

E-COMMERCE

DATABASE



~ ATHARVA PAITL



ATHARVA PAITL

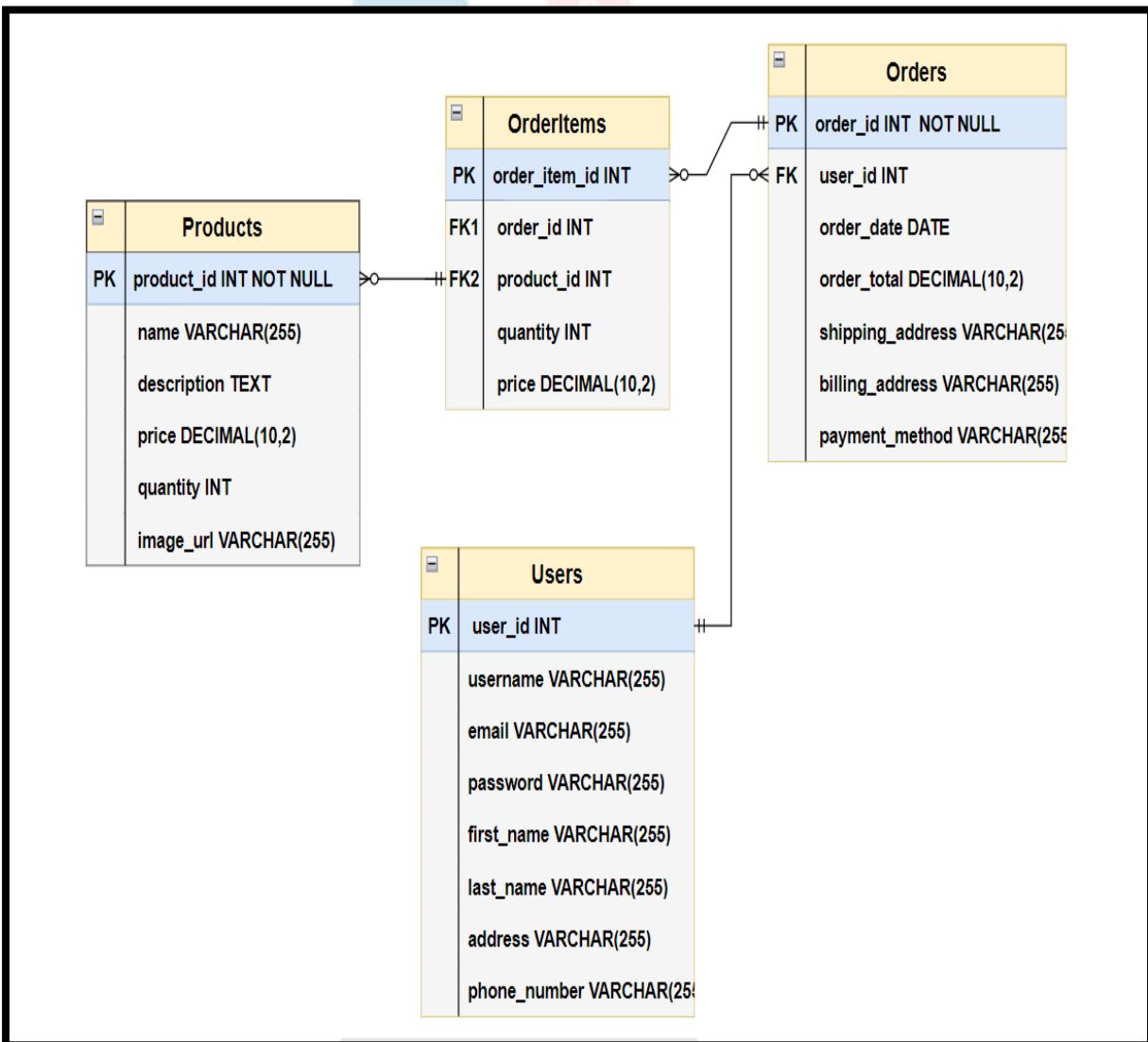
1. INTRODUCTION:

The goal of the e-commerce platform is to provide a seamless and secure online shopping experience for users. The system allows customers to browse, select, and purchase products, while also enabling administrators to manage inventory, user accounts, and track order fulfillment.

In this project, the information related to the e-commerce sales of various products is presented by using the relationships with each table to table. There are four tables in the database namely, products table, users table, orders table, and order items table.

These tables collectively provide a structured and organized way to store and retrieve data for an e-commerce system, facilitating efficient management of products, user information, orders, and the details of items within each order.

2. ER DIAGRAM:



3. DATABASE DESIGN:

Database: E_Com

Tables:

- Products
- Users
- Orders
- Orderitems



4. DATA DEFINITION LANGUAGE (DDL):

4.1 {CREATE}-

- CREATING DATABASE & TABLES

- a) Creating Database:

The screenshot shows the MySQL Workbench interface. In the SQL editor tab, the command `CREATE DATABASE E_COM;` is entered. In the Output tab, the results show a single row: `1 11:46:09 CREATE DATABASE E_COM` with the message `1 row(s) affected`.

#	Time	Action	Message
1	11:46:09	CREATE DATABASE E_COM	1 row(s) affected

- b) Using the Database:

The screenshot shows the MySQL Workbench interface. In the SQL editor tab, the command `use e_com;` is entered. In the Output tab, the results show a single row: `1 12:01:24 use e_com` with the message `0 row(s) affected`.

#	Time	Action	Message
1	12:01:24	use e_com	0 row(s) affected

- c) Creating Table (products):

The screenshot shows the MySQL Workbench interface. In the SQL editor tab, the command to create a table `CREATE TABLE Products (product_id INT PRIMARY KEY, name VARCHAR(255), description TEXT, price DECIMAL(10,2), quantity INT, image_url VARCHAR(255));` is entered. In the Output tab, the results show a single row: `1 12:07:43 CREATE TABLE Products (product_id INT PRIMARY KEY, name VARCHAR(255), description TEXT, price DECIMAL(10,2), quantity INT, image_url VARCHAR(255))` with the message `0 row(s) affected`.

