

## FINDED THE NUMBER OF PERSONS IN EACH COUNTRY

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'entri\_dd41' database schema with tables, views, stored procedures, and functions. The main editor window contains the following SQL query:

```
-- Finded the number of persons in each country
SELECT Country_name, COUNT(*) AS Number_of_Persons
FROM Persons GROUP BY Country_name;
```

The 'Result Grid' at the bottom displays the query results:

Country_name	Number_of_Persons
India	3
Syria	3
Pakistan	1
Poland	2
England	1
Japan	2

The 'Action Output' pane at the bottom shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	08:58:49	USE entri_dd41	0 row(s) affected	0.000 sec
2	09:00:56	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec

## FINDED NUMBER OF PERSONS IN EACH COUNTRY SORTED FROM HIGH TO LOW

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'entri\_dd41' database schema. The main editor window contains the following SQL query:

```
-- Finded the number of persons in each country
SELECT Country_name, COUNT(*) AS Number_of_Persons
FROM Persons GROUP BY Country_name;

-- Find the number of persons in each country started from high to low
SELECT Country_name, COUNT(*) AS Number_of_Persons
FROM Persons GROUP BY Country_name ORDER BY Number_of_Persons DESC;
```

The 'Result Grid' at the bottom displays the query results:

Country_name	Number_of_Persons
India	3
Syria	3
Poland	2
Japan	2
Pakistan	1
England	1

The 'Action Output' pane at the bottom shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	08:58:49	USE entri_dd41	0 row(s) affected	0.000 sec
2	09:00:56	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec
3	09:02:17	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec

## FINDED AVERAGE RATING FOR PERSON IN RESPECTIVE COUNTRIES AVERAGE GREATER THAN 3.0

The screenshot shows MySQL Workbench with the following SQL queries and results:

```
4 FROM Persons GROUP BY Country_name;
5
6 -- Find the number of persons in each country started from high to low
7
8 • SELECT Country_name, COUNT(*) AS Number_of_Persons
9 FROM Persons GROUP BY Country_name ORDER BY Number_of_Persons DESC;
10
11 -- Finded average rating for persons in respective countries if the average is greater than 3.0
12
13 • SELECT Country_name, AVG(Rating) AS Average_Rating
14 FROM Persons GROUP BY Country_name HAVING Average_Rating > 3.0;
```

**Result Grid:**

Country_name	Average_Rating
India	4.733333
Syria	4.333333
Pakistan	3.800000
Poland	3.250000
Japan	4.200000

**Output:**

#	Time	Action	Message	Duration / Fetch
1	08:58:49	USE entri_dd41	0 row(s) affected	0.000 sec
2	09:00:56	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec
3	09:02:17	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec
4	09:03:17	SELECT Country_name, AVG(Rating) AS Average_Rating FROM Persons GROUP BY Country_name ...	5 row(s) returned	0.000 sec / 0.000 sec

## FINDED COUNTRIES WITH SAME RATING AS THE USA

The screenshot shows MySQL Workbench with the following SQL queries and results:

```
8 • SELECT Country_name, COUNT(*) AS Number_of_Persons
9 FROM Persons GROUP BY Country_name ORDER BY Number_of_Persons DESC;
10
11 -- Finded average rating for persons in respective countries if the average is greater than 3.0
12
13 • SELECT Country_name, AVG(Rating) AS Average_Rating
14 FROM Persons GROUP BY Country_name HAVING Average_Rating > 3.0;
15
16 -- Finded the Countries with the Same Rating as the USA (Using Subqueries)
17
18 • SELECT DISTINCT Country_name FROM Persons WHERE Rating = (SELECT AVG(Rating) FROM Persons WHERE Country_name = 'USA');
```

**Result Grid:**

Country_name
--------------

**Output:**

#	Time	Action	Message	Duration / Fetch
1	09:08:28	USE entri_dd41	0 row(s) affected	0.000 sec
2	09:08:34	SELECT DISTINCT Country_name FROM Persons WHERE Rating = (SELECT AVG(Rating) FROM Persons WHERE Country_name = 'USA');	0 row(s) returned	0.015 sec / 0.000 sec

