Design for Human Resources (HR) Database System

**1.0 Executive Summary**

The objective of this proposal is to provide guidelines that can be used in the development, acquisition, and maintenance of a Human Resources (HR) database system. The proposal defines the basic components of the HR database system to be the database, including the entities and their relationship between one another. The database structure is described through the creation of an ER diagram (ERD) to understand the relationship between the entities in the HR system. The ERD visualizes the major entities in the scope of our HR database and describes the relationships among these entities to show the full picture of the entire database schema and to show the Key Constraints. The proposal shows how the ERD is translated into a Relational Database Management Systems to understand the conditions and the constraints captured by the relational model. The relational database is produced to understand how the entities and the relationships in the database can be translated into sets of tables with columns and rows. It shows how each row in a table is marked with primary keys, and how rows among multiple tables can be made related using foreign keys. The foreign keys link between the different entities and relationships in different tables. 10 sample queries are proposed to show the creation of the database tables and refer to the action of retrieving data from the database.

The team decided to work on the HR database management system to understand how the HR functions and what data is stored in the databases. It helps us to understand how the employee's data are stored, and how they are assigned to projects and what are the conditions that make specific employees get selected for specific projects. It also helps us to understand how HR manage the benefits administration. This proposal can lead to future projects involving more extensive mapping and the creation of a bigger database management system that includes multiple entities with a huge number of data.

