

Sure! I'll update the markdown to include captions for each output image with the problem number labeled as "output". Here's the revised version:

Employee Database Analysis

1. Create Database and Table

```
-- Create the database
CREATE DATABASE HREmployeeDB;
USE HREmployeeDB;

-- Drop the table if it already exists
IF OBJECT_ID('EmployeeData', 'U') IS NOT NULL
    DROP TABLE EmployeeData;

-- Create the EmployeeData table
CREATE TABLE EmployeeData (
    Attrition NVARCHAR(50),
    BusinessTravel NVARCHAR(50),
    CF_age_band NVARCHAR(50),
    CF_attrition_label NVARCHAR(50),
    Department NVARCHAR(50),
    EducationField NVARCHAR(50),
    emp_no NVARCHAR(50),
    EmployeeNumber INT,
    Gender NVARCHAR(50),
    JobRole NVARCHAR(50),
    MaritalStatus NVARCHAR(50),
    OverTime NVARCHAR(50),
    Over18 NVARCHAR(50),
    TrainingTimesLastYear INT,
    Age INT,
    CF_current NVARCHAR(50),
    DailyRate INT,
    DistanceFromHome INT,
    Education NVARCHAR(50),
    EmployeeCount INT,
    EnvironmentSatisfaction INT,
    HourlyRate INT,
    JobInvolvement INT,
    JobLevel INT,
    JobSatisfaction INT,
    MonthlyIncome INT,
    MonthlyRate INT,
    NumCompaniesWorked INT,
    PercentSalaryHike INT,
    PerformanceRating INT,
    RelationshipSatisfaction INT,
    StandardHours INT,
    StockOptionLevel INT,
```

```
TotalWorkingYears INT,  
WorkLifeBalance INT,  
YearsAtCompany INT,  
YearsInCurrentRole INT,  
YearsSinceLastPromotion INT,  
YearsWithCurrManager INT  
);  
  
-- Bulk insert data into the table  
BULK INSERT EmployeeData  
FROM 'C:\Users\Administrator\Downloads\HR_Employee1.csv'  
WITH  
(  
    FIELDTERMINATOR = ',', -- Delimiter for fields  
    ROWTERMINATOR = '0x0a', -- End of each row  
    FIRSTROW = 2 -- Skip header row  
);
```

A. Return the Shape of the Table

```
-- Number of rows and columns  
SELECT  
    (SELECT COUNT(*) FROM EmployeeData) AS row_bo,  
    (SELECT COUNT(*) FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME =  
    'EmployeeData') AS no_columns;
```



B. Gender Strength in Each Department

```
WITH GenderCounts AS (  
    SELECT  
        Department,  
        Gender,  
        COUNT(*) AS counts,  
        SUM(CASE WHEN Gender = 'Male' THEN 1 ELSE 0 END) AS Males,  
        SUM(CASE WHEN Gender = 'Female' THEN 1 ELSE 0 END) AS Females  
    FROM  
        EmployeeData  
    GROUP BY  
        Department,  
        Gender  
)  
SELECT  
    Department,  
    Gender AS ProminentGender,  
    counts,  
    RANK() OVER (PARTITION BY Department ORDER BY counts DESC) AS Gender_Rank  
FROM GenderCounts;
```

 Output

C. Create and Show Distribution of AGE_BAND

```
ALTER TABLE EmployeeData  
ADD Age_Band NVARCHAR(50);  
  
UPDATE EmployeeData  
SET Age_Band = CASE  
    WHEN Age < 25 THEN 'Below 25'  
    WHEN Age BETWEEN 25 AND 34 THEN '25-34'  
    WHEN Age BETWEEN 35 AND 44 THEN '35-44'  
    WHEN Age BETWEEN 45 AND 55 THEN '45-55'  
    ELSE 'Above 55'  
END;  
  
SELECT  
    Age_Band,  
    COUNT(*) AS Count  
FROM EmployeeData  
GROUP BY Age_Band;
```

 Output

D. Compare All Marital Status and Find Most Frequent

```
-- Query to get marital status count and frequency rank
SELECT
    MaritalStatus,
    Count,
    RANK() OVER (ORDER BY Count DESC) AS Freq_Rank
FROM (
    SELECT
        MaritalStatus,
        COUNT(*) AS Count
    FROM EmployeeData
    GROUP BY MaritalStatus
) AS _;
```

