EMPLOYEE ATTRITION PREDICTION: SQL

Creating table after connecting to the database

Query | Query History

```
CREATE TABLE Employees (
    EmployeeID SERIAL PRIMARY KEY, JobRole VARCHAR(50), Attrition Varchar(10),
    Department VARCHAR(255), Age INT, BusinessTravel VARCHAR(255),
    DistanceFromHome INT,Education INT, EducationField VARCHAR(255),
    EmployeeCount VARCHAR(255),EmployeeNumber varchar(255),
    EnvironmentSatisfaction INT, Gender VARCHAR(50), HourlyRate INT,
    JobInvolvement INT,JobLevel INT, JobSatisfaction INT,
    MaritalStatus VARCHAR(255), MonthlyIncome INT, DailyRate INT,
    MonthlyRate INT, NumCompaniesWorked INT, Over18 VARCHAR(5),
    OverTime VARCHAR(5), PercentSalaryHike INT, PerformanceRating INT,
    RelationshipSatisfaction INT, StandardHours INT, StockOptionLevel INT,
    TotalWorkingYears INT, TrainingTimesLastYear INT, WorkLifeBalance INT,
    YearsAtCompany INT,YearsInCurrentRole INT,
    YearsSinceLastPromotion INT, YearsWithCurrManager INT
);
```

First view of data set

Data Output | Messages | Notifications

Please click here for more information.

	employeeid [PK] integer	jobrole character varying (50)	attrition character varying (10)	department character varying (255)	age integer	businesstravel character varying (255)	distancefromhome integer	education integer	educationfield character varying (255)
1	1	Role_2	No	R&D	21	Travel_Rarely	2	2	Technical Degree
2	2	Role_3	Yes	R&D	22	Travel_Rarely	3	3	Technical Degree
3	4	Role_5	Yes	R&D	24	Travel_Rarely	5	5	Life Sciences
4	5	Role_1	No	R&D	25	Travel_Rarely	6	1	Technical Degree
5	14001	Data Scientist	No	Research & Development	28	Travel_Rarely	5	4	Life Sciences
6	7	Role_3	No	R&D	27	Travel_Rarely	8	3	Technical Degree
7	8	Role_4	Yes	R&D	28	Travel_Rarely	9	4	Life Sciences
8	1401	Data Scientist	No	Research & Development	28	Travel_Rarely	5	4	Life Sciences
9	10	Role_1	Yes	R&D	30	Travel_Rarely	11	1	Technical Degree
10	11	Role_2	No	R&D	31	Travel_Rarely	12	2	Technical Degree
11	13	Role_4	No	R&D	33	Travel_Rarely	14	4	Technical Degree
12	14	Role_5	Yes	R&D	34	Travel_Rarely	15	5	Technical Degree
13	16	Role_2	Yes	R&D	36	Travel_Rarely	17	2	Life Sciences
14	17	Role_3	No	R&D	37	Travel_Rarely	18	3	Technical Degree
15	19	Role_5	No	R&D	39	Travel_Rarely	20	5	Technical Degree
16	20	Role_1	Yes	R&D	40	Travel_Rarely	21	1	Life Sciences
17	22	Role_3	Yes	R&D	42	Travel_Rarely	23	3	Technical Degree
18	23	Role_4	No	R&D	43	Travel_Rarely	24	4	Technical Degree
19	25	Role_1	No	R&D	45	Travel_Rarely	26	1	Technical Degree

1. Departments with the Highest Attrition:

```
ECT department, count(attrition) as max_attrition
M employees
RE attrition = 'Yes'
UP BY department
ER BY max_attrition desc, department
```

	department character varying (255)	max_attrition bigint
1	R&D	467
2	Sales	233

2. Gender Distribution across Different Departments:

```
SELECT department, gender, count(*) as no_of_employees
FROM employees
group by 1,2
order by 3 desc
```

Data Output | Messages | Notifications

	department character varying (255)	gender character varying (50)	no_of_employees bigint
1	R&D	Male	467
2	R&D	Female	467
3	Sales	Male	233
4	Sales	Female	233
5	Research & Development	Female	2

3. Job Satisfaction

```
SELECT CASE WHEN jobsatisfaction=1 THEN 'Not satisfied'
WHEN jobsatisfaction=2 THEN 'Moderately satisfied'
WHEN jobsatisfaction=3 THEN 'Satisfied'
ELSE 'Very satisfied'
END AS Satisfaction_level, count(*) as employee_count
FROM employees
GROUP BY 1
order by 2 desc
```

Data Output | Messages | Notifications

	satisfaction_level text	employee_count bigint
1	Very satisfied	352
2	Moderately satisfied	350
3	Not satisfied	350
4	Satisfied	350