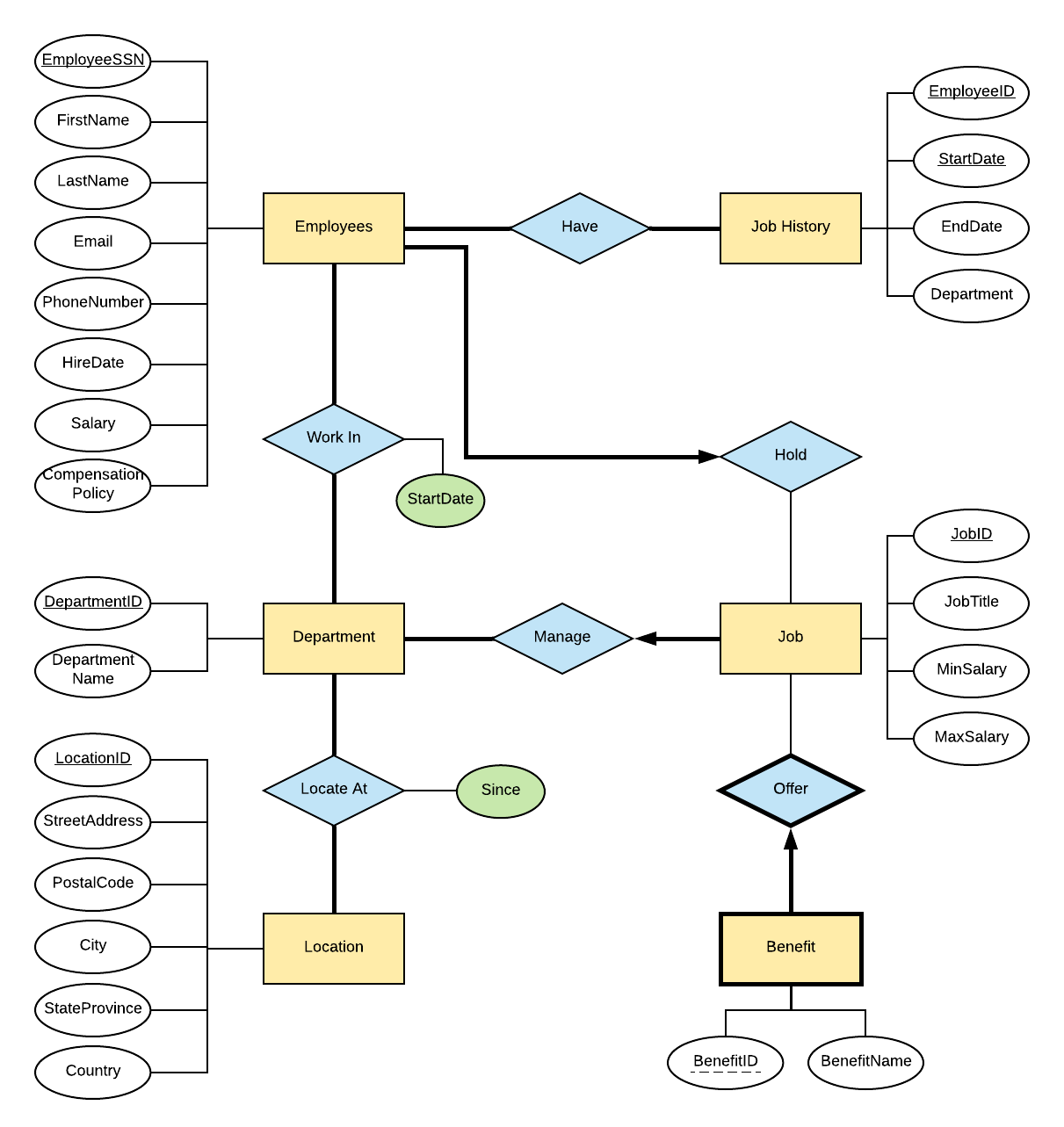
**2.0 Sample Queries**

|  |  |  |
| --- | --- | --- |
| **Query No.** | **Query Description** | **SQL Query** |
| **1** | What are all the instances within the table Employees? | SELECT \* FROM Employees |
| **2** | Return the first names, last names, and the associated job titles for all the employees from the Employees and Job tables | SELECT E.FirstName, E.LastName, J.JobTitle  FROM Employees E, Job J |
| **3** | Return the job titles and the associated maximum salary amount from the Job table and order the results in a descending order from the Job table. | SELECT DISTINCT JobTitle, MaxSalary  FROM Job  ORDER BY MaxSalary DESC |
| **4** | What is the number of the distinct countries contained in the Location table? | SELECT COUNT( DISTINCT Country )  FROM Location |
| **5** | Insert a new instance in the Employees table with EmployeeID 10041004, First name Emily, Last name Awesome, and Hire date of Feb. 05, 2019 (all plain string) | INSERT INTO Employees (EmployeeID, FirstName, LastName, HireDate)  VALUES (‘10041004’, ‘Emily’, ‘Awesome’, ‘2019-02-05’) |
| **6** | Insert a new instance in the Benefit table with Benefit ID of abc123, and benefit description of “$1M bonus” | INSERT INTO Benefit  VALUES (‘abc123’, ‘$1M bonus’) |
| **7** | Delete employee with ID number equals to ‘10041004’ from the Employees table. | DELETE FROM Employees  WHERE EmployeeID = ‘10041004’ |
| **8** | Create a temporary view called departmentView that contains information of Department ID, Department Name, and City Name | CREATE VIEW departmentView AS SELECT D.DepartmentID, D.DepartmentName, L.City  FROM Department D, Location L |
| **9** | Show all the information contained in the departmentView view | SELECT \* FROM departmentView |
| **10** | Delete the departmentView view if it exists | DROP VIEW IF EXISTS departmentView |
| **11** | Create a new table called Benefit with columns of BenefitID (instance value must be 10 characters and does not contain a null value) and BenefitName (20 characters). The primary key of the table shall be BenefitID and JobID, with JobID also being a foreign key referencing the Job table. If an instance is being deleted from the Job table, the associated record shall be deleted from the Benefit table as well. | CREATE TABLE Benefit (BenefitID CHAR(10) NOT NULL, BenefitName CHAR(20)  PRIMARY KEY (BenefitID, JobID),  FOREIGN KEY (JobID) REFERENCES Job,  ON DELETE CASCADE) |
| **12** | Delete any employees with the name David from the Employees table. | DELETE FROM Employees E WHERE E.name = ‘David’ |
| **13** | Insert a new instance in the Employees Table with Last name David, First name Jones, and Salary of 70000. | INSERT INTO Employees (Lastname, Firstname, Salary)  VALUES (‘David’, 'Jones',70000) |

