# DB Task 2 - Domain Normalization

*Submitted By,*

*Abhijith A Nair*

## Domain Selection – Recruitment Management System

The Recruitment Management System (RMS) is a vital component in the human resources landscape, designed to streamline and optimize the end-to-end process of acquiring, assessing, and onboarding new talent within an organization. In today's competitive job market, where attracting and retaining skilled professionals is crucial, an efficient RMS plays a pivotal role in ensuring that companies identify the right candidates for the right positions.

The system helps in maintaining a centralized repository of candidate data, facilitating data-driven decision-making and improving the overall hiring experience for both employers and candidates. Its relevance lies in the ability to minimize manual efforts, reduce time-to-hire, and ensure a more transparent, fair, and compliant recruitment process.

## Entities and Entity-Relationship Diagram

### Entities

1. **Role Table**

   *The table defines different roles associated with the system. Here only admin will be the role used.*

   <u>Attributes</u>

   - RoleID

- Datatype : INT
  - Constraints : PRIMARY KEY , NOT NULL
- RoleName
  - Datatype : VARCHAR(10)
  - Constraints : NOT NULL

2. **Department Table**

*The table defines different departments associated with the company.*

Attributes

- DeptID
  - Datatype : INT
  - Constraints : PRIMARY KEY , NOT NULL
- DeptName
  - Datatype : VARCHAR(20)
  - Constraints : NOT NULL
- DeptManager
  - Datatype : VARCHAR(20)
  - Constraints : NOT NULL
- NoofEmployees
  - Datatype : INT
  - Constraints : NOT NULL

3. **Candidate Table**

*The table defines the candidates the apply for job in the company.*

Attributes

- CandidateID

- Datatype : INT
- Constraints : PRIMARY KEY , NOT NULL

- CandidateName
  - Datatype : VARCHAR(20)
  - Constraints : NOT NULL

- CandidateEmail
  - Datatype : VARCHAR(20)
  - Constraints : NOT NULL

- CandidatePhone
  - Datatype : VARCHAR(10)
  - Constraints : NOT NULL

- Address
  - Datatype : VARCHAR(50)
  - Constraints : NIL

4. **JobPosition Table**

*The table defines the different job positions available in the company.*

Attributes

- JobPositionID
  - Datatype : INT
  - Constraints : PRIMARY KEY , NOT NULL

- JobPositionName
  - Datatype : VARCHAR(20)
  - Constraints : NOT NULL

5. **Level Table**

*The table defines different levels the candidates go through the interview process.*

<u>Attributes</u>

- LevelID

    - Datatype : INT

    - Constraints : PRIMARY KEY , NOT NULL

- LevelName

    - Datatype : VARCHAR(20)

    - Constraints : NOT NULL

6. **User Table**

*The table defines the login details of the system*

<u>Attributes</u>

- UserID

    - Datatype : INT

    - Constraints : PRIMARY KEY , NOT NULL

- UserName

    - Datatype : VARCHAR(20)

    - Constraints : NOT NULL

- Email

    - Datatype : VARCHAR(20)

    - Constraints : NOT NULL

- Password

    - Datatype : VARCHAR(20)

    - Constraints : NOT NULL

- UserRoleID

    - Datatype : INT

- Constraints : NOT NULL , FOREIGN KEY ( referencing Role Table )

7. **JobVacancy Table**

*The table defines the different vacancies available in departments.*

Attributes

- VacancyID
    - Datatype : INT
    - Constraints : PRIMARY KEY , NOT NULL
- DepartmentID
    - Datatype : INT
    - Constraints : NOT NULL , FOREIGN KEY ( referencing Department Table )
- JobPositionID
    - Datatype : INT
    - Constraints : NOT NULL , FOREIGN KEY ( referencing JobPosition Table )
- Date
    - Datatype : DATE
    - Constraints : NIL
- Status
    - Datatype : VARCHAR(10)
    - Constraints : NOT NULL

8. **JobApplication Table**

*The table defines the applications submitted by the clients.*

Attributes

- JobApplicationID
    - Datatype : INT

- Constraints : PRIMARY KEY , NOT NULL

- CandidateID

  - Datatype : INT

  - Constraints : NOT NULL , FOREIGN KEY ( referencing Candidate Table )

- JobVacancyID

  - Datatype : INT

  - Constraints : NOT NULL , FOREIGN KEY ( referencing JobVacancy Table )

- SubmissionDate

  - Datatype : DATE

  - Constraints : NOT NULL

- Status

  - Datatype : VARCHAR(10)