#	Time	Action	Message
1	12:07:43	CREATE TABLE Products (product_id INT PRIMARY KEY, name VARCHAR(255), description TEXT, price DECIMAL(10,2), quantity INT, image_url VARCHAR(255))	0 row(s) affected

d) Creating Table (users):

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The code entered is:

```
1 • CREATE TABLE Users (
2     user_id INT PRIMARY KEY,
3     username VARCHAR(255),
4     email VARCHAR(255),
5     password VARCHAR(255),
6     first_name VARCHAR(255),
7     last_name VARCHAR(255),
8     address VARCHAR(255),
9     phone_number VARCHAR(255)
10 );
```

The output pane shows the execution results:

#	Time	Action	Message
1	12:09:35	CREATE TABLE Users (user_id INT PRIMARY KEY, username VARCHAR...)	0 row(s) affected

e) Creating Table (orders):

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The code entered is:

```
1 • CREATE TABLE Orders (
2     order_id INT PRIMARY KEY,
3     user_id INT,
4     FOREIGN KEY (user_id) REFERENCES Users(user_id),
5     order_date DATE,
6     order_total DECIMAL(10,2),
7     shipping_address VARCHAR(255),
8     billing_address VARCHAR(255),
9     payment_method VARCHAR(255)
10 );
```

The output pane shows the execution results:

#	Time	Action	Message
1	12:10:23	CREATE TABLE Orders (order_id INT PRIMARY KEY, user_id INT, FO...)	0 row(s) affected

f) Creating Table (orderItems):

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The code entered is:

```
1 • CREATE TABLE OrderItems (
2     order_item_id INT PRIMARY KEY,
3     order_id INT,
4     FOREIGN KEY (order_id) REFERENCES Orders(order_id),
5     product_id INT,
6     FOREIGN KEY (product_id) REFERENCES Products(product_id),
7     quantity INT,
8     price DECIMAL(10,2)
9 );
```

The output pane shows the execution results:

#	Time	Action	Message
1	12:11:55	CREATE TABLE OrderItems (order_item_id INT PRIMARY KEY, order_id I...)	0 row(s) affected

- TO CHECK STRUCTURE OF TABLE

a) Products-

The screenshot shows the MySQL Workbench interface with the following details:

Query Editor (Top):
1 DESC Products;
2

Result Grid (Bottom):

Field	Type	Null	Key	Default	Extra
product_id	int	int	PRI	NULL	
name	varchar(255)	YES		NULL	
description	text	YES		NULL	
price	decimal(10,2)	YES		NULL	
quantity	int	YES		NULL	

Action Output (Bottom):

#	Time	Action	Message
1	12:30:37	DESC Products	6 row(s) returned

b) Users-

The screenshot shows the MySQL Workbench interface with the following details:

Query Editor (Top):
1 DESC Users;
2

Result Grid (Bottom):

Field	Type	Null	Key	Default	Extra
user_id	int	NO	PRI	NULL	
username	varchar(255)	YES		NULL	
email	varchar(255)	YES		NULL	
password	varchar(255)	YES		NULL	

Action Output (Bottom):

#	Time	Action	Message
1	12:35:13	DESC Users	8 row(s) returned

c) Orders-

The screenshot shows the MySQL Workbench interface with the title bar "DESC Orders;". The main area displays the "Result Grid" showing the structure of the Orders table:

Field	Type	Null	Key	Default	Extra
order_id	int	NO	PRI	NULL	
user_id	int	YES	MUL	NULL	
order_date	date	YES		NULL	
order_total	decimal(10,2)	YES		NULL	
shipping_address	varchar(255)	YES		NULL	

Below the grid, the message "7 row(s) returned" is displayed.

d) OrderItems-

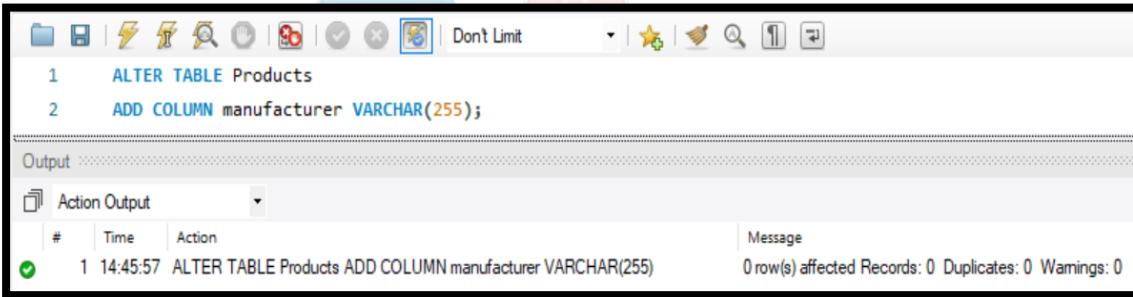
The screenshot shows the MySQL Workbench interface with the title bar "DESC OrderItems;". The main area displays the "Result Grid" showing the structure of the OrderItems table:

Field	Type	Null	Key	Default	Extra
order_item_id	int	NO	PRI	NULL	
order_id	int	YES	MUL	NULL	
product_id	int	YES	MUL	NULL	
quantity	int	YES		NULL	
price	decimal(10,2)	YES		NULL	

Below the grid, the message "5 row(s) returned" is displayed.

4.2 {ALTER}-

1. Alter Table Add Column



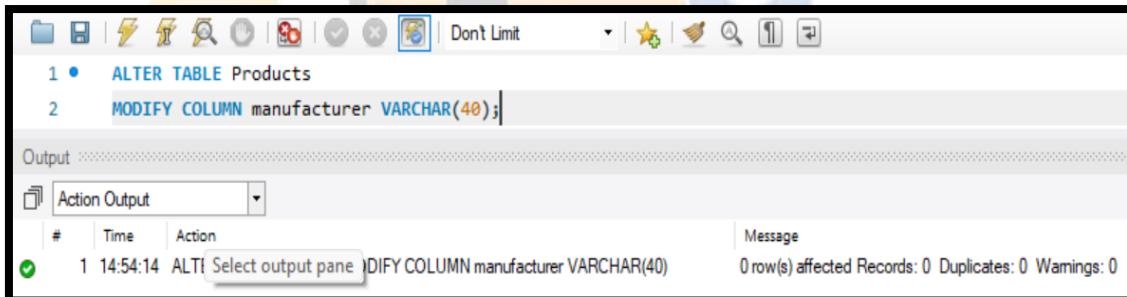
```
1 ALTER TABLE Products
2 ADD COLUMN manufacturer VARCHAR(255);
```

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:
1 ALTER TABLE Products
2 ADD COLUMN manufacturer VARCHAR(255);

The output pane shows the results of the execution:

#	Time	Action	Message
1	14:45:57	ALTER TABLE Products ADD COLUMN manufacturer VARCHAR(255)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

