Technical Report: Taha Employee Management Database

1. Introduction

This technical report details the design, creation, and usage of an SQL database for an Employee Management System. The database is designed to manage employee information, departmental structure, project assignments, and authorization roles within an organization.

2. Analysis of Organization's Needs, Users, and Entities

2.1. Organization's Needs

The primary needs for an employee management system include:

- **Employee Management**: The ability to add, update, and manage employee information including personal details, job titles, salaries, and departmental assignments.
- **Department Organization**: Managing the organizational structure through departments.
- **Project Tracking**: Assigning employees to various projects and tracking project details including timelines and budgets.
- **Role-based Access Control**: Different user roles with appropriate permissions to maintain data security and integrity.
- **Resource Allocation**: Tracking which employees are assigned to which projects for workload management.
- **Budget Management**: Monitoring project budgets and employee salaries.
- **Reporting Capabilities**: The ability for authorized users to generate reports on employee distribution, project status, and departmental performance.

2.2. Identified Users

Two primary categories of users were identified for this system:

1. **HR Users**: These users manage employee and department information:
   - Adding and updating employee records
   - Managing departmental structures
   - Overseeing salary information
   - Handling employee transfers between departments
2. **Project Management Users**: These users focus on project-related activities:
   - Creating and updating project information
   - Assigning employees to projects
   - Monitoring project timelines and budgets
   - Accessing employee information for project staffing decisions

2.3. Identified Entities and Key Attributes

Based on the needs analysis and user identification, the following core entities were established:

- **Departments**: Stores organizational division information.
  - Attributes: department_id (PK), department_name
- **Employees**: Contains comprehensive employee information.
  - Attributes: employee_id (PK), first_name, last_name, birth_date, email, phone_number, department_id (FK), job_title, salary
- **Projects**: Manages all project-related information.
  - Attributes: project_id (PK), project_name, start_date, end_date, budget
- **EmployeeProjects**: Junction table for the many-to-many relationship between employees and projects.
  - Attributes: employee_id (FK), project_id (FK), assigned_date

3. Database Design and Relationships

3.1. Entity-Relationship Diagram Description

The ER diagram for this database includes the entities identified in the analysis phase, along with their primary attributes and the relationships connecting them:

- **Departments and Employees**: A one-to-many relationship. One department can have multiple employees, but each employee belongs to exactly one department.
- **Employees and Projects**: A many-to-many relationship, resolved through the EmployeeProjects junction table. An employee can be assigned to multiple projects, and a project can have multiple employees working on it.

3.2. Relational Schema

The detailed relational schema, including table names, column names, data types, primary keys (PK), foreign keys (FK), and other constraints is as follows:

1. **Departments Table**:
   - department_id SERIAL, PK
   - department_name VARCHAR(100), NOT NULL, UNIQUE
2. **Employees Table**:
   - employee_id SERIAL, PK
   - first_name VARCHAR(50), NOT NULL
   - last_name VARCHAR(50), NOT NULL
   - birth_date DATE, NOT NULL
   - email VARCHAR(100), NOT NULL, UNIQUE
   - phone_number INT, NOT NULL
   - department_id INT, NOT NULL, FK REFERENCES Departments(department_id)

- job_title VARCHAR(100)
- salary NUMERIC(10,2), NOT NULL

3. **Projects Table**:
   - project_id SERIAL, PK
   - project_name VARCHAR(100), NOT NULL, UNIQUE
   - start_date DATE
   - end_date DATE
   - budget NUMERIC(15,2)

4. **EmployeeProjects Table**:
   - employee_id INT, NOT NULL, FK REFERENCES Employees(employee_id)
   - project_id INT, NOT NULL, FK REFERENCES Projects(project_id)
   - assigned_date DATE, NOT NULL
   - PRIMARY KEY (employee_id, project_id)

4. Data Security and Access Control

The database implements role-based access control to ensure that users can only access the data they need to perform their functions:

1. **HR User Role (hr_user)**:
   - Has full CRUD (Create, Read, Update, Delete) permissions on the Departments table
   - Has full CRUD permissions on the Employees table
   - Designed for human resources personnel to manage employee and department information

2. **Project Management User Role (pm_user)**:
   - Has read-only access to the Employees table

- Has read, create, and update permissions on the Projects table
- Has read and create permissions on the EmployeeProjects table
- Designed for project managers to create projects and assign employees to them

5. Sample Data and Usage Examples

5.1. Data Overview

The database is populated with sample data including:

- 10 departments
- 10 employees distributed across various departments
- 10 projects with varying budgets and timelines
- 12 employee-project assignments

5.2. Sample Queries

The following sample queries demonstrate typical usage patterns:

1. **Finding Engineers**: Search for employees whose job title contains 'Engineer'

```sql
SELECT employee_id, first_name, last_name, job_title
FROM Employees
WHERE job_title ILIKE '%Engineer%'
ORDER BY last_name ASC;
```

2. **Department Salary Analysis**: Find average salary per department

```sql
SELECT d.department_name, AVG(e.salary) AS average_salary
FROM Employees e
JOIN Departments d ON e.department_id = d.department_id
GROUP BY d.department_name
ORDER BY average_salary DESC;
```

3. **High-Budget Projects**: Retrieve projects with budget greater than 500,000

```sql
SELECT project_name, budget
FROM Projects
WHERE budget > 500000
ORDER BY budget DESC;
```

4. **Project Staffing Analysis**: Count number of employees assigned to each project

```sql
SELECT p.project_name, COUNT(ep.employee_id) AS num_employees
FROM Projects p
LEFT JOIN EmployeeProjects ep ON p.project_id = ep.project_id
GROUP BY p.project_name
ORDER BY num_employees DESC;
```

5. **Recent Hires**: List employees hired after 2015

```sql
SELECT first_name, last_name, hire_date
FROM Employees
WHERE hire_date > '2015-01-01'
ORDER BY hire_date ASC;
```

6. Conclusion

The Taha Employee Management Database provides a comprehensive solution for managing human resources and project information within an organization. Its normalized design ensures data integrity while the role-based permissions maintain security. The system successfully addresses the initial requirements for employee management, departmental organization, project tracking, and appropriate access control.

With this database structure in place, the organization can effectively manage its workforce, track project progress, and maintain appropriate access controls for different user types.