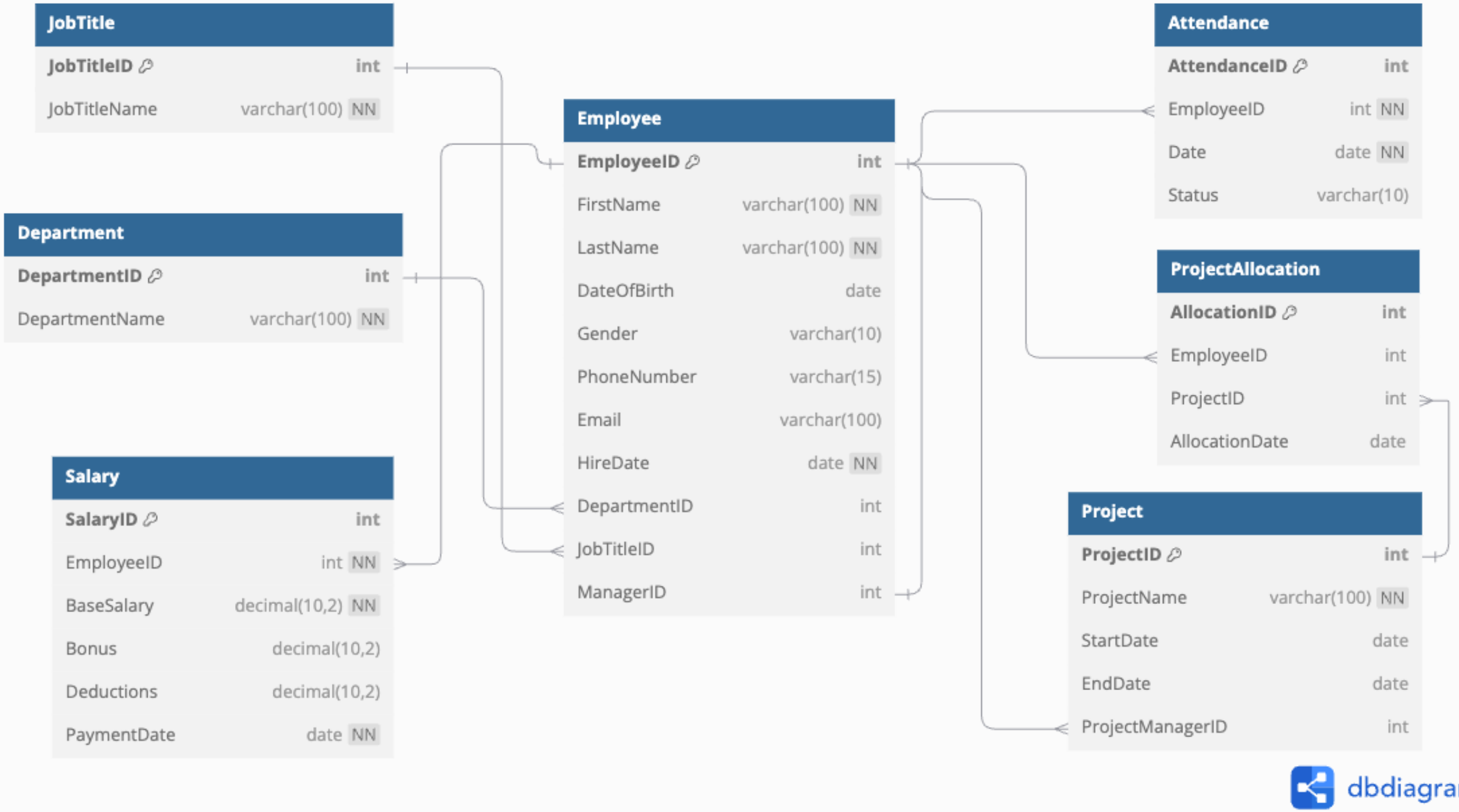
Employee Management System

In this case study, you'll design and query a database for "TechCorp," a mid-sized company aiming to manage employee data efficiently. The system should store details about employees, their departments, projects, salaries, and attendance, with additional constraints to ensure data integrity. The case study will focus on database design, SQL queries, triggers, transactions, string manipulation, and date/time functions.

