TASK #8

Scenario--Consider the Country table and Persons table that you created earlier and perform the following:

1. Find the number of persons in each country.

2. Find the number of persons in each country sorted from high to low.

3. Find out an average rating for Persons in respective countries if the average is greater than 3.0

4. Find the countries with the same rating as the USA. (Use Subqueries)

5. Select all countries whose population is greater than the average population of all nations.

6. Create a database named Product and create a table called Customer with the following fields in the Product database: Customer_Id - Make PRIMARY KEY First_name Last_name Email Phone_no Address City State Zip_code Country

7. Create a view named customer_info for the Customer table that displays Customer's Full name and email address. Then perform the SELECT operation for the customer_info view.

8. Create a view named US_Customers that displays customers located in the US.

9. Create another view named Customer_details with columns full name(Combine first_name and last_name), email, phone_no, and state.

10. Update phone numbers of customers who live in California for Customer_details view.

11. Count the number of customers in each state and show only states with more than 5 customers.

12. Write a query that will return the number of customers in each state, based on the "state" column in the "customer_details" view.

13. Write a query that returns all the columns from the "customer_details" view, sorted by the "state" column in ascending order.

1.—To find the number of persons in each country we used COUNT function on person's id and we grouped it by Country which results in calculating the number of person in each country.

Advantage of COUNT and GROUP BY—By performing COUNT we can count the number of records in a particular column and we can aggregate it based on another column by using the GROUP BY.

Here is the Query to achieve above advantage;

```
select Country_name,Count(id) as No_of_persons_each_country from Persons

group by Country_name;
```

2--Find the number of persons in each country sorted from high to low.

To find the number of persons in each country sorted from high to low we used ORDER BY Key .

Advantage of ORDER BY—ORDER BY key is used for sorting the records of particular column in Ascending or Descending order.

Here is the Query to achieve above advantage;

```
select Country_name,Count(id) as No_of_persons_each_country from Persons

group by Country_name

order by No_of_persons_each_country desc;
```

3-- Find out an average rating for Persons in respective countries if the average is greater than 3.0.

To achieve the above statement we used AVG(Rating) function .

Advantage of AVG(Column_name)—AVG helps to calculate the average value of records for the specified column.

Here is the query to achieve above advantage

```
select Country_name,AVG(Rating) as averageRating from Persons

group by Country_name
```

Having AVG(Rating)>3

4-- Find the countries with the same rating as the USA. (Use Subqueries)

Advantage of Using SUBQUERIES-- A Subquery or Inner query or Nested query is a query within an SQL query and embedded within the WHERE clause. A Subquery is a **SELECT statement** that is embedded in a clause of another SQL statement.

- Subqueries divide the complex query into isolated parts so that a complex query can be broken down into a series of logical steps.
- It is easy to understand and code maintenance is also at ease.
- Subqueries allow you to use the results of another query in the outer query.
- In some cases, subqueries can replace complex joins and unions

Here is the 2 Queries for above statement:

```
select Country_name from Persons

where Rating  = (Select AVG(Rating) from Persons where Country_name = "USA");
```

5-- Select all countries whose population is greater than the average population of all nations.

```
select Country_name from Persons

where Rating  > (Select AVG(Rating) from Persons);
```

6-- Create Database and Customer Table

- This query creates a new database called Product and a table named Customer with various fields like Customer_Id, First_name, Last_name, Email, etc. It then inserts sample data with Indian names and locations.

Commands Used:

- CREATE DATABASE and CREATE TABLE: These commands set up the environment and structure for storing customer information, defining the columns and their data types.
- INSERT INTO: Inserts sample rows of data into the Customer table for testing and demonstrating further queries.

Advantages:

- **Database and Table Creation**: Helps organize data, separating it logically (database) and structurally (table).
- **Data Insertion**: Adds initial data, allowing us to practice queries and understand how they would work in a real-world scenario.