# SELECTED ALL COUNTRIES WHOSE POPULATION IS GREATER THAN AVERAGE POPULATION OF ALL NATION

The screenshot displays the MySQL Workbench interface. The main editor window contains the following SQL queries:

```
8 • SELECT Country_name, COUNT(*) AS Number_of_Persons
9 FROM Persons GROUP BY Country_name ORDER BY Number_of_Persons DESC;
10
11 -- Finded average rating for persons in respective countries if the average is greater than 3.0
12
13 • SELECT Country_name, AVG(Rating) AS Average_Rating
14 FROM Persons GROUP BY Country_name HAVING Average_Rating > 3.0;
15
16 -- Selected All Countries Whose Population Is Greater Than the Average Population of All Nations
17
18 • SELECT * FROM Country WHERE Population > (SELECT AVG(Population) FROM Country);
```

The Result Grid shows the following data:

	Id	Country_name	Population	Area
▶	3	Pakistan	7000000	150000
▶	5	England	6000000	120000
▶	6	France	8000000	300000
▶	7	Italy	9000000	250000

The Action Output window shows the execution of the queries:

#	Time	Action	Message	Duration / Fetch
1	08:58:49	USE entri_dd41	0 row(s) affected	0.000 sec
2	09:00:56	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec
3	09:02:17	SELECT Country_name, COUNT(*) AS Number_of_Persons FROM Persons GROUP BY Country_name...	6 row(s) returned	0.000 sec / 0.000 sec
4	09:03:17	SELECT Country_name, AVG(Rating) AS Average_Rating FROM Persons GROUP BY Country_name ...	5 row(s) returned	0.000 sec / 0.000 sec
5	09:03:57	SELECT * FROM Country WHERE Population > (SELECT AVG(Population) FROM Country) LIMIT 0, 1...	4 row(s) returned	0.015 sec / 0.000 sec

## CREATED DATABASE AND USED DATABASE

The screenshot shows the MySQL Workbench interface with a SQL script in the editor. The script includes queries to select data from the 'Persons' table, create a 'Product' database, and use it. The output window shows the execution results of these queries.

```
8 • SELECT Country_name, COUNT(*) AS Number_of_Persons
9 FROM Persons GROUP BY Country_name ORDER BY Number_of_Persons DESC;
10
11 -- Finded average rating for persons in respective countries if the average is greater than 3.0
12
13 • SELECT Country_name, AVG(Rating) AS Average_Rating
14 FROM Persons GROUP BY Country_name HAVING Average_Rating > 3.0;
15
16 -- Finded the Countries with the Same Rating as the USA (Using Subqueries)
17
18 • SELECT DISTINCT Country_name FROM Persons WHERE Rating = (SELECT AVG(Rating) FROM Persons WHERE Country_name = 'USA');
19
20 -- Selected All Countries Whose Population Is Greater Than the Average Population of All Nations
21
22 • SELECT * FROM Country WHERE Population > (SELECT AVG(Population) FROM Country);
23
24 -----
25
26 -- Created Product database
27 • CREATE DATABASE Product;
28
29 -- Used the Product database
30
31 • USE Product;
32
```

**Action Output**

#	Time	Action	Message	Duration / Fetch
1	09:08:28	USE entri_dd41	0 row(s) affected	0.000 sec
2	09:08:34	SELECT DISTINCT Country_name FROM Persons WHERE Rating = (SELECT AVG(Rating) FROM P...	0 row(s) returned	0.015 sec / 0.000 sec
3	09:09:59	USE Product	0 row(s) affected	0.000 sec

## CUSTOMER TABLE CREATED

The screenshot shows the MySQL Workbench interface with a SQL script in the editor. The script creates a 'customer' table with various columns and constraints. The output window shows the execution results of these queries.

```
36 First_name VARCHAR(50),
37 Last_name VARCHAR(50),
38 Email VARCHAR(100),
39 Phone_no VARCHAR(15),
40 Address VARCHAR(100),
41 City VARCHAR(50),
42 State VARCHAR(50),
43 Zip_code VARCHAR(10),
44 Country VARCHAR(50)
45 );
46 • desc customer;
```