4. EMPLOYEE ATTRITION RATE BASED ON DIFFERENT FACTORS

4.a Attrition based on Age-group

```sql
SELECT
    CASE
        WHEN Age BETWEEN 18 AND 25 THEN '18-25'
        WHEN Age BETWEEN 26 AND 30 THEN '26-30'
        WHEN Age BETWEEN 31 AND 35 THEN '31-35'
        WHEN Age BETWEEN 36 AND 40 THEN '36-40'
        WHEN Age BETWEEN 41 AND 50 THEN '41-50'
        WHEN Age BETWEEN 51 AND 60 THEN '51-60'
    END AS Age_Range,
    ROUND(
        (SUM(CASE WHEN Attrition = 'Yes' THEN 1 ELSE 0 END)* 100.0) / COUNT(*),2) AS Attrition_Rate
FROM
    employees
GROUP BY
    CASE
        WHEN Age BETWEEN 18 AND 25 THEN '18-25'
        WHEN Age BETWEEN 26 AND 30 THEN '26-30'
        WHEN Age BETWEEN 31 AND 35 THEN '31-35'
        WHEN Age BETWEEN 36 AND 40 THEN '36-40'
        WHEN Age BETWEEN 41 AND 50 THEN '41-50'
        WHEN Age BETWEEN 51 AND 60 THEN '51-60' END
ORDER BY
    Age_Range;
```

Output Messages Notifications

	age_range text	attrition_rate numeric
1	18-25	49.82
2	26-30	59.49
3	31-35	40.00
4	36-40	60.00
5	41-50	44.44

4.b Attrition based on Job Role

```sql
SELECT jobrole, round((sum(case when attrition='Yes' then 1 else 0 end)*100.0)/count(*),2) as attrition_rate
from employees
group by jobrole
order by attrition_rate desc
```

Data Output | Messages | Notifications

	jobrole character varying (50)	attrition_rate numeric
1	Role_3	50.00
2	Role_2	50.00
3	Role_1	50.00
4	Role_5	50.00
5	Role_4	50.00

4.c Attrition rate based on Performance

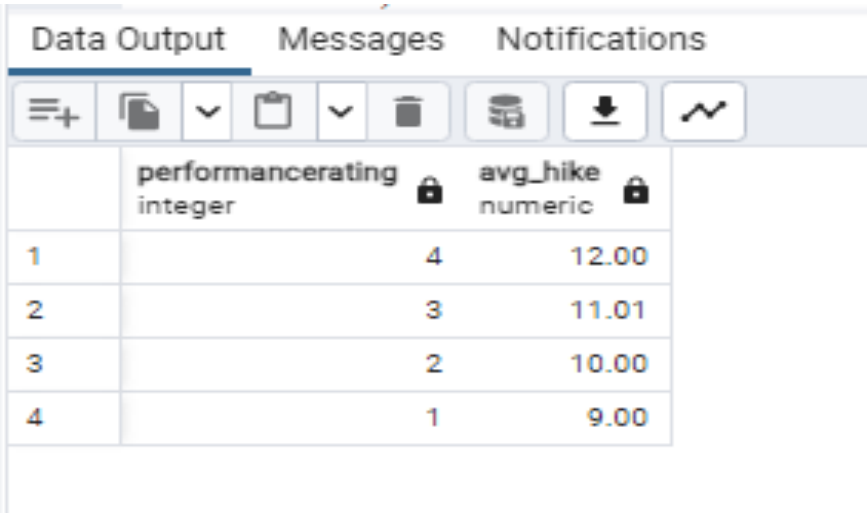
```
select stockoptionlevel as stock_level, count(*) as emp_count
from employees
where attrition='Yes'
group by stock_level
```

Data Output | Messages | Notifications

	performancerating integer	attrition_rate numeric
1	1	100.00
2	3	99.43
3	4	0.00
4	2	0.00

5. Correlation between Percent Salary Hike and Performance Rating:

```sql
select performancerating , round(avg(percentsalaryhike),2) as avg_hike
from employees
group by 1
order by 1 desc
```



Data Output | Messages | Notifications

	performancerating integer	avg_hike numeric
1	4	12.00
2	3	11.01
3	2	10.00
4	1	9.00

6. Career Progression Analysis: What is the average years at the company and total working years for different job roles?

```sql
SELECT
    JobRole,
    round(AVG(YearsAtCompany),2) AS Avg_Years_At_Company,
    round(AVG(TotalWorkingYears),2) AS Avg_Total_Working_Years
FROM employees
GROUP By JobRole
ORDER BY Avg_Total_Working_Years DESC;
```

	jobrole character varying (50)	avg_years_at_company numeric	avg_total_working_years numeric
1	Role_5	16.43	21.50
2	Role_4	15.43	20.50
3	Role_3	14.43	19.50
4	Role_2	13.43	18.50
5	Role_1	12.50	17.50

