**International Employee Immigration Data Management**

Milestone: Project Report

Group 4

Amogha Gadde  
Abhishek Kumar Sinha

857-693-9502   
857-961-6419

gadde.am@northeastern.edu  
sinha.ab@ northeastern.edu

Percentage of Effort Contributed by Amogha: 50%

Percentage of Effort Contributed by Abhishek: 50%

Signature of Student 1: Amogha Gadde

Signature of Student 2: Abhishek Kumar Sinha

Submission Date: December 10, 2023

Table of Contents

[Executive Summary:](#_Toc153129935) 1

[I. Introduction 2](#_Toc153129936)

[II. Conceptual Data Modeling 3](#_Toc153129937)

[1. EER Diagram 3](#_Toc153129938)

[2. UML Diagram 4](#_Toc153129939)

[III. Mapping Conceptual Model to Relational Model 5](#_Toc153129940)

[IV. Implementation of Relation Model via MySQL and NoSQL 6](#_Toc153129941)

[A. MySQL Implementation: 6](#_Toc153129942)

[1. NoSQL Implementation: 8](#_Toc153129943)

[V. Database Access via Python 9](#_Toc153129944)

[VI. Summary and Recommendation 10](#_Toc153129945)

**Executive Summary:**

The main goal of this study was to devise and deploy a practical relational database tailored for use in the International Employee Immigration domain, addressing the persistent challenges faced by companies in this industry. These companies have consistently voiced concerns about the laborious nature of the mandatory international employee data input procedures mandated by law. Upon a comprehensive examination of the industry-wide issue, it became evident that there was substantial data redundancy, presenting an opportunity for efficiency through data reuse facilitated by a relational database.

The implementation of this relational database has demonstrated a 50% reduction in the time required for the data input process, yielding significant cost savings across the entire industry. Additionally, the database incorporates a central analytics platform with extensive potential for analyzing employee information, studying citizenship patterns, and streamlining data management.

The database design considered the specific data fields mandated by the United States Citizenship and Immigration Services for VISA and Green Card renewal processes. Extensive Entity-Relationship (EER) and Unified Modeling Language (UML) diagrams were crafted, followed by the translation of the conceptual model into a relational model, complete with the necessary primary and foreign keys. The full implementation of this database was executed using MySQL, with a prototype adapted for MongoDB to explore the feasibility of employing this database in a NoSQL environment.

The resulting database has proven to be highly successful. Integration with Python has unlocked expansive analytics capabilities, some of which are showcased in the study. These analytical queries can be instrumental in monitoring and managing employees effectively. The subsequent phase of enhancement for this SQL Database involves the incorporation of Data Governance measures. Following this, the project aims to be presented to U.S. companies seeking simplified solutions for international employee management processes.

**I. Introduction**

A product centered, company X necessitates access to its employee immigration data to fulfill pivotal functions related to visa and green card sponsorship. This ensures strict compliance with legal standards and proficient navigation through the intricate immigration processes. Leveraging this database allows the organization to adeptly handle work permit visas, green card sponsorships, and visa renewals. This proactive strategy not only shields the company from potential legal complexities and penalties but also fosters stability and continuity in its workforce. This, in turn, empowers the company to retain its valuable talent pool, significantly contributing to its growth and success within the United States.

Moreover, the meticulous upkeep of accurate immigration records guarantees that each employee possesses the full authorization to work in the U.S., establishing a more dependable and compliant staff.

**Framework for Managing International Employee Immigration Data:**

Company X comprises seven distinct departments: Software Development, Data Science, Finance & Accounting, Human Resources, Research and Development, Customer Support and Service, and Product Design & User Experience. Employees can be assigned to any department based on their expertise. The organization diligently maintains essential employee details, encompassing visa types, green card availability, I-9 forms, employment history, contact information, compensation details, and passport particulars, adopting a comprehensive approach.

For each **employee**, the company needs to record the employee ID, first name, last name, date of birth, Nationality, social security number (SSN), gender, age, Job title, Department.

For **Passport information**, company needs employee id, nationality, passport number, passport expiration date, passport issued date.