**Result Grid**

Field	Type	Null	Key	Default	Extra
Customer_id	int	NO	PRI		
First_name	varchar(50)	YES			
Last_name	varchar(50)	YES			
Email	varchar(100)	YES			
Phone_no	varchar(15)	YES			
Address	varchar(100)	YES			
City	varchar(50)	YES			
State	varchar(50)	YES			
Zip_code	varchar(10)	YES			
Country	varchar(50)	YES			

**Table: customer**

**Columns:**

- Customer\_id int PK
- First\_name varchar(50)
- Last\_name varchar(50)
- Email varchar(100)
- Phone\_no varchar(15)
- Address varchar(100)
- City varchar(50)
- State varchar(50)
- Zip\_code varchar(10)
- Country varchar(50)

**Action Output**

#	Time	Action	Message	Duration / Fetch
1	21:10:00	CREATE DATABASE Product	1 row(s) affected	0.031 sec
2	21:10:39	USE Product	0 row(s) affected	0.000 sec
3	21:10:49	CREATE TABLE Customer ( Customer_id INT PRIMARY KEY, First_name VARCHAR(50), Las...	0 row(s) affected	0.046 sec
4	21:10:58	desc customer	10 row(s) returned	0.000 sec / 0.000 sec



## ADDED FIELDS TO CUSTOMER TABLE

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the 'customer' table structure with columns: Customer\_id, First\_name, Last\_name, Email, Phone\_no, Address, City, State, Zip\_code, and Country. The 'Table: customer' section lists these columns with their data types and constraints.

The main SQL editor shows a query: `select * from customer;`. The 'Result Grid' displays the output of this query, showing 10 rows of customer data. The 'Action Output' pane at the bottom shows the execution of the query, indicating that 10 rows were returned.

Customer_id	First_name	Last_name	Email	Phone_no	Address	City	State	Zip_code	Country
1	John	Doe	john.doe@example.com	123-456-7890	123 Elm St	Los Angeles	CA	90001	USA
2	Jane	Smith	jane.smith@example.com	234-567-8901	456 Oak St	New York	NY	10001	USA
3	Alice	Johnson	alice.johnson@example.com	345-678-9012	789 Pine St	Chicago	IL	60007	USA
4	Bob	Williams	bob.williams@example.com	456-789-0123	321 Maple St	Houston	TX	77001	USA
5	Charlie	Brown	charlie.brown@example.com	567-890-1234	654 Cedar St	Phoenix	AZ	85001	USA
6	Emily	Davis	emily.davis@example.com	678-901-2345	987 Spruce St	Philadelphia	PA	19019	USA
7	Frank	Garcia	frank.garcia@example.com	789-012-3456	159 Birch St	San Antonio	TX	78201	USA
8	Grace	Martinez	grace.martinez@example.com	890-123-4567	753 Willow St	San Diego	CA	92101	USA
9	Hannah	Lopez	hannah.lopez@example.com	901-234-5678	832 Palm St	Dallas	TX	75201	USA
10	Isaac	Wilson	isaac.wilson@example.com	012-345-6789	963 Fir St	Austin	TX	73301	USA

## CREATED CUSTOMER INFO FOR CUSTOMER TABLE THAT DISPLAY CUSTOMER'S FULL NAME AND EMAIL ADDRESS

The screenshot shows the MySQL Workbench interface. The 'SCHEMAS' pane on the left shows the 'customer\_info' view structure with columns: Full\_name and Email. The main SQL editor shows the following SQL code:

```
-- Created a view named customer_info for the Customer table that displays Customer's Full name and email address
CREATE VIEW customer_info AS
SELECT CONCAT(First_name, ' ', Last_name) AS Full_name, Email FROM Customer;
-- Performed SELECT operation for customer_info view
SELECT * FROM customer_info;
```

The 'Result Grid' displays the output of the `SELECT * FROM customer_info;` query, showing 10 rows of customer data with the full name and email address. The 'Action Output' pane at the bottom shows the execution of the view creation and the subsequent query, indicating that 10 rows were returned.