CREATE TABLES WITH NORMALIZATION AND CONSTRAINTS

General Instructions :

1. **Referential Integrity**: Ensure that all foreign key relationships are defined correctly between the tables.
2. **Normalization**: Ensure that the tables are normalized (at least 3NF), i.e., no repeating groups, no partial dependencies, and no transitive dependencies.
3. **Constraints**: Apply necessary constraints like NOT NULL, UNIQUE, CHECK, and DEFAULT values as specified.
4. **Data Integrity**: Handle scenarios like managing employees without a manager (nullable ManagerID), and ensure attendance status has valid options ('Present', 'Absent', 'Leave').


JobTitle
JobTitleID int
JobTitleName varchar(100) NN
Department
DepartmentID int
DepartmentName varchar(100) NN
Salary
SalaryID int
EmployeeID int NN
BaseSalary decimal(10,2) NN
Bonus decimal(10,2)
Deductions decimal(10,2)
PaymentDate date NN
Employee
EmployeeID int
FirstName varchar(100) NN
LastName varchar(100) NN
DateOfBirth date
Gender varchar(10)
PhoneNumber varchar(15)
Email varchar(100)
HireDate date NN
DepartmentID int
JobTitleID int
ManagerID int
Attendance
AttendanceID int
EmployeeID int NN
Date date NN
Status varchar(10)
ProjectAllocation
AllocationID int
EmployeeID int
ProjectID int
AllocationDate date
Project
ProjectID int
ProjectName varchar(100) NN
StartDate date
EndDate date
ProjectManagerID int
dbdiagram.io

T1. Department Table

Question: Create a table named Department to store department information. The table should have the following columns:

- DepartmentID (Primary Key)
- DepartmentName (Name of the department, not null)

Ensure that DepartmentID is set as the primary key and DepartmentName is not allowed to be NULL.

2. Job Title Table

Question: Create a table named JobTitle to store job titles for employees. The table should include:

- JobTitleID (Primary Key)
- JobTitleName (Name of the job title, not null)

Make sure to enforce that JobTitleName is unique and not NULL.

3. Employee Table (Without Foreign Key Constraints)

Question: Create a table named Employee to store employee information.
The table should include the following columns:

- EmployeeID (Primary Key)
- FirstName (First name of the employee, not null)
- LastName (Last name of the employee, not null)
- DateOfBirth (Date of birth of the employee)
- Gender (Gender of the employee)
- PhoneNumber (Phone number of the employee, unique)
- Email (Email address of the employee, unique)
- HireDate (Hire date of the employee)
- DepartmentID (ID of the department the employee belongs to)
- JobTitleID (ID of the job title of the employee)
- ManagerID (ID of the manager of the employee, self-referencing foreign key)

Note: The ManagerID column should be nullable, allowing employees to have no manager (e.g., top-level managers).

4. Attendance Table

Question: Create a table named Attendance to track employee attendance. The table should include the following columns:

- AttendanceID (Primary Key)
- EmployeeID (ID of the employee, foreign key referencing Employee)
- Date (Date of attendance)
- Status (Attendance status: 'Present', 'Absent', or 'Leave')

Ensure that a foreign key constraint is placed on EmployeeID to reference the Employee table, and enforce that Status can only contain the values 'Present', 'Absent', or 'Leave'.

5. Salary Table

Question: Create a table named Salary to store salary details of employees. The table should include the following columns:

- SalaryID (Primary Key)
- EmployeeID (ID of the employee, foreign key referencing Employee)
- BaseSalary (Base salary of the employee, not null)
- Bonus (Bonus of the employee, default value of 0.00)
- Deduction (Deductions for the employee, default value of 0.00)
- PaymentDate (Date of salary payment)

Ensure that a foreign key constraint is placed on EmployeeID to reference the Employee table.

6. Project Table

Question: Create a table named Project to store information about projects. The table should include the following columns:

- ProjectID (Primary Key)
- ProjectName (Name of the project, not null)
- StartDate (Start date of the project)
- EndDate (End date of the project)
- ProjectManagerID (ID of the manager responsible for the project, foreign key referencing Employee)

Make sure to enforce that ProjectManagerID references the Employee table and indicates who is the manager of the project.

7. Project Allocation Table

Question: Create a table named ProjectAllocation to manage the many-to-many relationship between employees and projects. The table should include the following columns:

- AllocationID (Primary Key)
- EmployeeID (ID of the employee, foreign key referencing Employee)
- ProjectID (ID of the project, foreign key referencing Project)
- AllocationDate (Date when the employee is allocated to the project)

Ensure that:

- Foreign key constraints are placed on EmployeeID (referencing Employee) and ProjectID (referencing Project).
- The combination of EmployeeID and ProjectID should be unique to prevent an employee from being allocated to the same project more than once.