Advanced MySQL Practice Questions

1. Write a query to display customer-wise total purchase amount along with the ranking of customers based on their total purchase in descending order using appropriate window functions.

2. Generate a report that lists all employees along with their manager name using self join concept.

3. Retrieve month-wise sales totals for each year and show the percentage contribution of each month to its respective year.

4. Display product details where the selling price is higher than the average selling price of all products using subqueries and aggregate functions.

5. Create a query to show customers who have placed more than 5 orders along with their total order value.

6. Using appropriate joins, generate a list of all customers and their corresponding payment amounts. Customers without payments must also be included.

7. Write a query to find the top 3 highest selling products for each year using window functions such as DENSE_RANK.

8. Display order details along with previous order amount for the same customer using LAG function.

9. Retrieve the first and last payment amount made by each customer using FIRST_VALUE and LAST_VALUE window functions.

10. Create a report that shows total sales by year, quarter and month in a single result set using date time functions.

11. Find all customers whose total payment amount is greater than the average payment amount of all customers.

12. Generate a query that categorizes orders into High, Medium and Low value orders using CASE expression.

13. Display details of products that have never been ordered by any customer using appropriate join technique.

14. Create a stored procedure that accepts two date parameters and returns total sales amount between those dates.

15. Write a query to find employees who handle customers from more than 3 different countries.

16. Generate a result showing year-on-year growth percentage of total sales.

17. Display customer details along with the difference in days between their first order and last order using date functions.

18. Write a query to find products whose total quantity ordered is greater than the overall average quantity ordered of all products.

19. Create a stored procedure with multiple input parameters that returns customer-wise total orders and total payments within a given date range and minimum amount criteria.

20. Create a trigger that prevents deletion of any record from a table if related transactional records exist in another table and raises an appropriate error message.