For **visa type**, their visa id, employee id, visa type name, visa expiration date, visa issued date will be recorded.

For **Green card availability**, company need to store the green card id, employee id, green card status, green card expiration date, green card issued date.

For **I-9 form**, company needs to save I-9 form id, employee id, date of form completion, document type, document number, document expiration date, e - verify result, status.

For **employment history**, employee id, work ex, previous company name, previous company location, previous job title, last working day.

For **contact information**, company needs to save employee id, phone number, address, email id.

For **salary information**, company needs to save employee id, department name, base salary, visa id.

**The process of managing work permit VISA and Green card sponsorship** needs to be recorded. Company needs to store the employee ID, first name, last name, phone number, email id, Nationality, visa id, base salary, green card id.

**The process of VISA renewal process should be recorded**. First, company need to know the employee id, first name, last name, visa type, visa id, visa expiration date, visa issued date, phone number, email id, nationality, passport number.

# **II. Conceptual Data Modeling**

1. EER Diagram

A diagram of a company

Description automatically generated

d

## 2. UML Diagram

A diagram of a company

Description automatically generated

# 

# **III. Mapping Conceptual Model to Relational Model**

Primary key – **Underlined** | Foreign Key – ***Italicized***

1. Employee (**ID\_emp**, DOB, SSN, Name, Gender, ID\_dept, Job title, Age)
2. Contact (**Phone\_No**, Email\_ID, Address, ***ID\_emp***)
3. Works\_in (**Project\_ID**, **ID\_emp**, Hours)

*FOREIGN KEY* ***Project\_ID*** *refers to Project\_ID in Project; NULL ALLOWED*

*FOREIGN KEY* ***ID\_emp*** *refers to ID\_emp in Employee; NULL NOT ALLOWED*

1. Project (**Project\_ID**, Project\_start\_date, Project\_end\_date, Project\_name, Duration)
2. Belongs\_to (**Dep\_ID**, **Project\_ID**)

*FOREIGN KEY* ***Project\_ID*** *refers to Project\_ID in Project; NULL NOT ALLOWED*

*FOREIGN KEY* ***Dep\_ID*** *refers to Dep\_ID in Department; NULL NOT ALLOWED*

1. Department (**Dep\_ID**, Dep\_Name)
2. Previous\_works\_experience (**ID emp**, Num\_years\_exp, Prev\_Job\_title, Last\_working\_date, Prev\_company\_loc, Previous\_company\_name)
3. Passport (**Passport\_No**, Nationality, Passport\_exp\_date, Passport\_issue\_date, ***ID\_emp***)

*FOREIGN KEY* ***ID\_emp*** *refers to ID\_emp in Employee; NULL NOT ALLOWED*

1. I9-Form (**Form\_ID**, Document\_type, Form\_date\_completion, Form\_status, e\_verify\_result, Doc\_exp\_date, Doc\_No, ***ID\_emp***)

*FOREIGN KEY* ***ID\_emp*** *refers to ID\_emp in Employee; NULL NOT ALLOWED*

1. Green\_Card(**GreenCard\_ID**, GreenCard\_status, GreenCard\_exp\_date, GreenCard\_issue\_date, ***ID\_emp***)

*FOREIGN KEY* ***ID\_emp*** *refers to ID\_emp in Employee; NULL NOT ALLOWED*

1. Visa (**Visa\_ID**, Visa\_type, Visa\_exp\_Date, Visa\_Issue\_Date, ***ID\_emp***)

*FOREIGN KEY* ***ID\_emp*** *refers to ID\_emp in Employee; NULL NOT ALLOWED*

1. Budget (**ID\_emp**, Total\_Acc\_amt)
2. Visa\_Sponshership(**ID\_emp**, Legal\_Fee, Form\_Fill\_Fee)
3. On-Boarding (**ID\_emp**, Onboarding\_cost, Relocation\_cost)
4. Salary (**ID\_emp**, Base\_salary)