2. Alter Table Modify Column



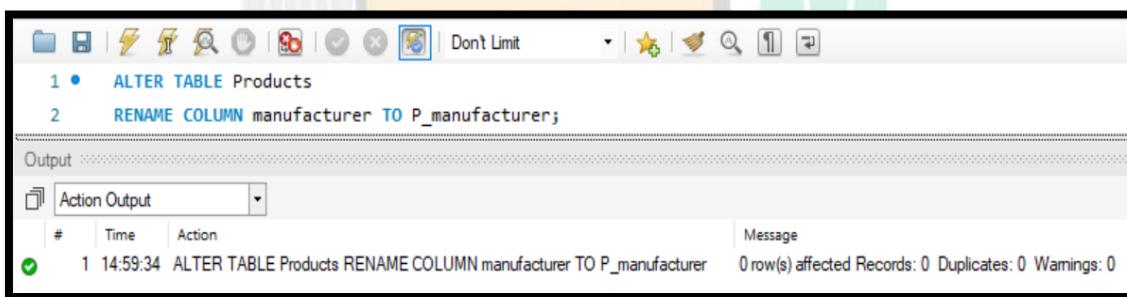
```
1 • ALTER TABLE Products
2 MODIFY COLUMN manufacturer VARCHAR(40);
```

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:
1 • ALTER TABLE Products
2 MODIFY COLUMN manufacturer VARCHAR(40);

The output pane shows the results of the execution:

#	Time	Action	Message
1	14:54:14	ALTER TABLE Products MODIFY COLUMN manufacturer VARCHAR(40)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

3. Alter Table Rename Column



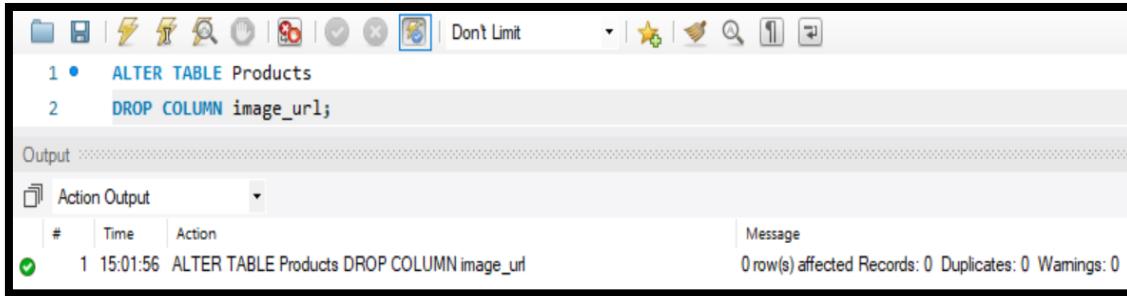
```
1 • ALTER TABLE Products
2 RENAME COLUMN manufacturer TO P_manufacturer;
```

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:
1 • ALTER TABLE Products
2 RENAME COLUMN manufacturer TO P_manufacturer;

The output pane shows the results of the execution:

#	Time	Action	Message
1	14:59:34	ALTER TABLE Products RENAME COLUMN manufacturer TO P_manufacturer	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

4. Alter Table Drop Column



```
1 • ALTER TABLE Products
2 DROP COLUMN image_url;
```

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:
1 • ALTER TABLE Products
2 DROP COLUMN image_url;

The output pane shows the results of the execution:

#	Time	Action	Message
1	15:01:56	ALTER TABLE Products DROP COLUMN image_url	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

5. Alter Column using Change

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following command is entered:

```
1 • ALTER TABLE Products
2     CHANGE Name P_name varchar(100);
```

In the Output pane, the results of the execution are shown:

#	Time	Action	Message
1	16:25:56	ALTER TABLE Products CHANGE Name P_name varchar(100)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

4.3 {DROP}-

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following command is entered:

```
1 • DROP TABLE sample_table;
```

In the Output pane, the results of the execution are shown:

#	Time	Action	Message
1	16:34:33	DROP TABLE sample_table	0 row(s) affected

4.4 {TRUNCATE}-

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following command is entered:

```
1 • TRUNCATE TABLE sample_table;
```

In the Output pane, the results of the execution are shown:

#	Time	Action	Message
1	16:38:39	TRUNCATE TABLE sample_table	0 row(s) affected

5. DATA MANIPULATION LANGUAGE (DML):

5.1 {INSERT INTO}:

The screenshot shows the MySQL Workbench interface with a query editor and an output pane. The query editor contains the following SQL code:

```
1 • INSERT INTO Products (product_id, name, description, price, quantity, image_url)
2 VALUES
3 (1, 'Laptop', 'High-performance laptop with the latest specs', 1200.00, 50, 'laptop_image.jpg'),
4 (2, 'Smartphone', 'Feature-rich smartphone with a sleek design', 800.00, 100, 'smartphone_image.jpg'),
5 (3, 'Tablet', 'Portable tablet for on-the-go productivity', 400.00, 30, 'tablet_image.jpg'),
6 (4, 'Smartwatch', 'Fitness and health tracking smartwatch', 150.00, 80, 'smartwatch_image.jpg'),
7 (5, 'Headphones', 'Noise-canceling wireless headphones', 100.00, 120, 'headphones_image.jpg'),
8 (6, 'Camera', 'Professional-grade digital camera', 1500.00, 20, 'camera_image.jpg'),
9 (7, 'Gaming Console', 'Powerful gaming console for immersive gameplay', 300.00, 40, 'console_image.jpg'),
10 (8, 'Printer', 'High-quality printer for home and office use', 200.00, 60, 'printer_image.jpg');
```

The output pane shows the results of the execution:

#	Time	Action	Message
1	17:11:55	INSERT INTO Products (product_id, name, description, price, quantity, image_url) VALUE...	8 row(s) affected Records: 8 Duplicates: 0 Warnings: 0

5.2 {UPDATE} With SET & WHERE Clause:

The screenshot shows the MySQL Workbench interface with a query editor and an output pane. The query editor contains the following SQL code:

```
1 • UPDATE Products
2 SET quantity = 60, price = 1300.00
3 WHERE product_id = 1;
```

The output pane shows the results of the execution:

#	Time	Action	Message
1	17:15:56	UPDATE Products SET quantity = 60, price = 1300.00 WHERE product_id = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0

5.3 {DELETE FROM} With WHERE Clause:

The screenshot shows the MySQL Workbench interface with a query editor and an output pane. The query editor contains the following SQL code:

```
1 • DELETE FROM Products
2 WHERE name = 'gaming console';
```

The output pane shows the results of the execution:

#	Time	Action	Message
1	17:19:47	DELETE FROM Products WHERE name = 'gaming console'	1 row(s) affected

6. DATA QUERY LANGUAGE (DQL):

6.1 {SELECT COMMAND}:

- To display all records in a table:

The screenshot shows the MySQL Workbench interface. In the SQL editor pane, the query `SELECT * FROM PRODUCTS;` is entered. Below the editor is the 'Result Grid' which displays the following data:

product_id	name	description	price	quantity	image_url
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
3	Tablet	Portable tablet for on-the-go productivity	400.00	30	tablet_image.jpg
4	Smartwatch	Fitness and health tracking smartwatch	150.00	80	smartwatch_image.jpg
5	Headphones	Noise-canceling wireless headphones	100.00	120	headphones_image.jpg
6	Camera	Professional-grade digital camera	1500.00	20	camera_image.jpg
8	Printer	High-quality printer for home and office use	200.00	60	printer_image.jpg

Below the grid, the message `7 row(s) returned` is displayed.

