

Task 7: Creating Views

Objective: Learn to create and use views

Tools: MySQL Workbench

Deliverables: View definitions and usage examples

Objectives:

1. Use CREATE VIEW with complex SELECT
2. Use views for abstraction and security

1. Creating Customers table

```
CREATE TABLE Customers (customer_id INT PRIMARY KEY, name VARCHAR(30), phone VARCHAR(10), city VARCHAR(20), age INT);
```

2. Creating Orders table

```
CREATE TABLE Orders (order_id INT PRIMARY KEY, customer_id INT, product VARCHAR(20), price FLOAT, FOREIGN KEY(customer_id) REFERENCES Customers(customer_id));
```

3. Inserting data into Customers

```
INSERT INTO Customers VALUES (1, 'Ravi Sharma', '9745834678', 'Mumbai', 21);
```

```
INSERT INTO Customers VALUES (2, 'Priya Verma', '8345587878', 'Delhi', 32);
```

```
INSERT INTO Customers VALUES (3, 'Amit Kumar', '9793258778', 'Pune', 25);
```

```
INSERT INTO Customers VALUES (4, 'Neha Singh', '9432885346', 'Chennai', 40);
```

4. Inserting data into Orders

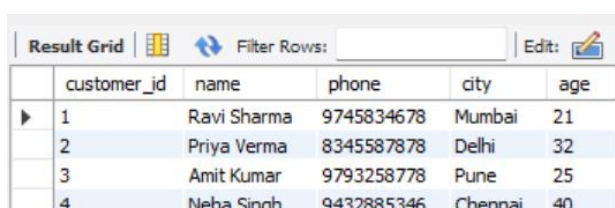
```
INSERT INTO Orders VALUES (101, 1, 'Laptop', 55000.00);
```

```
INSERT INTO Orders VALUES (102, 1, 'Keyboard', 1500.00);
```

```
INSERT INTO Orders VALUES (103, 2, 'Smartphone', 18000.00);
```

```
INSERT INTO Orders VALUES (104, 3, 'Tablet', 12000.00);
```



5. Displaying data from Customers table: SELECT * FROM Customers;



The screenshot shows the MySQL Workbench Result Grid interface. At the top, there are tabs for 'Result Grid', 'Table Grid', and 'Filter Rows'. The 'Result Grid' tab is active. Below the tabs, there is a table with 6 columns: 'customer_id', 'name', 'phone', 'city', and 'age'. The table contains 4 rows of data. The first row is highlighted in blue. The second row is highlighted in light blue. The third row is highlighted in light blue. The fourth row is highlighted in light blue.

	customer_id	name	phone	city	age
▶	1	Ravi Sharma	9745834678	Mumbai	21
	2	Priya Verma	8345587878	Delhi	32
	3	Amit Kumar	9793258778	Pune	25
	4	Neha Singh	9432885346	Chennai	40

6. Displaying data from Orders table: SELECT * FROM Orders;

Result Grid				Filter Rows:	
	order_id	customer_id	product	price	
▶	101	1	Laptop	55000	
	102	1	Keyboard	1500	
	103	2	Smartphone	18000	
	104	3	Tablet	12000	
✱	NULL	NULL	NULL	NULL	

7. Creating a simple view: View to display customer names with their orders and prices

```
CREATE VIEW CustomerOrders AS
```

```
SELECT C.customer_id, C.name, C.city, O.product, O.price
```

```
FROM Customers C INNER JOIN Orders O
```

```
ON C.customer_id = O.customer_id;
```

```
SELECT * FROM CustomerOrders;
```

Result Grid

Filter Rows:

Export:

	customer_id	name	city	product	price
▶	1	Ravi Sharma	Mumbai	Laptop	55000
	1	Ravi Sharma	Mumbai	Keyboard	1500
	2	Priya Verma	Delhi	Smartphone	18000
	3	Amit Kumar	Pune	Tablet	12000

8. Creating a Filtered View: View to show orders above 20000

```
CREATE VIEW HighValueOrders AS
```

```
SELECT C.name, O.product, O.price
```

```
FROM Customers C
```

```
INNER JOIN Orders O
```

```
ON C.customer_id = O.customer_id
```

```
WHERE O.price > 20000;
```

```
SELECT * FROM HighValueOrders;
```

Result Grid

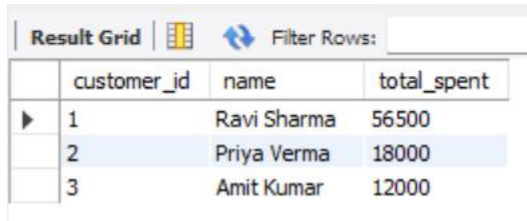
Filter Rows:

	customer_id	name	product	price
	1	Ravi Sharma	Laptop	55000

9. Creating an Aggregated View: View to show total spending per customer

```
CREATE VIEW TotalSpending AS SELECT C.customer_id, C.name, SUM(O.price) AS total_spent FROM Customers C INNER JOIN Orders O ON C.customer_id = O.customer_id GROUP BY C.customer_id, C.name;
```

```
SELECT * FROM TotalSpending;
```



The screenshot shows a database interface with a 'Result Grid' tab. It displays the results of a query, showing three columns: 'customer_id', 'name', and 'total_spent'. There are three rows of data. The first row shows customer_id 1, name Ravi Sharma, and total_spent 56500. The second row shows customer_id 2, name Priya Verma, and total_spent 18000. The third row shows customer_id 3, name Amit Kumar, and total_spent 12000. The second row is highlighted with a blue background.

	customer_id	name	total_spent
▶	1	Ravi Sharma	56500
	2	Priya Verma	18000
	3	Amit Kumar	12000

10. Dropping a View:

```
DROP VIEW IF EXISTS TotalSpending;
```