# **IV. Implementation of Relation Model via MySQL and NoSQL**

1. MySQL Implementation:

**A screenshot of a computer

Description automatically generated**

1. **List the department names along with the corresponding department IDs**

Select Dept\_ID, Dept\_Name

From department;

1. **A screenshot of a computer

   Description automatically generatedFind the total number of employees in each department. Display the department name and the count of employees**

Select d.Dept\_Name, COUNT(e.ID\_emp) AS Employee\_Count

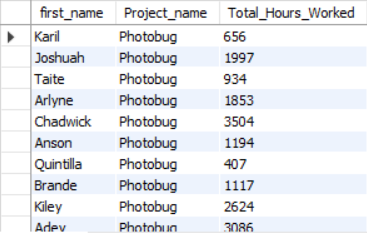
From department d

LEFT JOIN employee e ON d.Dept\_ID = e.ID\_Dept

GROUP BY d.Dept\_Name;

1. **Retrieve the names of employees who have worked on projects for more than 100 hours. Include the project names and the total hours worked**

SELECT e.first\_name, p.Project\_name, SUM(w.Hours) AS Total\_Hours\_Worked

 FROM employee e

JOIN works\_in w ON e.ID\_emp = w.ID\_emp

JOIN Project p ON w.Project\_ID = p.Project\_ID

GROUP BY e.first\_name, p.Project\_name

HAVING SUM(w.Hours) > 100;

1. **Retrieve the names of employees who have more than twice the average number of hours worked**

SELECT first\_name, last\_name

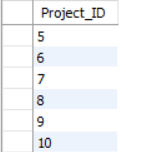
FROM employee E

WHERE ((SELECT AVG(Hours)

FROM works\_in

WHERE ID\_Emp = E.ID\_emp) <

2 \* (SELECT AVG(Hours) FROM works\_in));

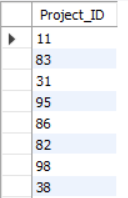
1. **Find the projects where the total hours worked by any employee exceed the total hours worked by employee with ID 18-0400226**

SELECT DISTINCT Project\_ID

FROM works\_in

WHERE Hours > ANY (SELECT Hours FROM works\_in WHERE ID\_emp = '18-0400226');

1. **Retrieve a list of unique project IDs from the Works\_in table and Belongs\_to table**

****

SELECT Project\_ID

FROM works\_in

UNION

SELECT Project\_ID

FROM Belongs\_to;

1. **Retrieve the employee names along with the percentage of the total hours they have worked compared to the overall total hours worked by all employees**

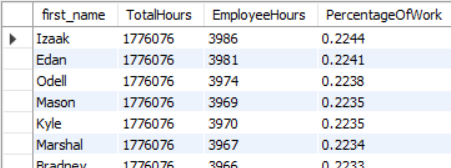
SELECT

first\_name,

(SELECT SUM(Hours) FROM works\_in) AS TotalHours,

(SELECT SUM(Hours) FROM works\_in WHERE ID\_emp = employee.ID\_emp) AS EmployeeHours,

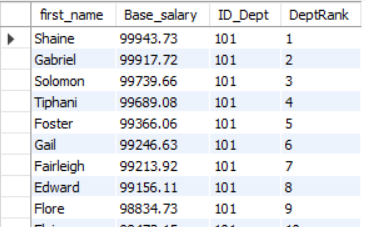
(SELECT SUM(Hours) FROM works\_in WHERE ID\_emp = employee.ID\_emp) / (SELECT SUM(Hours) FROM works\_in) \* 100 AS PercentageOfWork

FROM

employee

ORDER BY PercentageOfWork DESC;

1. **Rank employees based on their salary within each department**

****

SELECT

e.first\_name,

s.Base\_salary,

e.ID\_Dept,

RANK() OVER (PARTITION BY e.ID\_Dept

ORDER BY s.Base\_salary DESC) AS DeptRank

FROM

employee e, Salary s

WHERE e.ID\_Emp = s.ID\_Emp;

1. NoSQL Implementation:
2. **Retrieve all employees with their basic information (ID, First name, Last name, Job title, Age, DOB, SSN)**