- SELECT command with WHERE clause:

The screenshot shows the MySQL Workbench interface. In the SQL editor pane, the query `SELECT * FROM PRODUCTS WHERE quantity > 50;` is entered. Below the editor is the 'Result Grid' which displays the following data:

product_id	name	description	price	quantity	image_url
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
4	Smartwatch	Fitness and health tracking smartwatch	150.00	80	smartwatch_image.jpg
5	Headphones	Noise-canceling wireless headphones	100.00	120	headphones_image.jpg
8	Printer	High-quality printer for home and office use	200.00	60	printer_image.jpg
*	NONE	NONE	NONE	NONE	NONE

Below the grid, the message `5 row(s) returned` is displayed.

6.2 {SHOW COMMAND}:

- To display all tables in a database:

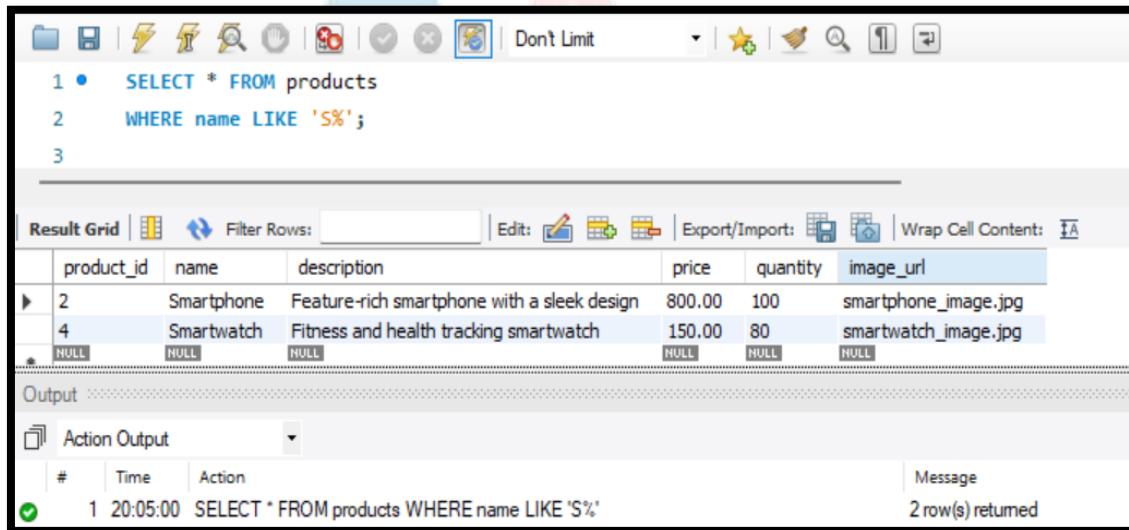
The screenshot shows the MySQL Workbench interface. In the SQL editor at the top, the command `SHOW TABLES;` is entered. Below it, the results are displayed in a grid titled "Tables_in_e_com". The grid lists four tables: `orderitems`, `orders`, `products`, and `users`. At the bottom of the interface, the "Action Output" section shows a log entry for the query: `1 12:24:09 SHOW TABLES` with a message indicating `4 row(s) returned`.

- To display databases:

The screenshot shows the MySQL Workbench interface. In the SQL editor at the top, the command `SHOW databases;` is entered. Below it, the results are displayed in a grid titled "Database". The grid lists ten databases: `college`, `e_com`, `information_schema`, `mysql`, `performance_schema`, `spotify`, and `sq7to9`. At the bottom of the interface, the "Action Output" section shows a log entry for the query: `1 19:54:15 SHOW databases` with a message indicating `10 row(s) returned`.

7. LIKE Query:

- Using '%':



The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1 •  SELECT * FROM products
2      WHERE name LIKE 'S%';
3
```

The Result Grid displays the following data:

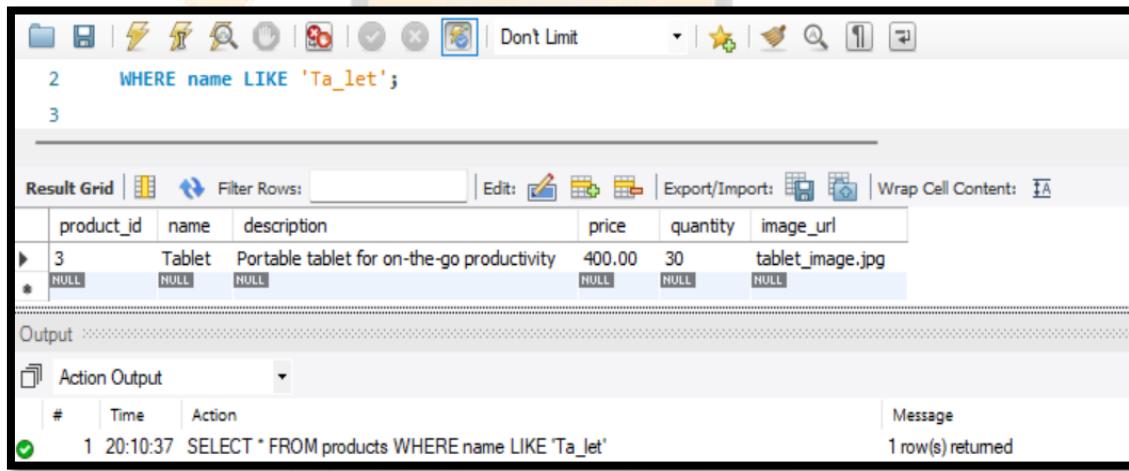
product_id	name	description	price	quantity	image_url
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
4	Smartwatch	Fitness and health tracking smartwatch	150.00	80	smartwatch_image.jpg
*	NULL	NULL	NULL	NULL	NULL

The Output pane shows the executed query and its result:

```
Output :>
Action Output
# Time Action
1 20:05:00 SELECT * FROM products WHERE name LIKE 'S%'
```

Message: 2 row(s) returned

- Using '_':



The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
2      WHERE name LIKE 'Ta_let';
3
```

The Result Grid displays the following data:

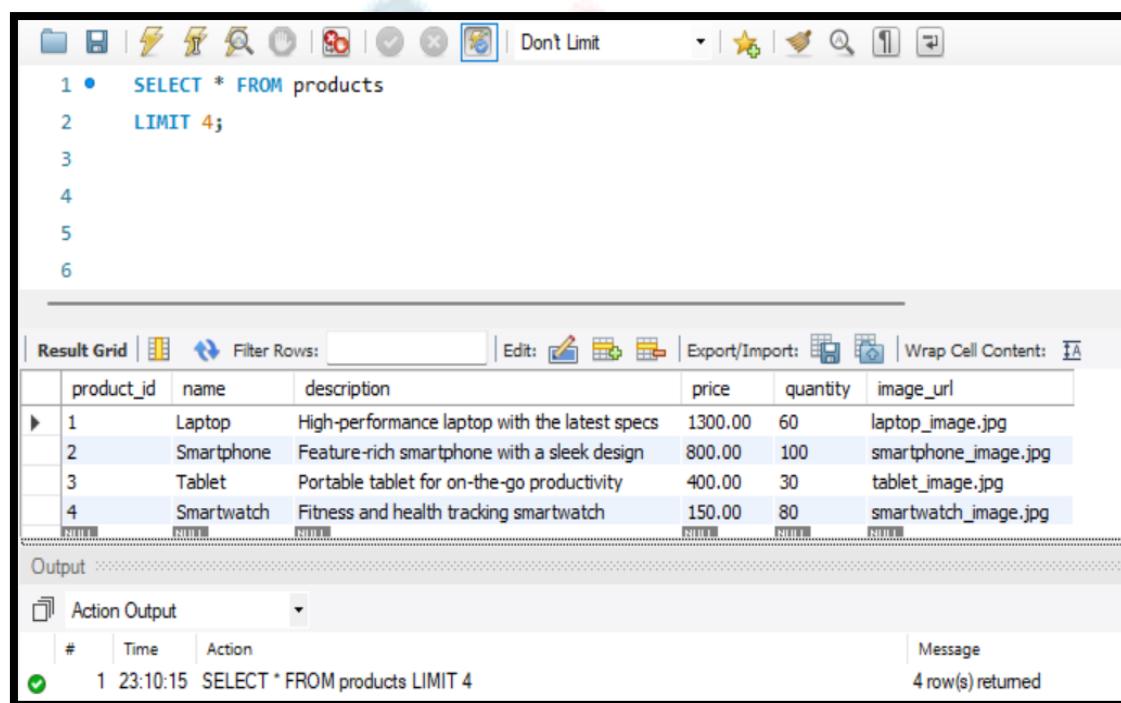
product_id	name	description	price	quantity	image_url
3	Tablet	Portable tablet for on-the-go productivity	400.00	30	tablet_image.jpg
*	NULL	NULL	NULL	NULL	NULL

The Output pane shows the executed query and its result:

```
Output :>
Action Output
# Time Action
1 20:10:37 SELECT * FROM products WHERE name LIKE 'Ta_let'
```

Message: 1 row(s) returned

8. LIMIT Clause:



The screenshot shows the MySQL Workbench interface. In the SQL editor tab, the following query is displayed:

```
1 •  SELECT * FROM products
2      LIMIT 4;
3
4
5
6
```

The result grid displays the following data:

product_id	name	description	price	quantity	image_url
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
3	Tablet	Portable tablet for on-the-go productivity	400.00	30	tablet_image.jpg
4	Smartwatch	Fitness and health tracking smartwatch	150.00	80	smartwatch_image.jpg

In the Output pane, the message "4 row(s) returned" is shown.

9. GROUP BY Clause:

a) GROUP BY:

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
1 •  SELECT user_id, SUM(order_total) AS total_order_amount
2   FROM orders
3   GROUP BY user_id;
```

The results grid displays the following data:

user_id	total_order_amount
1	2100.00
2	150.00
3	200.00
5	400.00
6	200.00

The output pane shows the query and its execution details:

Action Output

#	Time	Action	Message
1	13:20:58	SELECT user_id, SUM(order_total) AS total_order_amount FROM orders GROUP BY user_id	7 row(s) returned

b) GROUP BY with HAVING Clause:

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
1 •  SELECT user_id, COUNT(order_id) AS order_count, SUM(order_total) AS total_order_amount
2   FROM orders
3   GROUP BY user_id
4   HAVING total_order_amount > 100;
5
```

The results grid displays the following data:

user_id	order_count	total_order_amount
1	2	2100.00
2	1	150.00
3	1	200.00
5	1	400.00
6	1	200.00

The output pane shows the query and its execution details:

Action Output

#	Time	Action	Message
1	13:22:47	SELECT user_id, COUNT(order_id) AS order_count, SUM(order_total) AS total_order_amount F...	6 row(s) returned

10. ORDER BY Clause:

a) ORDER BY ASC:

The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query is:

```
1 • SELECT * FROM products
2 ORDER BY name ASC;
```

The result grid displays the following data:

product_id	name	description	price	quantity	image_url
6	Camera	Professional-grade digital camera	1500.00	20	camera_image.jpg
5	Headphones	Noise-canceling wireless headphones	100.00	120	headphones_image.jpg
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
8	Printer	High-quality printer for home and office use	200.00	60	printer_image.jpg
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
4	Smartwatch	Fitness and health tracking smartwatch	150.00	80	smartwatch_image.jpg

The output message at the bottom indicates 7 row(s) returned.

b) ORDER BY DESC:

The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query is:

```
1 • SELECT * FROM products
2 ORDER BY price DESC;
```

The result grid displays the following data:

product_id	name	description	price	quantity	image_url
6	Camera	Professional-grade digital camera	1500.00	20	camera_image.jpg
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
3	Tablet	Portable tablet for on-the-go productivity	400.00	30	tablet_image.jpg
8	Printer	High-quality printer for home and office use	200.00	60	printer_image.jpg

The output message at the bottom indicates 7 row(s) returned.

