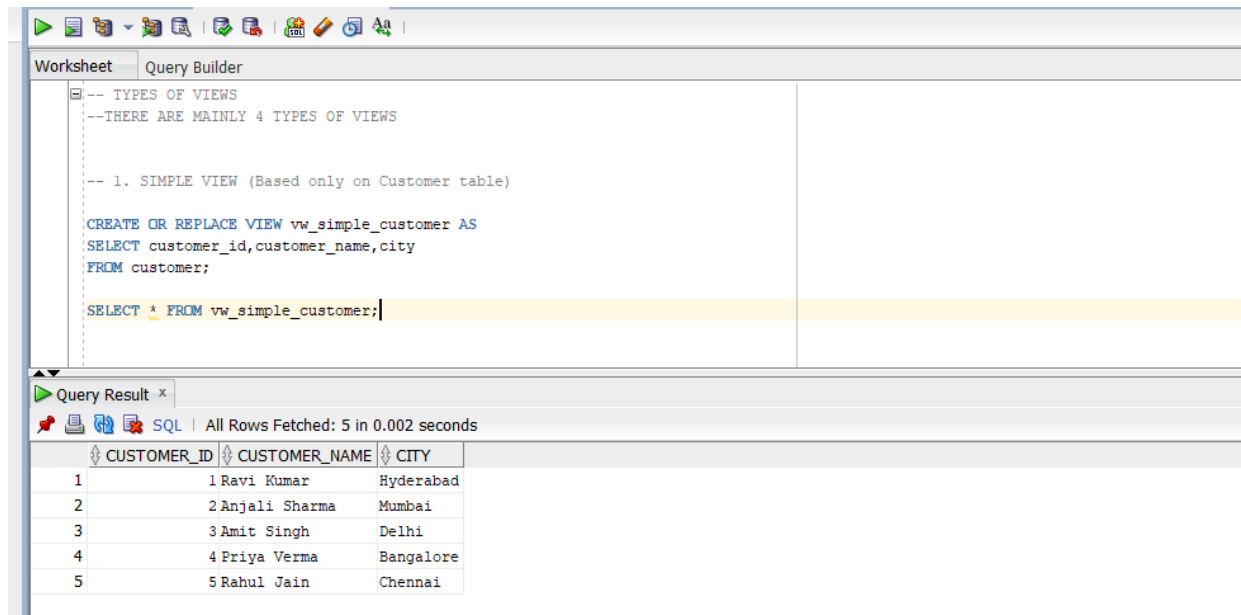


```
-- TYPES OF VIEWS
--THERE ARE MAINLY 4 TYPES OF VIEWS

-- 1. SIMPLE VIEW (Based only on Customer table)

CREATE OR REPLACE VIEW vw_simple_customer AS
SELECT customer_id,customer_name,city
FROM customer;

SELECT * FROM vw_simple_customer;
```



The screenshot shows a SQL query editor with a 'Query Builder' tab. The query text is as follows:

```
-- TYPES OF VIEWS
--THERE ARE MAINLY 4 TYPES OF VIEWS

-- 1. SIMPLE VIEW (Based only on Customer table)

CREATE OR REPLACE VIEW vw_simple_customer AS
SELECT customer_id,customer_name,city
FROM customer;

SELECT * FROM vw_simple_customer;
```

Below the query editor, the 'Query Result' tab is active, displaying the results of the executed query. The status bar indicates 'All Rows Fetched: 5 in 0.002 seconds'. The results are shown in a table with the following columns: CUSTOMER\_ID, CUSTOMER\_NAME, and CITY.

	CUSTOMER_ID	CUSTOMER_NAME	CITY
1	1	Ravi Kumar	Hyderabad
2	2	Anjali Sharma	Mumbai
3	3	Amit Singh	Delhi
4	4	Priya Verma	Bangalore
5	5	Rahul Jain	Chennai

```
-- 2. COMPLEX VIEW (Using JOIN + Aggregation)

CREATE OR REPLACE VIEW vw_complex_customer_orders AS
SELECT c.customer_id,c.customer_name,
COUNT(o.order_id) AS total_orders,
SUM(o.order_amount) AS total_purchase
FROM customer c
JOIN
orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.customer_name;

SELECT * FROM vw_complex_customer_orders;
```

Worksheet Query Builder

```
-- 2. COMPLEX VIEW (Using JOIN + Aggregation)

CREATE OR REPLACE VIEW vw_complex_customer_orders AS
SELECT c.customer_id, c.customer_name,
COUNT(o.order_id) AS total_orders,
SUM(o.order_amount) AS total_purchase
FROM customer c
JOIN
orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.customer_name;

SELECT * FROM vw_complex_customer_orders;
```