Full_name	Email
John Doe	john.doe@example.com
Jane Smith	jane.smith@example.com
Alice Johnson	alice.johnson@example.com
Bob Williams	bob.williams@example.com
Charlie Brown	charlie.brown@example.com
Emily Davis	emily.davis@example.com
Frank Garcia	frank.garcia@example.com
Grace Martinez	grace.martinez@example.com
Hannah Lopez	hannah.lopez@example.com
Isaac Wilson	isaac.wilson@example.com

## CREATED VIEW NAMED US\_CUSTOMER THAT DISPLAYS CUSTOMER LOCATED IN THE US

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

```
64
65 CREATE VIEW customer_info AS
66 SELECT CONCAT(First_name, ' ', Last_name) AS Full_name, Email FROM Customer;
67
68 -- Performed SELECT operation for customer_info view
69 SELECT * FROM customer_info;
70
71 -- Created a view named US_Customers that displays customers located in the US
72 CREATE VIEW US_Customers AS
73 SELECT * FROM Customer WHERE Country = 'USA';
74 select * from us_customers;
```

The Result Grid shows the output of the queries:

Customer_Id	First_name	Last_name	Email	Phone_no	Address	City	State	Zip_code	Country
1	John	Doe	john.doe@example.com	123-456-7890	123 Elm St	Los Angeles	CA	90001	USA
2	Jane	Smith	jane.smith@example.com	234-567-8901	456 Oak St	New York	NY	10001	USA
3	Alice	Johnson	alice.johnson@example.com	345-678-9012	789 Pine St	Chicago	IL	60007	USA
4	Bob	Williams	bob.williams@example.com	456-789-0123	321 Maple St	Houston	TX	77001	USA
5	Charlie	Brown	charlie.brown@example.com	567-890-1234	654 Cedar St	Phoenix	AZ	85001	USA
6	Emily	Davis	emily.davis@example.com	678-901-2345	987 Spruce St	Philadelphia	PA	19019	USA
7	Frank	Garcia	frank.garcia@example.com	789-012-3456	159 Birch St	San Antonio	TX	78201	USA
8	Grace	Martinez	grace.martinez@example.com	890-123-4567	753 Willow St	San Diego	CA	92101	USA
9	Hannah	Lopez	hannah.lopez@example.com	901-234-5678	852 Palm St	Dallas	TX	75201	USA
10	Isaac	Wilson	isaac.wilson@example.com	012-345-6789	963 Fir St	Austin	TX	73301	USA

The Output pane shows the execution results:

#	Time	Action	Message	Duration / Fetch
10	21:18:17	SELECT * FROM customer_info LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
11	21:18:27	CREATE VIEW US_Customers AS SELECT * FROM Customer WHERE Country = 'USA'	Error Code: 1050. Table 'US_Customers' already exists	0.000 sec
12	21:19:10	CREATE VIEW Customer_details AS SELECT CONCAT(First_name, ' ', Last_name) AS Full_name, E...	0 row(s) affected	0.016 sec
13	21:19:59	SELECT * FROM customer_info LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
14	21:20:23	select * from customer_details LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
15	21:21:47	select * from us_customers LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

## CREATED ANOTHER VIEW NAMED CUSTOMER\_DETAILS WITH COLUMNS FULL NAME,EMAIL,PHONE NO, STATE

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

```
70
71 -- Created a view named US_Customers that displays customers located in the US
72 CREATE VIEW US_Customers AS
73 SELECT * FROM Customer WHERE Country = 'USA';
74 select * from us_customers;
75
76 -- Created another view named Customer_details with columns full name(Combine first_name and last_name), email, phone_no, and state
77
78 CREATE VIEW Customer_details AS
79 SELECT CONCAT(First_name, ' ', Last_name) AS Full_name, Email, Phone_no, State FROM Customer;
80 select * from customer_details;
```

The Result Grid shows the output of the queries:

Full_name	Email	Phone_no	State
John Doe	john.doe@example.com	123-456-7890	CA
Jane Smith	jane.smith@example.com	234-567-8901	NY
Alice Johnson	alice.johnson@example.com	345-6 345-678-9012	TX
Bob Williams	bob.williams@example.com	456-789-0123	TX
Charlie Brown	charlie.brown@example.com	567-890-1234	AZ
Emily Davis	emily.davis@example.com	678-901-2345	PA
Frank Garcia	frank.garcia@example.com	789-012-3456	TX
Grace Martinez	grace.martinez@example.com	890-123-4567	CA
Hannah Lopez	hannah.lopez@example.com	901-234-5678	TX
Isaac Wilson	isaac.wilson@example.com	012-345-6789	TX