11. BUILT IN FUNCTIONS:

a) CONCAT:

The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query is:

```
1 •  SELECT CONCAT(name, ' ', price, ' ', quantity) AS Ordered_qty_with_price FROM products;
```

The result grid displays the following data:

Ordered_qty_with_price
Laptop 1300.00 60
Smartphone 800.00 100
Tablet 400.00 30
Smartwatch 150.00 80
Headphones 100.00 120

The message area at the bottom right indicates "7 row(s) returned".

b) LOWER:

The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query is:

```
1 •  SELECT product_id, LOWER(name) FROM products;
```

The result grid displays the following data:

product_id	LOWER(name)
1	laptop
2	smartphone
3	tablet
4	smartwatch
5	headphones

The message area at the bottom right indicates "7 row(s) returned".

c) UPPER:

The screenshot shows the MySQL Workbench interface with a query editor and result grid. The query is:

```
1 •  SELECT product_id, UPPER(name) FROM products;
```

The result grid displays the following data:

product_id	UPPER(name)
1	LAPTOP
2	SMARTPHONE
3	TABLET
4	SMARTWATCH
5	HEADPHONES

The output pane shows the query and its execution details:

```
Result 8 ×  
Output  
Action Output  
# Time Action Message  
1 23:31:57 SELECT product_id, UPPER(name) FROM products 7 row(s) returned
```

d) REPLACE:

The screenshot shows the MySQL Workbench interface with a query editor and result grid. The query is:

```
1 •  SELECT REPLACE('HELLO WORLD', 'HELLO', 'HOLA') AS REPLACED_STRING;
```

The result grid displays the following data:

REPLACED_STRING
HOLA WORLD

The output pane shows the query and its execution details:

```
Result 8 ×  
Output  
Action Output  
# Time Action Message  
1 23:38:48 SELECT REPLACE('HELLO WORLD', 'HELLO', 'HOLA') AS REPLACED_STRING 1 row(s) returned
```

e) REVERSE:

Screenshot of MySQL Workbench showing the results of a REVERSE function query. The query is:

```
1 • SELECT REVERSE(name) AS REVERSED_NAME FROM Products;
```

The Result Grid shows the following data:

REVERSED_NAME
potpal
enohptrams
tellbaT
htctawtramS
senohodaeH

The Action Output shows the query and its execution details:

#	Time	Action
1	23:40:54	SELECT REVERSE(name) AS REVERSED_NAME FROM Products

Message: 7 row(s) returned

f) LENGTH:

Screenshot of MySQL Workbench showing the results of a LENGTH function query. The query is:

```
1 • SELECT LENGTH(description) AS len_des FROM Products;
```

The Result Grid shows the following data:

len_des
45
43
42
38
35

The Action Output shows the query and its execution details:

#	Time	Action
1	23:42:29	SELECT LENGTH(description) AS len_des FROM Products

Message: 7 row(s) returned

g) SUBSTRING:

Screenshot of MySQL Workbench showing the results of a SUBSTR function query. The query is:

```
1 • SELECT SUBSTR(name,1,6) AS EXTRACTED_STRING FROM Products;
```

The Result Grid shows the following data:

EXTRACTED_STRING
Laptop
Smartp
Tablet
Smartw
Headph

The Action Output shows the query and its execution details:

#	Time	Action
1	23:44:16	SELECT SUBSTR(name,1,6) AS EXTRACTED_STRING FROM Products

Message: 7 row(s) returned

12. MATH() FUNCTIONS:

a) ABSOLUTE:

The screenshot shows the MySQL Workbench interface. The query editor contains the SQL statement: `SELECT ABS(-100);`. The results pane, titled "Output", shows the action "Action Output" with one row returned: `1 12:24:03 SELECT ABS(-100)`. The message indicates "1 row(s) returned".

b) MOD:

The screenshot shows the MySQL Workbench interface. The query editor contains the SQL statement: `SELECT MOD(20,3);`. The results pane, titled "Result Grid", displays the result: `MOD(20,3)`. The message indicates "1 row(s) returned".

c) FLOOR:

The screenshot shows the MySQL Workbench interface. The query editor contains the SQL statement: `SELECT FLOOR(19.56);`. The results pane, titled "Result Grid", displays the result: `FLOOR(19.56)`. The message indicates "1 row(s) returned".

d) CEILING:

The screenshot shows a MySQL Workbench interface. The query editor contains the SQL statement: `SELECT CEILING(19.56) AS ANSWER;`. The results grid shows one row with the column 'ANSWER' containing the value '20'. The action output pane shows a single log entry: '1 12:29:55 SELECT CEILING(19.56) AS ANSWER'. The message pane indicates '1 row(s) returned'.

e) EXPONENTIAL:

The screenshot shows a MySQL Workbench interface. The query editor contains the SQL statement: `SELECT EXP(6) AS ANSWER;`. The results grid shows one row with the column 'ANSWER' containing the value '403.4287934927351'. The action output pane shows a single log entry: '1 12:32:15 SELECT EXP(6) AS ANSWER'. The message pane indicates '1 row(s) returned'.

f) POWER:

The screenshot shows a MySQL Workbench interface. The query editor contains the SQL statement: `SELECT POWER(9,3) AS ANSWER;`. The results grid shows one row with the column 'ANSWER' containing the value '729'. The action output pane shows a single log entry: '1 12:33:47 SELECT POWER(9,3) AS ANSWER'. The message pane indicates '1 row(s) returned'.

g) SQUARE ROOT:

The screenshot shows a MySQL Workbench interface. The query editor contains the SQL statement: `SELECT SQRT(729) AS ANSWER;`. The results grid shows one row with the column 'ANSWER' containing the value '27'. The action output pane shows a single log entry: '1 12:35:31 SELECT SQRT(729) AS ANSWER'. The message pane indicates '1 row(s) returned'.

