1) What is the highest salary paid in each organization, ordered by salary?

```
SELECT OC.Organization_Branch, MAX(E.Employee_Salary) AS Highest_Salary
FROM ORGANIZATION_CENTER OC
FULL OUTER JOIN EMPLOYEE E ON OC.Organization_Center_ID = E.Organization_ID
GROUP BY OC.Organization_Branch
ORDER BY Highest_Salary DESC;
```

Expected Solution: A list of organizations and the highest salary paid within each, ordered by salary.
Concepts Used (4): FULL OUTER JOIN, GROUP BY, MAX, ORDER BY.

2) Which blood drives collected more blood than the average collected across all drives?

```
SELECT BD.Blood_Drive_ID, BD.Drive_Location, SUM(D.Amount_Collected) AS
Total_Collected
FROM BLOOD_DRIVE BD
JOIN DONATION D ON BD.Blood_Drive_ID = D.Blood_Drive_ID
GROUP BY BD.Blood_Drive_ID, BD.Drive_Location
HAVING SUM(D.Amount_Collected) > (
   SELECT AVG(Total_Collected) FROM (
      SELECT Blood_Drive_ID, SUM(Amount_Collected) AS Total_Collected
      FROM DONATION
      GROUP BY Blood_Drive_ID
   )
)
ORDER BY Total_Collected DESC;
```

Expected Solution: A list of blood drives that exceeded the average amount of blood collected.
Concepts Used (5): JOIN (Inner Join), GROUP BY, HAVING, SUM, Subquery.

3) Which blood groups are most in demand at hospitals (based on the blood groups of patients in severe conditions)?

```
SELECT P.Patient_Blood_Group, COUNT(*) AS Number_of_Patients
FROM PATIENT P
WHERE P.Patient_Severity = 'Severe'
```

```
GROUP BY P.Patient_Blood_Group;
```

Expected Solution: A list of blood groups with the number of severe patients needing each group.
Concepts Used (3): WHERE, GROUP BY, COUNT.

4) Which employees are affiliated with organizations that have a capacity of more than 50?

```
SELECT E.Employee_Name, E.Employee_Role, OC.Organization_Branch
FROM EMPLOYEE E
LEFT JOIN ORGANIZATION_CENTER OC ON E.Organization_ID =
OC.Organization_Center_ID
WHERE OC.Organization_Capacity > 50;
```

Expected Solution: A list of employees working at organization centers with a capacity greater than 50.
Concepts Used (2): LEFT OUTER JOIN, WHERE.

5) Which blood group has the highest average donation and how does it compare to the overall average?

```
SELECT
   D.Donor_Blood_group,
   AVG(Do.Amount_Collected) AS Avg_Collected,
   (SELECT AVG(Amount_Collected) FROM DONATION) AS Overall_Avg
FROM DONOR D
JOIN DONATION Do ON D.Donor_ID = Do.Donor_ID
GROUP BY D.Donor_Blood_group
HAVING AVG(Do.Amount_Collected) = (
   SELECT MAX(Avg_Amount) FROM (
      SELECT AVG(Amount_Collected) AS Avg_Amount
      FROM DONATION
      GROUP BY Blood_Drive_ID
   )
)
ORDER BY Avg_Collected DESC;
```

Expected Solution: A list of blood groups with the highest average donation amount compared to the overall average.

6) Identify donors and recipients with matching blood groups and their donation and requirement amounts.

```
SELECT
   D.Donor_Name,
   R.Recipient_Name,
   D.Donor_Blood_group,
   Do.Amount_Collected,
   R.Blood_Bags_Count
FROM DONOR D
JOIN DONATION Do ON D.Donor_ID = Do.Donor_ID
JOIN RECIPIENT R ON D.Donor_Blood_group = R.Recipient_Blood_Group
WHERE Do.Amount_Collected >= R.Blood_Bags_Count
ORDER BY D.Donor_Blood_group, Do.Amount_Collected DESC;
```

Expected Solution: List of donors and recipients with matching blood groups, including donation and requirement amounts.

Concepts Used (4): ORDER BY, Comparison Operators, INNER Joins, and Creating Joins with the ON Clause

7) Find donors who have made the highest number of donations, their total donation count, and the average donation count across all donors.

```
SELECT
   D.Donor_Name,
   COUNT(*) AS Total_Donations,
   AVG(COUNT(*)) OVER () AS Average_Donations
FROM DONOR D
JOIN DONATION Do ON D.Donor_ID = Do.Donor_ID
GROUP BY D.Donor_Name
HAVING COUNT(*) = (
   SELECT MAX(Total_Donations) FROM (
      SELECT D.Donor_Name, COUNT(*) AS Total_Donations
      FROM DONOR D
      JOIN DONATION Do ON D.Donor_ID = Do.Donor_ID
      GROUP BY D.Donor_Name
   )
);
```

8) List donors who have donated more than the average donation amount for their blood group.

```
SELECT D.Donor_ID, D.Donor_Name, D.Donor_Blood_group, SUM(Do.Amount_Collected) AS
Total_Amount_Collected
FROM DONOR D
JOIN DONATION Do ON D.Donor_ID = Do.Donor_ID
GROUP BY D.Donor_ID, D.Donor_Name, D.Donor_Blood_group
HAVING SUM(Do.Amount_Collected) > (
   SELECT AVG(Amount_Collected)
   FROM DONATION
   WHERE Donor_ID IN (
      SELECT Donor_ID
      FROM DONOR
      WHERE Donor_Blood_group = D.Donor_Blood_group
   )
);
```

Expected Solution: List of donors whose total donation amount exceeds the average for their blood group.