 Output

E. Job Role with Highest Attrition Rate

```
WITH AttritionRate AS (
    SELECT
        JobRole,
        (SUM(CASE WHEN Attrition = 'Yes' THEN 1 ELSE 0 END) * 100.0) / COUNT(*) AS
Attrition_Percentage
    FROM EmployeeData
    GROUP BY JobRole
)
SELECT TOP 1
    JobRole,
    Attrition_Percentage
FROM AttritionRate
ORDER BY Attrition_Percentage DESC;
```

 Output

F. Distribution of Employee's Promotion

```
SELECT YearsSinceLastPromotion, COUNT(*) AS EmployeeCount
FROM EmployeeData
GROUP BY YearsSinceLastPromotion
ORDER BY YearsSinceLastPromotion;
```

 Output

G. Cumulative Sum of Total Working Years for Each Department

```
SELECT
    Department,
```

```
TotalWorkingYears,  
SUM(TotalWorkingYears) OVER (PARTITION BY Department ORDER BY  
TotalWorkingYears ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS  
Cumulative_sum_year  
FROM EmployeeData;
```

 Output

H. Rank Employees by Monthly Income Within Each Department

```
SELECT  
EmployeeNumber,  
Department,  
MonthlyIncome,  
RANK() OVER (PARTITION BY Department ORDER BY MonthlyIncome DESC) AS  
Income_Rank  
FROM EmployeeData;
```

 Output

I. Running Total of Total Working Years for Each Employee

```
SELECT  
EmployeeNumber,  
Department,  
AGE_BAND,  
TotalWorkingYears,  
SUM(TotalWorkingYears) OVER (PARTITION BY Department, AGE_BAND ORDER BY  
EmployeeNumber) AS Running_Total_Working_Years  
FROM EmployeeData;
```

 Output

J. Years Worked Before Leaving vs. Average Years in Department

```
WITH YearsWorked AS (  
SELECT  
EmployeeNumber,  
Department,  
TotalWorkingYears AS Years_Worked_Before_Leaving  
FROM EmployeeData  
WHERE Attrition = 'Yes'  
)  
AvgYears AS (  
SELECT  
Department,
```

```

        AVG(TotalWorkingYears) AS Avg_Years_Worked
    FROM EmployeeData
    GROUP BY Department
)
SELECT
    Y.EmployeeNumber,
    Y.Department,
    Y.Years_Worked_Before_Leaving,
    A.Avg_Years_Worked
FROM YearsWorked Y, AvgYears A
WHERE Y.Department = A.Department AND Y.Years_Worked_Before_Leaving IS NOT NULL
ORDER BY
    Department,
    Years_Worked_Before_Leaving;

```

 Output

K. Rank Departments by Average Monthly Income of Employees Who Have Left

```

WITH DepartmentIncome AS (
    SELECT
        Department,
        AVG(MonthlyIncome) AS Avg_Monthly_Income
    FROM EmployeeData
    WHERE Attrition = 'Yes'
    GROUP BY Department
)
SELECT
    Department,
    Avg_Monthly_Income,
    RANK() OVER (ORDER BY Avg_Monthly_Income DESC) AS Department_Rank
FROM DepartmentIncome;

```

 Output

L. Relation Between Attrition Rate and Marital Status

```

SELECT
    MaritalStatus,
    (SUM(CASE WHEN Attrition = 'Yes' THEN 1 ELSE 0 END) * 100.0) / COUNT(*) AS
    Attrition_Percentage
FROM EmployeeData
GROUP BY MaritalStatus;

```

 Output

M. Department with Highest Attrition Rate

```
SELECT TOP 1
    Department,
    (SUM(CASE WHEN Attrition = 'Yes' THEN 1 ELSE 0 END) * 100.0) / COUNT(*) AS
    Attrition_Percentage
FROM EmployeeData
GROUP BY Department
ORDER BY Attrition_Percentage DESC;
```

 Output

N. Moving Average of Monthly Income Over the Past 3 Employees

```
WITH RankedEmployees AS (
    SELECT
        JobRole,
        MonthlyIncome,
        ROW_NUMBER() OVER (PARTITION BY JobRole ORDER BY EmployeeNumber DESC) AS
rn
    FROM EmployeeData
)
SELECT
    JobRole,
    MonthlyIncome,
    AVG(MonthlyIncome) OVER (PARTITION BY JobRole ORDER BY rn ROWS BETWEEN 2
PRECEDING AND CURRENT ROW) AS MovingAverage
FROM RankedEmployees;
```