13. AGGREGATE FUNCTIONS:

a) COUNT:

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 •  SELECT COUNT(order_id) FROM ORDERS;
```

The result grid shows one row:

COUNT(order_id)
8

Below the result grid, the output pane shows the execution details:

#	Time	Action
1	19:12:35	SELECT COUNT(order_id) FROM ORDERS

Message: 1 row(s) returned

b) AVERAGE:

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 •  SELECT AVG(order_total) FROM ORDERS;
```

The result grid shows one row:

AVG(order_total)
581.250000

Below the result grid, the output pane shows the execution details:

#	Time	Action
1	19:15:48	SELECT AVG(order_total) FROM ORDERS

Message: 1 row(s) returned

c) SUM:

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 •  SELECT SUM(quantity) FROM PRODUCTS;
```

The result grid shows one row:

SUM(quantity)
470

Below the result grid, the output pane shows the execution details:

#	Time	Action
1	19:17:01	SELECT SUM(quantity) FROM PRODUCTS

Message: 1 row(s) returned

14. DATE () FUNCTIONS:

a) CURRDATE:

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 •  SELECT CURDATE() AS TODAYS_DATE;
```

The results grid shows one row:

TODAYS_DATE
2024-01-28

The output pane shows the action log:

#	Time	Action	Message
1	12:40:54	SELECT CURDATE() AS TODAYS_DATE	1 row(s) returned

b) NOW:

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 •  SELECT NOW();
```

The results grid shows one row:

NOW()
2024-01-28 12:42:12

The output pane shows the action log:

#	Time	Action	Message
1	12:42:12	SELECT NOW()	1 row(s) returned

c) LAST_DAY:

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
1 •  SELECT LAST_DAY('2024-2-12') AS ANSWER;
```

The results grid shows one row:

ANSWER
2024-02-29

The output pane shows the action log:

#	Time	Action	Message
1	12:43:22	SELECT LAST_DAY('2024-2-12') AS ANSWER	1 row(s) returned

d) DATE_FORMAT:

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
1 •  SELECT DATE_FORMAT(NOW(), '%b %d %Y %h:%i %p') AS ANSWER;
```

The results grid shows one row with the column 'ANSWER' containing 'Jan 28 2024 12:45 PM'. Below the results grid is an 'Output' section with an 'Action Output' table. The table has columns '#', 'Time', 'Action', and 'Message'. It contains one entry: '# 1 12:45:06 SELECT DATE_FORMAT(NOW(), "%b %d %Y %h:%i %p") AS ANSWER' with 'Message' '1 row(s) returned'.

e) DATEDIFF:

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
1 •  SELECT DATEDIFF('2023-04-12', '2023-03-11') AS DIFF;
```

The results grid shows one row with the column 'DIFF' containing '32'. Below the results grid is an 'Output' section with an 'Action Output' table. The table has columns '#', 'Time', 'Action', and 'Message'. It contains one entry: '# 1 12:46:09 SELECT DATEDIFF('2023-04-12', '2023-03-11') AS DIFF' with 'Message' '1 row(s) returned'.

15. UNION Operation:

The screenshot shows the MySQL Workbench interface with a query editor and results grid.

Query Editor:

```
1 • SELECT 'Product' AS data_source, product_id AS id, name, description, price, quantity, image_url AS additional_info
2   FROM Products
3
4 UNION
5   SELECT 'User' AS data_source, user_id AS id, username AS name, email AS description, NULL AS price, NULL AS quantity, phone_number AS additional_info
6   FROM Users;
```

Result Grid:

data_source	id	name	description	price	quantity	additional_info
Product	1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
Product	2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg
Product	3	Tablet	Portable tablet for on-the-go productivity	400.00	30	tablet_image.jpg
Product	4	Smartwatch	Fitness and health tracking smartwatch	150.00	80	smartwatch_image.jpg
Product	5	Headphones	Noise-canceling wireless headphones	100.00	120	headphones_image.jpg
Product	6	Camera	Professional-grade digital camera	1500.00	20	camera_image.jpg
Product	8	Printer	High-quality printer for home and office use	200.00	60	printer_image.jpg
User	1	alex_miller	alex.miller@example.com	NULL	NULL	555-1234-5678
User	2	soohia_davis	soohia.davis@example.com	NULL	NULL	555-9876-5432

Action Output:

#	Time	Action
1	19:37:05	SELECT 'Product' AS data_source, product_id AS id, name, description, price, quantity, image_url AS addti... 15 row(s) returned

The screenshot shows the MySQL Workbench interface with a query editor and results grid.

Query Editor:

```
1 • SELECT product_id, name, description, price
2   FROM Products
3
4 UNION
5   SELECT user_id, username, email, NULL AS price
6   FROM Users;
```

Result Grid:

product_id	name	description	price
1	Laptop	High-performance laptop with the latest specs	1300.00
2	Smartphone	Feature-rich smartphone with a sleek design	800.00
3	Tablet	Portable tablet for on-the-go productivity	400.00
4	Smartwatch	Fitness and health tracking smartwatch	150.00
5	Headphones	Noise-canceling wireless headphones	Fitness and health tracking smartwatch
6	Camera	Professional-grade digital camera	1500.00
8	Printer	High-quality printer for home and office use	200.00
1	alex_miller	alex.miller@example.com	NULL
2	soohia_davis	soohia.davis@example.com	NULL

Action Output:

#	Time	Action
1	19:40:09	SELECT product_id, name, description, price FROM Products UNION SELECT user_id, username, email, N... 15 row(s) returned

16. SUB-QUERY:

a) SINGLE ROW SUBQUERY:

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following query:

```
2 WHERE order_total > (SELECT AVG(order_total) FROM orders)
3 ORDER BY Order_id;
```

The Result Grid displays three rows of data from the 'orders' table:

order_id	user_id	order_date	order_total	shipping_address	billing_address	payment_method
1	1	2024-01-20	1300.00	123 Main St	123 Main St	Credit Card
4	7	2024-01-01	1500.00	890 Maple St	890 Maple St	Credit Card
7	1	2024-01-04	800.00	123 Main St	123 Main St	Cash on Delivery

The Output pane shows the execution log:

```
1 19:50:11 SELECT * FROM Orders WHERE order_total > (SELECT AVG(order_total) FROM orders) ORDER BY Order_id; 3 row(s) returned
```

b) MULTIPLE ROW SUBQUERY:

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following query:

```
1 • SELECT * FROM PRODUCTS
2 WHERE quantity IN (SELECT quantity FROM orders WHERE name = 'laptop' OR name = 'smartphone')
3 ORDER BY product_id;
```

The Result Grid displays two rows of data from the 'products' table:

product_id	name	description	price	quantity	image_url
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
2	Smartphone	Feature-rich smartphone with a sleek design	800.00	100	smartphone_image.jpg

The Output pane shows the execution log:

```
1 19:58:00 SELECT * FROM PRODUCTS WHERE quantity IN (SELECT quantity FROM orders WHERE name = 'laptop' OR name = 'smartphone') ORDER BY product_id; 2 row(s) returned
```

c) MULTIPLE COLUMN SUBQUERY:

The screenshot shows a MySQL Workbench interface with a query editor and results grid. The query is:

```
1 •  SELECT * FROM products
2 WHERE (name, price) IN (SELECT name,price FROM products WHERE quantity = 60)
3 ORDER BY product_id;
```