Concepts Used (5): JOIN, GROUP BY, SUM, HAVING, Subquery.

9) Identify organization centers with their average operational costs and total revenue, having revenue greater than average revenue.

```
SELECT OC.Organization_Branch, AVG(A.Operational_cost) AS Average_Operational_Cost,
SUM(A.Blood_Sales_Revenue) AS Total_Revenue
FROM ORGANIZATION_CENTER OC
JOIN ACCOUNTS A ON OC.Organization_Center_ID = A.Organization_ID
GROUP BY OC.Organization_Branch
HAVING SUM(A.Blood_Sales_Revenue) > (
   SELECT AVG(Blood_Sales_Revenue) FROM ACCOUNTS
);
```

Expected Solution: Organization centers with their average operational costs and total revenue, where revenue is above average.

Concepts Used (5): JOIN, GROUP BY, AVG, SUM, HAVING with Subquery.

10) Calculate the average number of blood bags required per age group for each blood type among recipients.

```
SELECT R.Recipient_Blood_Group,
      CASE
        WHEN R.Recipient_age < 30 THEN 'Under 30'
```

```
        WHEN R.Recipient_age BETWEEN 30 AND 60 THEN '30-60'
        ELSE 'Over 60'
    END AS Age_Group,
    AVG(R.Blood_Bags_Count) AS Average_Bags_Required
FROM RECIPIENT R
GROUP BY R.Recipient_Blood_Group, CASE
                        WHEN R.Recipient_age < 30 THEN 'Under 30'
                        WHEN R.Recipient_age BETWEEN 30 AND 60 THEN '30-60'
                        ELSE 'Over 60'
                    END;
```

Expected Solution: Average number of blood bags required for each blood type categorized by recipient age group.
Concepts Used (4): GROUP BY, CASE, AVG, Aliasing.

11) Find donors who have never donated to blood drives that collected less than the average amount of blood.

```
SELECT DISTINCT D.Donor_ID, D.Donor_Name
FROM DONOR D
WHERE NOT EXISTS (
  SELECT 1
  FROM DONATION Do
  JOIN BLOOD_DRIVE BD ON Do.Blood_Drive_ID = BD.Blood_Drive_ID
  WHERE D.Donor_ID = Do.Donor_ID AND BD.Blood_Bag_Count < (
    SELECT AVG(Blood_Bag_Count) FROM BLOOD_DRIVE
  )
);
```

Expected Solution: List of donors who have never donated to underperforming blood drives.
Concepts Used (4): DISTINCT, NOT EXISTS, Subquery, JOIN.

12) Rank donors within each blood group based on their total amount of blood donated.

```
SELECT D.Donor_Blood_group, D.Donor_Name, SUM(Do.Amount_Collected) AS
Total_Amount_Collected,
    RANK() OVER (PARTITION BY D.Donor_Blood_group ORDER BY
SUM(Do.Amount_Collected) DESC) AS Rank
FROM DONOR D
JOIN DONATION Do ON D.Donor_ID = Do.Donor_ID
GROUP BY D.Donor_Blood_group, D.Donor_Name
ORDER BY D.Donor_Blood_group, Total_Amount_Collected DESC;
```

Expected Solution: Each donor's rank within their blood group based on the total amount of blood donated.
Concepts Used (5): JOIN, GROUP BY, SUM, Window Function (RANK), ORDER BY.

1, 5, 7, 9, 10, 12 covers 10 concepts from the given list of concepts

JOIN

Types: INNER JOIN, FULL OUTER JOIN, LEFT OUTER JOIN
Count: 4 (INNER JOIN in queries 5, 7, 9, 12; FULL OUTER JOIN in query 1; LEFT OUTER JOIN is conceptually included as part of OUTER JOINs)
GROUP BY

Count: 6 (Used in every query)
ORDER BY

Count: 3 (Queries 1, 5, 12)
HAVING

Count: 3 (Queries 5, 7, 9)
Subqueries

Count: 3 (Queries 5, 7, 9)
Aggregating Data Using Group Functions

Functions: AVG, SUM, COUNT, MAX
Count: 6 (AVG in queries 5, 9, 10; SUM in queries 5, 7, 9, 12; COUNT in queries 7, 12; MAX in query 1)
Window Function

Function: RANK
Count: 1 (Query 12)
Aliasing

Count: 1 (Query 10 with the CASE statement)
Comparison Operators

Operators: >, =
Count: 3 (Used in the HAVING clauses and WHERE clauses)
CASE

Count: 1 (Query 10)