 Output


O. Identify Employees with Outliers in Monthly Income

```
WITH IncomeStats AS (
    SELECT
        JobRole,
        EmployeeNumber,
        MonthlyIncome,
        PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY MonthlyIncome) OVER
(PARTITION BY JobRole) AS Q1,
        PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY MonthlyIncome) OVER
(PARTITION BY JobRole) AS Q3
    FROM EmployeeData
)
SELECT
    EmployeeNumber,
    JobRole,
```

```

MonthlyIncome,
CASE
    WHEN MonthlyIncome < Q1 - (Q3 - Q1) * 1.5 THEN 'Low'
    WHEN MonthlyIncome > Q3 + (Q3 - Q1) * 1.5 THEN 'High'
    ELSE 'Not an Outlier'
END AS OutlierType
FROM IncomeStats
WHERE MonthlyIncome < Q1 - (Q3 -
Q1) * 1.5
OR MonthlyIncome > Q3 + (Q3 - Q1) * 1.5;

```

 Output

P. Gender Distribution Within Each Job Role

```

WITH GenderCount AS
(
    SELECT
        JobRole,
        SUM(CASE WHEN Gender = 'Male' THEN 1 ELSE 0 END) AS Male_Count,
        SUM(CASE WHEN Gender = 'Female' THEN 1 ELSE 0 END) AS Female_Count
    FROM EmployeeData
    GROUP BY JobRole
)
SELECT
    *,
    CASE
        WHEN Male_count > Female_Count THEN 'Male'
        WHEN Female_Count > Male_count THEN 'Female'
        ELSE 'Equal'
    END AS Dominant_Gender
FROM
    GenderCount;

```

 Output

Q. Percent Rank of Employees Based on Training Times Last Year

```


SELECT
    EmployeeNumber,
    TrainingTimesLastYear,
    PERCENT_RANK() OVER (ORDER BY TrainingTimesLastYear) * 100 AS PercentRank
FROM EmployeeData;

```

 Output

R. Divide Employees into 5 Groups Based on Training Times Last Year


```
SELECT
    EmployeeNumber,
    TrainingTimesLastYear,
    NTILE(5) OVER (ORDER BY TrainingTimesLastYear) AS Training_Group
FROM EmployeeData;
```

 Output

S. Categorize Employees Based on Training Times Last Year

```
WITH TrainingTimeTiling AS
(
    SELECT
        EmployeeNumber,
        TrainingTimesLastYear,
        NTILE(3) OVER (ORDER BY TrainingTimesLastYear) AS TrainingTile
    FROM EmployeeData
)
SELECT
    EmployeeNumber,
    TrainingTimesLastYear,
    CASE
        WHEN TrainingTile = 3 THEN 'Frequent Trainee'
        WHEN TrainingTile = 2 THEN 'Moderate Trainee'
        ELSE 'Infrequent Trainee'
    END AS TraineeCategory
FROM TrainingTimeTiling;
```

 Output

T. Categorize Employees as 'High', 'Medium', or 'Low' Performers

```
SELECT
    EmployeeNumber,
    PerformanceRating,
    CASE
        WHEN PerformanceRating >= 4 THEN 'High'
        WHEN PerformanceRating = 3 THEN 'Medium'
        ELSE 'Low'
    END AS PerformanceCategory
FROM EmployeeData;
```

 Output

U. Categorize Employees into Work-Life Balance Categories

```
SELECT
    EmployeeNumber,
    WorkLifeBalance,
    CASE
        WHEN WorkLifeBalance = 1 THEN 'Poor'
        WHEN WorkLifeBalance = 2 THEN 'Fair'
        WHEN WorkLifeBalance = 3 THEN 'Good'
        ELSE 'Excellent'
    END AS Work_Life_Balance_Category
FROM EmployeeData;
```

 Output

V. Group Employees Based on Stock Option Level

```
SELECT
    EmployeeNumber,
    StockOptionLevel,
    NTILE(3) OVER (ORDER BY StockOptionLevel) AS StockOption_Group
FROM EmployeeData;
```

 Output

W. Key Reasons for Attrition in the Company

```
SELECT
    Attrition,
    BusinessTravel,
    Department,
    MaritalStatus,
    JobRole,
    AVG(MonthlyIncome) AS AvgIncome,
    COUNT(*) AS Count
FROM EmployeeData
WHERE Attrition = 'Yes'
GROUP BY Attrition, BusinessTravel, Department, MaritalStatus, JobRole
ORDER BY Count DESC;
```

 Output