**3.0 Data Requirements**

1. Every Employee must have at least one Job History.
2. Every Job History must be associated with at least one employee.
3. Every employee must work in at least one department.
4. Every department must be worked by at least one employees.
5. Every employee must have exactly one job.
6. Some Jobs had by some employees.
7. Every department must manage at least one job.
8. Every job must be managed by exactly one department.
9. Every department must be located at one or more locations.
10. Every location must locate at least one department.
11. Some jobs offer some benefits. And every benefit must be offered by exactly one job. If a job is deleted, you should not keep track of its benefits any longer.

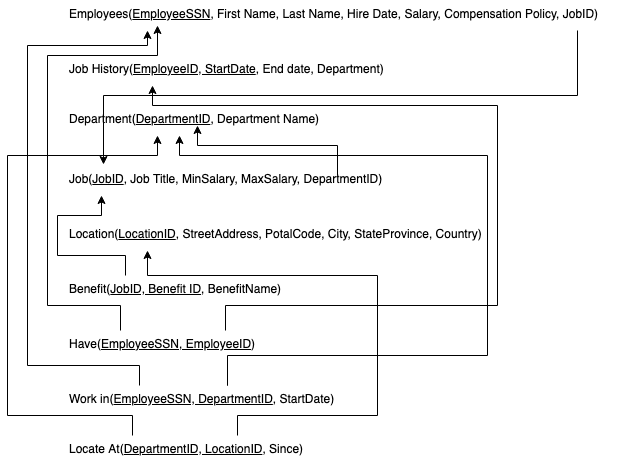
**4.0 The ER Diagram**



**The ER model cannot capture certain general constraints. For example:**

1. An employee’s salary must be greater than or equal to the MinSalary and less than or equal to the MaxSalary of the corresponding job title from the Job entity.
2. The StartDate that an employee starts working in a department must be greater than or equal to the JobHistory.EndDate associated with the latest instance identified by JobHistory.EmployeeID and JobHistory.StartDate.
3. JobHistory.StartDate must be smaller than the JobHistory.EndDate by at least one day.

**5.0 Translation of the ER Diagram to Relations Model**



**5.1 Constraints**

Employees(**EmployeeSSN**, First Name, Last Name, Hire Date, Salary, Compensation Policy, JobID)

**Key constraint**: JobID is a foreign key referencing JOB

**Participation constraint**: employees need total participation in relationship HAVE; employees; need total participation in relationship WORK\_IN

Job History(**EmployeeID**, **StartData**, End date, Department)

**Participation constraint**: Job History need total participation in relationship HAVE;

Department(**DepartmentID**, Department Name)

**Participation constraint**: department need total participation in relationship WORK\_IN; MANAGE; LOCATE\_AT

Job(**JobID**, Job Title, MinSalary, MaxSalary, DepartmentID)

**Key constraint:** DepartmentID is a foreign key referencing Department

**Participation constraint**: Job need total participation in relationship MANAGE

Benefit(**JobID, Benefit ID**, BenefitName)

**Key constraint:** JobID is a foreign key referencing JOB

**Participation constraint**: Benefit need total participation in relationship OFFER

Have(**EmployeeSSN, EmployeeID**)

**Key constraint:** EmployeeSSN is a foreign key referencing Employee; EmloyeeID is a foreign key referencing Job History

Work in(**EmployeeSSN, DepartmentID**, StartDate)

**Key constraint:** EmployeeSSN is a foreign key referencing Employee; Department ID is a foreign key referencing Department

Locate At(**DepartmentID, LocationID**, Since)

**Key constraint:** Department ID is a foreign key referencing Department; LocationID is a foreign key referencing Location

Location(**LocationID**, StreetAddress, PotalCode, City, StateProvince, Country)

**Participation constraint**: Location need total participation in relationship LOCATE\_AT