The Output pane shows the execution results:

#	Time	Action	Message	Duration / Fetch
11	21:18:27	CREATE VIEW US_Customers AS SELECT * FROM Customer WHERE Country = 'USA'	Error Code: 1050. Table 'US_Customers' already exists	0.000 sec
12	21:19:10	CREATE VIEW Customer_details AS SELECT CONCAT(First_name, ' ', Last_name) AS Full_name, E...	0 row(s) affected	0.016 sec
13	21:19:59	SELECT * FROM customer_info LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
14	21:20:23	select * from customer_details LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
15	21:21:47	select * from us_customers LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
16	21:22:09	select * from customer_details LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

## WRITED A QUERY THAT WILL RETURN NUMBER OF CUSTOMERS IN EACH STATE

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with a filter on 'customer\_details'. The main editor shows a SQL query:

```
88
89 • SELECT State, COUNT(*) AS Customer_Count
90 FROM Customer
91 GROUP BY State
92 HAVING COUNT(*) > 5;
93
94 -- Writed a query that will return the number of customers in each state, based on the "state" column in the "customer_details" view.
95
96 • SELECT State, COUNT(*) AS Customer_Count
97 FROM Customer_details
98 GROUP BY State;
```

The 'Result Grid' shows the output of the second query:

State	Customer_Count
CA	2
NY	1
IL	1
TX	4
AZ	1
PA	1

The 'Output' pane shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
16	21:22:09	select * from customer_details LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
17	21:23:28	UPDATE Customer_details SET Phone_no = 'CALIFORNIA-UPDATED-NUMBER' WHERE State = 'CA'	Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE...	0.000 sec
18	21:24:09	UPDATE Customer_details SET Phone_no = 'CALIFORNIA-UPDATED-NUMBER' WHERE State = 'CA'	Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE...	0.000 sec
19	21:24:51	SELECT State, COUNT(*) AS Customer_Count FROM Customer_details GROUP BY State HAVING COUNT(*) > 5	0 row(s) returned	0.000 sec / 0.000 sec
20	21:25:36	SELECT State, COUNT(*) AS Customer_Count FROM Customer_details GROUP BY State HAVING COUNT(*) > 5	0 row(s) returned	0.000 sec / 0.000 sec
21	21:26:12	SELECT State, COUNT(*) AS Customer_Count FROM Customer_details GROUP BY State LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec

## WRITED A QUERY THAT RETURNS ALL THE COLUMN FROM THE "CUSTOMER\_DETAILS" VIEW, SORTED BY "STATE" COLUMN IN ASCENDING ORDER

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with a filter on 'customer\_details'. The main editor shows a SQL query:

```
94 -- Writed a query that will return the number of customers in each state, based on the "state" column in the "customer_details" view.
95
96 • SELECT State, COUNT(*) AS Customer_Count
97 FROM Customer_details
98 GROUP BY State;
99
100 -- Write a query that returns all the columns from the "customer_details" view, sorted by the "state" column in ascending order.
101
102 • SELECT *
103 FROM Customer_details
104 ORDER BY State ASC;
```

The 'Result Grid' shows the output of the second query:

Full_name	Email	Phone_no	State
Charlie Brown	charlie.brown@example.com	567-890-1234	AZ
John Doe	john.doe@example.com	123-456-7890	CA
Grace Martinez	grace.martinez@example.com	890-123-4567	CA
Alice Johnson	alice.johnson@example.com	345-678-9012	IL
Jane Smith	jane.smith@example.com	234-567-8901	NY
Emily Davis	emily.davis@example.com	678-901-2345	PA
Bob Williams	bob.williams@example.com	456-789-0123	TX
Frank Garcia	frank.garcia@example.com	789-012-3456	TX
Hannah Lopez	hannah.lopez@example.com	frank.garcia@example.com	TX
Isaac Wilson	isaac.wilson@example.com	012-345-6789	TX

The 'Output' pane shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
1	21:27:15	USE Product		0.000 sec
2	21:27:30	SELECT * FROM Customer_details ORDER BY State ASC LIMIT 0, 1000	10 row(s) returned	0.032 sec / 0.000 sec