* db.employee.find({}, { first\_name: 1, last\_name: 1, Job\_Title: 1, Age: 1, Gender: 1, SSN: 1, DOB: 1, Age: 1})

A screenshot of a computer

Description automatically generated

1. **Calculate the total hours worked by each employee in all projects.**

* db.works\_in.aggregate([{ $group: { \_id: "$ID\_Emp", totalHours: { $sum: "$Hours" } } }])

A screenshot of a computer

Description automatically generated

1. **Find employees who have worked on a project for more than 100 hours.**

* db.works\_in.find({ "Hours": { $gt: 100 } }, { "ID\_emp": 1 })

A screenshot of a computer

Description automatically generated

1. **Find Employees with the Highest Salary**

* db.salary.aggregate([

A computer screen shot of a computer code

Description automatically generated{ $sort: { "Base\_salary": -1 } },

{ $limit: 1 },

{ $lookup: {

from: "Employee",

localField: "ID\_Emp",

foreignField: "ID\_Emp",

as: "employee\_info" } },

{ $project: {\_id: 0 employee\_info: { $arrayElemAt: ["$employee\_info", 0] }, base\_salary: "$Base\_salary" } },

{ $project: {

employee\_name: { $concat: ["$employee\_info.first\_name"," ","$employee\_info.last\_name" ] },

base\_salary: 1 } } ]);

1. **Retrieve a list of the top 30 employees with the highest total budget utilization, including their full names and the corresponding total budget**

* db.budget.aggregate([

{ $match: {

$and: [

{ "Total\_Acc\_amt\_$": { $ne: "" } },

{ "Total\_Acc\_amt\_$": { $type: "double" } } ] } },

{ $addFields: { Total\_Acc\_amt\_numeric: { $toDouble: "$Total\_Acc\_amt\_$" } } },

{ $group: {

A screenshot of a computer program

Description automatically generated\_id: "$ID\_Emp",

total\_budget: { $sum: "$Total\_Acc\_amt\_numeric" } } },

{ $sort: { "total\_budget": -1 } },

{ $limit: 30 },

{ $lookup: {

from: "Employee",

localField: "\_id",

foreignField: "ID\_Emp",

as: "employee\_info" } },

{ $unwind: "$employee\_info" },

{ $project: {

\_id: 0,

employee\_name: { $concat: ["$employee\_info.first\_name", " ", "$employee\_info.last\_name"] },

total\_budget: "$total\_budget" } } ] );

# V. Database Access via Python

Access to the database is facilitated through Python, leveraging the mysql.connector for MySQL connectivity. The process involves executing queries with cursor.execute and retrieving results using fetchall. Subsequently, the obtained data is transformed into a dataframe using the pandas library. For the visualization of analyzed data, seaborn and matplotlib are employed to craft informative and visually appealing graphs

A graph with a red bar and a white box

Description automatically generated with medium confidence

**Fig: Distribution of Green Cards and Visas Across Employee Nationalities (Countries)**

A graph of a graph of a number of columns

Description automatically generated with medium confidence

**Fig: Stacked Bar Chart of Partial Budget Components**

A graph of different colored bars

Description automatically generated

**Fig: Stacked Bar Chart of Total Budget Components**

VI**. Summary and Recommendation**

The MySQL-based International Employee Immigration database is a fully operational relational database that is ready for industry use. It can be seamlessly integrated into U.S. companies seeking streamlined solutions for managing international employees. Implementing this database promises significant cost savings in the employee data input process and offers robust analytics capabilities, a fraction of which is exemplified in this report through Python.

To enhance the database, the introduction of data governance measures is crucial for ensuring data quality. Employing Python's Passport-Eye package for Machine-Readable Zone extraction from passports presents an option to enforce data governance, particularly focusing on maintaining the quality of the primary key and attributes such as Name, Date of Birth, and Passport Expiry Date.

However, a limitation lies in the NoSQL implementation of this database on MongoDB. Further investigation is warranted to explore how this unique relational database can be effectively translated into a NoSQL environment. Despite some initial tables being implemented on the MongoDB database and leveraging map-reduction for querying, there is potential for broader use in constructing the database