  - Constraints : NOT NULL

9. **JobLevel Table**

*The table defines the different job levels a candidate is passing through after submitting application.*

Attributes

- JobLevelID

  - Datatype : INT

  - Constraints : PRIMARY KEY , NOT NULL

- ApplicationID

  - Datatype : INT

  - Constraints : NOT NULL , FOREIGN KEY ( referencing JobApplication Table )

- LevelID

- Datatype : INT

- Constraints : NOT NULL , FOREIGN KEY ( referencing Level Table )

- Status

  - Datatype : VARCHAR(10)

  - Constraints : NOT NULL


10. **JobClearance Table**

*The table defines whether the applicant has cleared job or not.*

Attributes

- JobClearanceID

  - Datatype : INT

  - Constraints : PRIMARY KEY , NOT NULL

- JobLevelID

  - Datatype : INT

  - Constraints : NOT NULL , FOREIGN KEY ( referencing JobLevel Table )

- Status

  - Datatype : VARCHAR(10)

  - Constraints : NOT NULL

11. **CandidateUserMapping Table**

*The table defines mapping between the user and candidate.*

Attributes
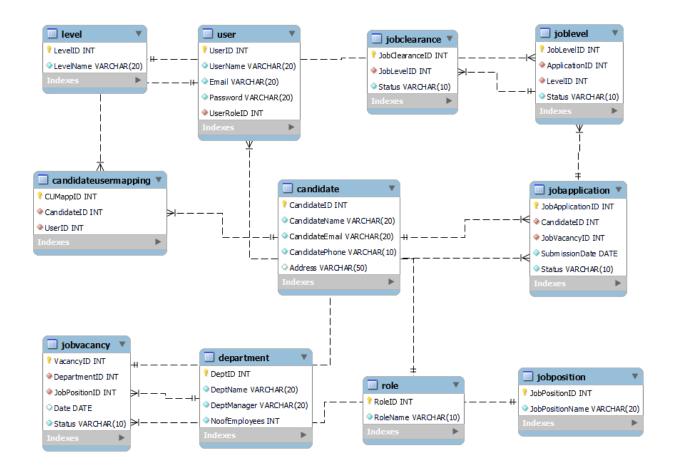
- CUMappID

  - Datatype : INT

  - Constraints : NOT NULL , PRIMARY KEY

- CandidateID

  - Datatype : INT

- Constraints : NOT NULL , FOREIGN KEY ( referencing Candidate Table )

- UserID

  - Datatype : INT

  - Constraints : NOT NULL , FOREIGN KEY ( referencing User Table )

## ER Diagram



# Normalization & Final Tables

The provided database tables, including Role, Department, Candidate, JobPosition, Level, User, JobVacancy, JobApplication, JobLevel, and JobClearance, CandidateUserMapping exhibit a well-structured and normalized design. Each table

adheres to the principles of normalization, ensuring atomic values, eliminating repeating groups, and avoiding partial or transitive dependencies. The tables are organized up to the 3rd Normal Form (3NF), signifying optimal data integrity and minimizing redundancy. The normalized structure promotes efficient data management, supports relational integrity, and enhances overall database performance.

Finally these are all the tables in the system :

### Role Table

| RoleID (PK) | RoleName |
|---|---|
| INT | VARCHAR(10) |

### Department Table

| DeptID (PK) | DeptName | DeptManager | NoofEmployees |
|---|---|---|---|
| INT | VARCHAR(20) | VARCHAR(20) | INT |

### Candidate Table

| CandidateID (PK) | CandidateName | CandidateEmail | CandidatePhone | Address |
|---|---|---|---|---|
| INT | VARCHAR(20) | VARCHAR(20) | VARCHAR(10) | VARCHAR(50) |

### JobPosition Table

| JobPositionID (PK) | JobPositionName |
|---|---|
| INT | VARCHAR(20) |

### Level Table

| LevelID (PK) | LevelName |
|---|---|
| INT | VARCHAR(20) |

## User Table

| UserID (PK) | UserName | Email | Password | UserRoleID (FK) |
|---|---|---|---|---|
| INT | VARCHAR(20) | VARCHAR(20) | VARCHAR(20) | INT |

## JobVacancy Table

| VacancyID (PK) | DepartmentID (FK) | JobPositionID (FK) | Date | Status |
|---|---|---|---|---|
| INT | INT | INT | DATE | VARCHAR(10) |