The results grid displays two rows of data from the 'products' table:

product_id	name	description	price	quantity	image_url
1	Laptop	High-performance laptop with the latest specs	1300.00	60	laptop_image.jpg
8	Printer	High-quality printer for home and office use	200.00	60	printer_image.jpg
*	NULL	NULL	NULL	NULL	NULL

The output pane shows the message: "2 row(s) returned".

d) MULTIPLE TABLE SUBQUERY:

The screenshot shows a MySQL Workbench interface with a query editor and results grid. The query is:

```
1 •  SELECT * FROM Users
2 WHERE user_id in (SELECT user_id FROM ORDERS WHERE payment_method = (SELECT payment_method FROM orders WHERE order_total = 1300))
3 ORDER BY user_id;
```

The results grid displays four rows of data from the 'Users' table:

user_id	username	email	password	first_name	last_name	address	phone_number
1	alex_miller	alex.miller@example.com	alex_pass123	Alex	Miller	123 Main St	555-1234-5678
6	eva_roberts	eva.roberts@example.com	eva_pass	Eva	Roberts	567 Cedar St	555-4444-9999
7	mason_martin	mason.martin@example.com	mason123	Mason	Martin	890 Maple St	555-6666-3333
8	ava_brown	ava.brown@example.com	ava_secure	Ava	Brown	765 Oak St	555-3333-7777
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The output pane shows the message: "4 row(s) returned".

17.JOINS:

a) INNER JOIN:

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
4 •  SELECT u.user_id, username, email, order_id, order_total
5   FROM Users U
6   INNER JOIN Orders O
7     ON U.user_id = O.user_id;
```

The results grid displays the following data:

	user_id	username	email	order_id	order_total
▶	1	alex_miller	alex.miller@example.com	1	1300.00
	3	carter_wilson	carter.wilson@example.com	2	200.00
	5	noah_taylor	noah.taylor@example.com	3	400.00
	7	mason_martin	mason.martin@example.com	4	1500.00
	6	eva roberts	eva.roberts@example.com	5	200.00

The output pane shows the query and its execution details:

```
Result 102 ×
Output:
Action Output
# Time Action
1 20:47:11 SELECT u.user_id,username,email,order_id,order_total FROM Users U INNER JOIN Orders O ON U.use... 8 row(s) returned
```

b) LEFT JOIN:

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
4 •  SELECT u.user_id, username, email, order_id, order_total
5   FROM Users U
6   LEFT JOIN Orders O
7     ON U.user_id = O.user_id;
```

The results grid displays the following data:

	user_id	username	email	order_id	order_total
▶	1	alex_miller	alex.miller@example.com	1	1300.00
	1	alex_miller	alex.miller@example.com	7	800.00
	2	sophia_davis	sophia.davis@example.com	6	150.00
	3	carter_wilson	carter.wilson@example.com	2	200.00
	4	lily_jackson	lily.jackson@example.com	NULL	NULL

The output pane shows the query and its execution details:

```
Result 104 ×
Output:
Action Output
# Time Action
1 20:52:10 SELECT u.user_id,username,email,order_id,order_total FROM Users U LEFT JOIN Orders O ON U.user... 9 row(s) returned
```

c) RIGHT JOIN:

The screenshot shows the MySQL Workbench interface with a query editor and result grid. The query is:

```
4 •  SELECT u.user_id, username, email, order_id, order_total
5   FROM Users U
6   RIGHT JOIN Orders O
7     ON U.user_id = O.user_id;
```

The result grid displays the following data:

	user_id	username	email	order_id	order_total
▶	1	alex_miller	alex.miller@example.com	1	1300.00
	3	carter_wilson	carter.wilson@example.com	2	200.00
	5	noah_taylor	noah.taylor@example.com	3	400.00
	7	mason_martin	mason.martin@example.com	4	1500.00
	6	eva_roberts	eva.roberts@example.com	5	200.00

Below the grid, the message "Result 105" is shown. In the output pane, there is one entry:

#	Time	Action	Message
1	20:53:04	SELECT u.user_id, username, email, order_id, order_total FROM Users U RIGHT JOIN Orders O ON U.user...	8 row(s) returned

d) FULL JOIN:

The screenshot shows the MySQL Workbench interface with a query editor and result grid. The query is:

```
1 •  SELECT u.user_id, username, email, order_id, order_total FROM Users U
2   RIGHT JOIN Orders O
3     ON U.user_id = O.user_id
4   UNION
5   SELECT u.user_id, username, email, order_id, order_total FROM Users U
6   RIGHT JOIN Orders O
7     ON U.user_id = O.user_id;
```

The result grid displays the following data:

	user_id	username	email	order_id	order_total
▶	1	alex_miller	alex.miller@example.com	1	1300.00
	3	carter_wilson	carter.wilson@example.com	2	200.00
	5	noah_taylor	noah.taylor@example.com	3	400.00
	7	mason_martin	mason.martin@example.com	4	1500.00
	6	eva_roberts	eva.roberts@example.com	5	200.00
	2	sophia_davis	sophia.davis@example.com	6	150.00
	1	alex_miller	alex.miller@example.com	7	800.00
	8	ava_brown	ava.brown@example.com	8	100.00

Below the grid, the message "Result 107" is shown. In the output pane, there is one entry:

#	Time	Action	Message
1	20:56:46	SELECT u.user_id, username, email, order_id, order_total FROM Users U RIGHT JOIN Orders O ON U.user...	8 row(s) returned

18. VIWES:

The screenshot shows the MySQL Workbench interface. In the top-left pane, there is a code editor with the following SQL statement:

```
1 • CREATE VIEW p_view AS
2     SELECT product_id, name, price, quantity
3     FROM products;
4
```

In the bottom-right pane, there is a log entry:

#	Time	Action	Message
1	12:52:45	CREATE VIEW p_view AS SELECT product_id, name, price, quantity FROM products	0 row(s) affected

The screenshot shows the MySQL Workbench interface. In the top-left pane, there is a code editor with the following SQL statement:

```
1 • SELECT * FROM p_view;
```

In the middle section, there is a result grid displaying the data from the view:

	product_id	name	price	quantity
1	1	Laptop	1300.00	60
2	2	Smartphone	800.00	100
3	3	Tablet	400.00	30
4	4	Smartwatch	150.00	80
5	5	Headphones	100.00	120

In the bottom-right pane, there is a log entry:

#	Time	Action	Message
1	12:53:55	SELECT * FROM p_view	7 row(s) returned