7. Gender Pay Gap Analysis: What is the average monthly income of male and female employees?

```
SELECT
    Gender,round(AVG(MonthlyIncome),0) AS Avg_Monthly_Income
FROM employees
GROUP BY Gender;
```

Data Output Messages Notifications

	gender character varying (50)	avg_monthly_income numeric
1	Female	3949
2	Male	3947

8. Distribution of performance ratings across different job roles and departments

```sql
SELECT
    JobRole,PerformanceRating,COUNT(*) AS Count
FROM employees
GROUP BY JobRole, PerformanceRating
ORDER BY  JobRole, PerformanceRating;
```

Data Output | Messages | Notifications

	jobrole character varying (50)	performancerating integer	count bigint
1	Role_1	1	70
2	Role_1	2	70
3	Role_1	3	70
4	Role_1	4	70
5	Role_2	1	70
6	Role_2	2	70
7	Role_2	3	70

9. Find the highest earners in each job role

```sql
SELECT JobRole,EmployeeID, MonthlyIncome
FROM (
    SELECT
        EmployeeID,JobRole, MonthlyIncome,
  RANK() OVER (PARTITION BY JobRole ORDER BY MonthlyIncome DESC) AS IncomeRank
  FROM employees
) subquery
WHERE  IncomeRank =1 ;
```

Data Output Messages Notifications

	jobrole character varying (50)	employeeid [PK] integer	monthlyincome integer
1	Role_1	1395	5274
2	Role_2	1386	5263
3	Role_3	1392	5270
4	Role_4	1398	5278
5	Role_5	1389	5267

10. What is the Employee Retention Rate by Job Role

```
SELECT
    JobRole,
    COUNT(*) AS Total_Employees,
    SUM(CASE WHEN Attrition = 'No' THEN 1 ELSE 0 END) AS Retained_Employees,
   round( SUM(CASE WHEN Attrition = 'No' THEN 1 ELSE 0 END) * 100.0 / COUNT(*),2) AS Retention_Rate
FROM
    employees
GROUP BY
    JobRole
ORDER BY Retention_Rate DESC;
```

Data Output Messages Notifications

	jobrole character varying (50)	total_employees bigint	retained_employees bigint	retention_rate numeric
1	Role_3	280	140	50.00
2	Role_2	280	140	50.00
3	Role_1	280	140	50.00
4	Role_5	280	140	50.00
5	Role_4	280	140	50.00

UPDATE Monthly income for Sales Department by 20%

```
--Start a transaction
BEGIN;

UPDATE Employees
SET MonthlyIncome = MonthlyIncome * 1.20
WHERE Department = 'Sales';
```

```
-- Commit the transaction
COMMIT;

select department, round(avg(monthlyincome),0) as Avg_monthly_income from employees
where department  = 'Sales'
group by department
```

Data Output | Messages | Notifications

	department character varying (255)	avg_monthly_income numeric
1	Sales	5329

13. Delete Records of Employees Who Left the Company

```
DELETE FROM Employees
WHERE Attrition = 'Yes';

Select * from Employees
where Attrition = 'Yes'
```

Suggest improvements in the database schema to reduce data redundancy and improve data integrity.

Normalization: Ensure the database follows normalization principles to minimize data redundancy and dependencies.

Foreign Keys: Use foreign keys to establish relationships, ensuring referential integrity and preventing orphaned records.

Indexes: Create indexes on frequently used columns to improve query
performance, but avoid excessive indexing.

Default Values and Constraints: Employ default values and constraints to enforce data integrity rules, reducing the risk of invalid data.

Audit Trails: Implement audit trails to track changes, providing a
historical record and enhancing accountability.

--16. Explain how you can optimize the performance of SQL queries on this dataset.

/*Here are few points for optimizing SQL queries on this dataset:

Indexing: Create indexes on columns frequently used in WHERE clauses or JOIN conditions to enhance query performance.

Limit SELECT Columns: Select only the necessary columns in your queries to reduce data transfer and improve efficiency.

Optimize WHERE Clauses: Ensure efficient WHERE clauses by avoiding functions on indexed columns and optimizing conditions.

Use JOINs Efficiently: Optimize JOIN operations by selecting the appropriate type and ensuring efficient join conditions.

Update Statistics Regularly: Keep table statistics up-to-date to assist the query planner in making informed execution plans.*/