## JobApplication Table

| JobApplicationID (PK) | CandidateID (FK) | JobVacancyID (FK) | SubmissionDate | Status |
|---|---|---|---|---|
| INT | INT | INT | DATE | VARCHAR(10) |

## JobLevel Table

| JobLevelID (PK) | ApplicationID (FK) | LevelID (FK) | Status |
|---|---|---|---|
| INT | INT | INT | VARCHAR(10) |

## JobClearance Table

| JobClearanceID (PK) | JobLevelID (FK) | Status |
|---|---|---|
| INT | INT | VARCHAR(10) |

## CandidateUserMapping Table

| CUMappingID (PK) | CandidateID (FK) | UserID (FK) |
|---|---|---|
| INT | INT | INT |

**Code**

```
CREATE DATABASE RecruitmentManagementSystem;
USE RecruitmentManagementSystem;
```

- - Role Table
  ```
  CREATE TABLE Role (
  RoleID INT PRIMARY KEY NOT NULL,
  RoleName VARCHAR(10) NOT NULL
  );
  ```

- - Department Table
  ```
  CREATE TABLE Department (
  DeptID INT PRIMARY KEY NOT NULL,
  DeptName VARCHAR(20) NOT NULL,
  DeptManager VARCHAR(20) NOT NULL,
  NoofEmployees INT NOT NULL
  );
  ```

- - Candidate Table
  ```
  CREATE TABLE Candidate (
  CandidateID INT PRIMARY KEY NOT NULL,
  CandidateName VARCHAR(20) NOT NULL,
  CandidateEmail VARCHAR(20) NOT NULL,
  CandidatePhone VARCHAR(10) NOT NULL,
  Address VARCHAR(50)
  );
  ```

- - JobPosition Table
  ```
  CREATE TABLE JobPosition (
  JobPositionID INT PRIMARY KEY NOT NULL,
  JobPositionName VARCHAR(20) NOT NULL
  );
  ```

- - Level Table
  ```
  CREATE TABLE Level (
  LevelID INT PRIMARY KEY NOT NULL,
  ```

LevelName VARCHAR(20) NOT NULL

);

- - User Table

  CREATE TABLE User (

  UserID INT PRIMARY KEY NOT NULL,

  UserName VARCHAR(20) NOT NULL,

  Email VARCHAR(20) NOT NULL,

  Password VARCHAR(20) NOT NULL,

  UserRoleID INT NOT NULL,

  FOREIGN KEY (UserRoleID) REFERENCES Role(RoleID)

  );

- - JobVacancy Table

  CREATE TABLE JobVacancy (

  VacancyID INT PRIMARY KEY NOT NULL,

  DepartmentID INT NOT NULL,

  JobPositionID INT NOT NULL,

  Date DATE,

  Status VARCHAR(10) NOT NULL,

  FOREIGN KEY (DepartmentID) REFERENCES Department(DeptID),

  FOREIGN KEY (JobPositionID) REFERENCES JobPosition(JobPositionID)

  );

- - JobApplication Table

  CREATE TABLE JobApplication (

  JobApplicationID INT PRIMARY KEY NOT NULL,

  CandidateID INT NOT NULL,

  JobVacancyID INT NOT NULL,

  SubmissionDate DATE NOT NULL,

  Status VARCHAR(10) NOT NULL,

  FOREIGN KEY (CandidateID) REFERENCES Candidate(CandidateID),

  FOREIGN KEY (JobVacancyID) REFERENCES JobVacancy(VacancyID)

  );

- - JobLevel Table

  CREATE TABLE JobLevel (

  JobLevelID INT PRIMARY KEY NOT NULL,

```
ApplicationID INT NOT NULL,
LevelID INT NOT NULL,
Status VARCHAR(10) NOT NULL,
FOREIGN KEY (ApplicationID) REFERENCES JobApplication(JobApplicationID),
FOREIGN KEY (LevelID) REFERENCES Level(LevelID)
);
```

- - JobClearance Table
```
CREATE TABLE JobClearance (
JobClearanceID INT PRIMARY KEY NOT NULL,
JobLevelID INT NOT NULL,
Status VARCHAR(10) NOT NULL,
FOREIGN KEY (JobLevelID) REFERENCES JobLevel(JobLevelID)
);
```

```
CREATE TABLE CandidateUserMapping (
CUMappID INT NOT NULL PRIMARY KEY,
CandidateID INT NOT NULL,
UserID INT NOT NULL,
FOREIGN KEY (CandidateID) REFERENCES Candidate(CandidateID),
FOREIGN KEY (UserID) REFERENCES User(UserID)
);
```