7-- Create customer_info View

CREATE VIEW Customer_Info AS SELECT CONCAT(First_name, ' ', Last_name) AS Full_Name, Email FROM Customer;

Query Purpose:

- Creates a view that combines First_name and Last_name into a full name and displays it with the Email column. This simplifies the data presentation by focusing on just the name and email address.
- CONCAT: Combines First_name and Last_name into a single Full_Name column.

Advantages:

- **View Creation**: A view simplifies complex queries by saving them as a virtual table, making it easier to retrieve specific data.
- **Concatenation**: Using CONCAT helps generate a full name column, enhancing readability and reducing redundancy by not requiring First_name and Last_name separately.

8-- Create US_customers View

Create View US_customers AS select Concat(First_name,' ',Last_name) as full_name from Customer where Country='US';

Select * from US_customers;

Query Purpose:

- Creates a view to filter customers located in US, displaying all fields from the Customer table. This segmentation makes it easy to work with country-specific data.

Commands Used:

- WHERE clause: Filters records to only include those with Country = 'US'.

Advantages:

- **Filtered View**: Allows focused access to customers from US without affecting the base table, useful for location-specific analyses.
- **Simplified Access**: Users can retrieve all US customers quickly without adding multiple conditions to every query.

9-- Create Customer_details View

Create view Customer_details as Select CONCAT(First_name,' ',Last_name) as Full_name , Email,Phone_no,State from Customer;

Select * from Customer_details;

Query Purpose:

- Creates a view that combines First_name and Last_name into Full_Name, and includes email, phone number, and state. This view simplifies access to key customer contact details.

Advantages:

- **Data Simplification**: Reduces clutter by focusing on only essential columns.
- **Improved Querying**: Allows quick access to customer contact details without repeatedly typing out field concatenations and selections.

10-- Update Phone Numbers for California Customers

```
UPDATE Customer_details
SET Phone_no = '123456789' WHERE State = 'California';
```

Query Purpose:

- Updates the phone numbers of customers located in California to a standard placeholder, simulating a batch update for a specific condition.

Commands Used:

- UPDATE and WHERE: Allows conditional updates, applying changes only to records that match specific criteria (e.g., State = 'California').

Advantages:

Using WHERE ensures only relevant records are modified, helping maintain data accuracy.

11-- Count Customers in Each State with More Than 5 Customers

```
Select State from Customer
Group by State
Having count(Customer_Id)>5;
```

Query Purpose:

- Counts the number of customers in each state and filters results to show only states with more than 5 customers. This helps in identifying states with higher customer density.

Functions Used:

- GROUP BY and HAVING: GROUP BY groups the data by State, and HAVING filters the grouped results based on a condition.

Advantages:

- **Grouping and Filtering**: Helps in aggregating data by state and applying conditions post-aggregation, allowing a quick overview of customer distribution.

12-- Count Customers in Each State from Customer_details View

SELECT State, COUNT(*) AS Customer_Count FROM Customer_details

GROUP BY State;

Query Purpose:

- Similar to the previous query but uses the Customer_details view. It groups customers by state and counts the total in each state, useful for understanding customer distribution.

13--Retrieve All Columns from Customer_details, Sorted by State

SELECT *FROM Customer_details

ORDER BY State ASC;

Query Purpose:

- Returns all columns from the Customer_details view, sorted alphabetically by State. This ordering can help in region-based reporting or analysis.

Commands Used:

- ORDER BY: Sorts the result by the State column in ascending order.

Advantages: ORDER BY key is used for sorting the records of particular column in Ascending or Descending order.

Summary of Benefits

Using views, functions like CONCAT, and commands such as GROUP BY, HAVING, and ORDER BY provides:

- **Data Simplification**: Views and functions simplify complex queries, making data more accessible.
- **Targeted Data Manipulation**: Commands like UPDATE, WHERE, and HAVING offer control over data modifications and retrieval.
- **Efficient Analysis**: Aggregation and filtering enhance the ability to analyze data meaningfully without altering the original table structure.

These tools collectively make it easier to manage, access, and analyze customer data in a structured, organized way.