Query Result x

SQL | All Rows Fetched: 5 in 0.008 seconds

	CUSTOMER_ID	CUSTOMER_NAME	TOTAL_ORDERS	TOTAL_PURCHASE
1	3	Amit Singh	4	14000.5
2	5	Rahul Jain	4	9500.5
3	2	Anjali Sharma	4	9201.5
4	4	Priya Verma	4	3291
5	1	Ravi Kumar	4	12000.5

```
-- 3. MATERIALIZED VIEW
```

```
CREATE MATERIALIZED VIEW LOG ON CUSTOMERS;
```

```
CREATE MATERIALIZED VIEW mv_customer_sales TABLESPACE TBS_DATA
REFRESH FAST ON COMMIT
```

```
AS
```

```
SELECT c.customer_id, c.customer_name,
SUM(o.order_amount) AS total_sales
```

```
FROM customer c
```

```
JOIN orders o ON c.customer_id = o.customer_id
```

```
GROUP BY c.customer_id, c.customer_name;
```

```
-- 4. INLINE VIEW
```

```
SELECT * FROM (
```

```
SELECT c.customer_id, c.customer_name,
```



```
AVG(o.order_amount) AS avg_order_amount
```

```
FROM customer c JOIN
```

```
orders o ON c.customer_id = o.customer_id
```

```
GROUP BY c.customer_id, c.customer_name) inline_view
```

```
WHERE avg_order_amount > 500;
```

Worksheet		
Query Builder		
<pre> GROUP BY c.customer_id, c.customer_name;  -- 4. INLINE VIEW  SELECT * FROM ( SELECT c.customer_id, c.customer_name, AVG(o.order_amount) AS avg_order_amount FROM customer c JOIN orders o ON c.customer_id = o.customer_id GROUP BY c.customer_id, c.customer_name) inline_view WHERE avg_order_amount &gt; 500; </pre>		
Query Result x		
  All Rows Fetched: 5 in 0.01 seconds		
CUSTOMER_ID	CUSTOMER_NAME	AVG_ORDER_AMOUNT
1	3 Amit Singh	3500.125
2	5 Rahul Jain	2375.125
3	2 Anjali Sharma	2300.375
4	4 Priya Verma	822.75
5	1 Ravi Kumar	3000.125

```
-- COMPLEX VIEW WITH ADVANCED SELECT LOGIC
```

```

CREATE OR REPLACE VIEW vw_customer_order_analysis AS
SELECT c.customer_id, c.customer_name, c.city,
COUNT(o.order_id) AS total_orders,
SUM(o.order_amount) AS total_amount_spent,
AVG(o.order_amount) AS average_order_value,
CASE
WHEN SUM(o.order_amount) > 5000 THEN 'PLATINUM CUSTOMER'
WHEN SUM(o.order_amount) BETWEEN 2000 AND 5000 THEN 'GOLD CUSTOMER'
ELSE 'SILVER CUSTOMER'
END AS customer_category
FROM customer c
LEFT JOIN
orders o ON c.customer_id = o.customer_id
GROUP BY
c.customer_id, c.customer_name, c.city
HAVING
COUNT(o.order_id) > 0;

SELECT * FROM vw_customer_order_analysis;

```

Worksheet Query Builder

```
-- COMPLEX VIEW WITH ADVANCED SELECT LOGIC
CREATE OR REPLACE VIEW vw_customer_order_analysis AS
SELECT c.customer_id, c.customer_name, c.city,
COUNT(o.order_id) AS total_orders,
SUM(o.order_amount) AS total_amount_spent,
AVG(o.order_amount) AS average_order_value,
CASE
WHEN SUM(o.order_amount) > 5000 THEN 'PLATINUM CUSTOMER'
WHEN SUM(o.order_amount) BETWEEN 2000 AND 5000 THEN 'GOLD CUSTOMER'
ELSE 'SILVER CUSTOMER'
END AS customer_category
FROM customer c
LEFT JOIN
orders o ON c.customer_id = o.customer_id
GROUP BY
c.customer_id, c.customer_name, c.city
HAVING
COUNT(o.order_id) > 0;

SELECT * FROM vw_customer_order_analysis;
```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.009 seconds

	CUSTOMER_ID	CUSTOMER_NAME	CITY	TOTAL_ORDERS	TOTAL_AMOUNT_SPENT	AVERAGE_ORDER_VALUE	CUSTOMER_CATEGORY
1	5	Rahul Jain	Chennai	4	9500.5	2375.125	PLATINUM CUSTOMER
2	1	Ravi Kumar	Hyderabad	4	12000.5	3000.125	PLATINUM CUSTOMER
3	3	Amit Singh	Delhi	4	14000.5	3500.125	PLATINUM CUSTOMER
4	4	Priya Verma	Bangalore	4	3291	822.75	GOLD CUSTOMER
5	2	Anjali Sharma	Mumbai	4	9201.5	2300.375	PLATINUM CUSTOMER

-- SECURE VIEW (Hides sensitive data)

```
CREATE OR REPLACE VIEW vw_customer_public AS
SELECT customer_id, customer_name, city
FROM customer;
```

```
SELECT * FROM vw_customer_public;
```

```
-- SECURE VIEW (Hides sensitive data)

CREATE OR REPLACE VIEW vw_customer_public AS
SELECT customer_id, customer_name, city
FROM customer;

SELECT * FROM vw_customer_public;
```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.004 seconds

	CUSTOMER_ID	CUSTOMER_NAME	CITY
1	1	Ravi Kumar	Hyderabad
2	2	Anjali Sharma	Mumbai
3	3	Amit Singh	Delhi
4	4	Priya Verma	Bangalore
5	5	Rahul Jain	Chennai