**Mục lục**

[I. Giới thiệu 1](#_Toc154577556)

[1. Mô tả bài toán 1](#_Toc154577557)

[2. Mục tiêu 2](#_Toc154577558)

[II. Thực hành 2](#_Toc154577559)

[Bài 1: Thiết kế class diagram và implement code 2](#_Toc154577560)

[**1.** **Class diagram** 2](#_Toc154577561)

[**2.** **Implement code** 3](#_Toc154577562)

[Bài 2: Xây dựng database và thực hiện truy vấn theo yêu cầu 4](#_Toc154577563)

[1. Xác định yêu cầu 4](#_Toc154577564)

[2. Phân tích yêu cầu 5](#_Toc154577565)

[3. Lược đồ quan hệ và mô tả dữ liệu đích 5](#_Toc154577566)

[**6.1 Employee** 6](#_Toc154577567)

[**6.2 Role** 6](#_Toc154577568)

[**6.3 Department** 6](#_Toc154577569)

[**6.4 Employee\_Role** 6](#_Toc154577570)

[**6.5 Employee\_Department** 7](#_Toc154577571)

[4. Thực hiện yêu cầu (SQL) 7](#_Toc154577572)

[5. Link github dự án 14](#_Toc154577573)

[III. Tài liệu tham khảo 15](#_Toc154577574)

# Giới thiệu

## Mô tả bài toán

**Bài 1:** Một hệ thống ETL dữ liệu cần lấy data từ 3 nguồn: MySQL, MongoDB, DynamoDB và trong tương lai còn lấy thêm từ nhiều cơ sở dữ liệu khác nữa

Yêu cầu: Sử dụng kiến thức Lập trình hướng đối tượng, design pattern, để:

* Thiết kế class diagram cho hệ thống trên
* Viết code implement, tối thiểu phải có chức năng kết nối, lấy dữ liệu. Chụp kết quả cách thực thi code và kết quả dữ liệu lấy được

**Bài 2**: Một doanh nghiệp có một cơ sở dữ liệu quan hệ như sau:

Employee (ID, Name, Nationality, DOB, Phone Number, Employee ID of Manager, Salary, Onboard Date)

Role (ID, Role Name)

Employee\_Role (Employee ID, Role ID, Start Date)

Department (ID, Name)

Employee\_Department (Employee ID, Department ID, Start date, End Date)

Yêu cầu: sử dụng SQL để:

1. Tạo bảng, khởi tạo dữ liệu

2. Lập danh sách nhân viên, phòng ban, vị trí công việc của từng nhân viên ở thời điểm hiện tại

3. Tìm phòng ban có nhiều nhân viên nhất.

4. Tìm nhân viên làm việc cho nhiều phòng ban nhất.

5. Tìm tất cả những nhân viên mà làm chung ít nhất 1 phòng ban với nhân viên tên John Doe

6. Tính mức lương trung bình theo từng phòng ban.

7. Tìm tất cả những nhân viên có làm việc chung với nhân viên John Doe tại tất cả phòng ban.

8. Tìm nhân viên có nhiều cấp dưới nhất

9. Lập danh sách nhân viên và vị trí công việc mà nhân viên đã làm ngay trước khi làm ở vị trí hiện tại (Ví dụ: nhân viên John Doe có quá trình làm việc qua các vị trí RoleA → RoleB → RoleC. Hiện tại đang làm RoleC. Kết quả cần trả về RoleB)

10. Xóa dup dữ liệu bảng Employee\_Department

11. Giả sử doanh nghiệp trên sử dụng cơ sở dữ liệu MySQL. Viết câu lệnh để tạo mới một user và cấp quyền read-all cho user đó, không cấp quyền chỉnh sửa dữ liệu. Với mỗi câu SQL, chụp ảnh kết quả trả về

## Mục tiêu

Với deadline trong vòng 1 ngày, hoàn thành các yêu cầu của 2 bài toán trên

# Thực hành

## Bài 1: Thiết kế class diagram và implement code

### **Class diagram**

A diagram of a computer

Description automatically generated with medium confidence

### **Implement code**

Github:

Kết quả:

A screenshot of a computer program

Description automatically generated

## Bài 2: Xây dựng database và thực hiện truy vấn theo yêu cầu

## Xác định yêu cầu

1. Tạo bảng, khởi tạo dữ liệu

2. Lập danh sách nhân viên, phòng ban, vị trí công việc của từng nhân viên ở thời điểm hiện tại

3. Tìm phòng ban có nhiều nhân viên nhất.

4. Tìm nhân viên làm việc cho nhiều phòng ban nhất.

5. Tìm tất cả những nhân viên mà làm chung ít nhất 1 phòng ban với nhân viên tên John Doe

6. Tính mức lương trung bình theo từng phòng ban.

7. Tìm tất cả những nhân viên có làm việc chung với nhân viên John Doe tại tất cả phòng ban.

8. Tìm nhân viên có nhiều cấp dưới nhất

9. Lập danh sách nhân viên và vị trí công việc mà nhân viên đã làm ngay trước khi làm ở vị trí hiện tại (Ví dụ: nhân viên John Doe có quá trình làm việc qua các vị trí RoleA → RoleB → RoleC. Hiện tại đang làm RoleC. Kết quả cần trả về RoleB)

10. Xóa dup dữ liệu bảng Employee\_Department

11. Giả sử doanh nghiệp trên sử dụng cơ sở dữ liệu MySQL. Viết câu lệnh để tạo mới một user và cấp quyền read-all cho user đó, không cấp quyền chỉnh sửa dữ liệu. Với mỗi câu SQL, chụp ảnh kết quả trả về

## Phân tích yêu cầu

|  |  |  |
| --- | --- | --- |
| ID | Requirement | Describe |
| 1 | Tạo bảng, khởi tạo dữ liệu | Tạo sample database với tên hrm, các bảng tương ứng , và khởi tạo sample data |
| 2 | Lập danh sách nhân viên, phòng ban, vị trí công việc của từng nhân viên ở thời điểm hiện tại | SQL Query |
| 3 | Tìm phòng ban có nhiều nhân viên nhất | SQL Query |
| 4 | Tìm nhân viên làm việc cho nhiều phòng ban nhất. | SQL Query |
| 5 | Tìm tất cả những nhân viên mà làm chung ít nhất 1 phòng ban với nhân viên tên John Doe | SQL Query |
| 6 | Tính mức lương trung bình theo từng phòng ban. | SQL Query |
| 7 | Tìm tất cả những nhân viên có làm việc chung với nhân viên John Doe tại tất cả phòng ban. | SQL Query |
| 8 | Tìm nhân viên có nhiều cấp dưới nhất | SQL Query |
| 9 | Lập danh sách nhân viên và vị trí công việc mà nhân viên đã làm ngay trước khi làm ở vị trí hiện tại (Ví dụ: nhân viên John Doe có quá trình làm việc qua các vị trí RoleA → RoleB → RoleC. Hiện tại đang làm RoleC. Kết quả cần trả về RoleB) | SQL Query |
| 10 | Xóa dup dữ liệu bảng Employee\_Department | Tạo ngữ cảnh data bị duplicate trong bảng Employee\_Department |
| 11 | Giả sử doanh nghiệp trên sử dụng cơ sở dữ liệu MySQL. Viết câu lệnh để tạo mới một user và cấp quyền read-all cho user đó, không cấp quyền chỉnh sửa dữ liệu. | Database Administration Statements |

## Lược đồ quan hệ và mô tả dữ liệu đích

Employee (**ID**, Name, Nationality, DOB, Phone Number, Employee ID of Manager, Salary, Onboard Date)

Role (**ID**, Role Name)

Employee\_Role (*Employee ID, Role ID*, Start Date)

Department (**ID**, Name)

Employee\_Department (*Employee ID, Department ID,* Start date, End Date)

### **6.1 Employee**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Column\_name | Data\_type | Description |
| 1 | ID | INT | Mã nhân viên (Primary Key) |
| 2 | Name | VARCHAR(100) | Tên nhân viên |
| 3 | Nationality | VARCHAR (100) | Quốc tịch |
| 4 | DOB | DATE | Ngày sinh |
| 5 | PhoneNumber | VARCHAR(15) | Số điện thoại |
| 6 | ManagerID | INT | Mã nhân viên Quản lý |
| 7 | Salary | INT | Lương |
| 8 | OnboardDate | DATE | Ngày vào làm |

### **6.2 Role**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Column\_name | Data\_type | Description |
| 1 | ID | INT | Mã vai trò (Primary Key) |
| 2 | RoleName | VARCHAR (100) | Tên vai trò |

### **6.3 Department**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Column\_name | Data\_type | Description |
| 1 | ID | INT | Mã phòng ban (Primary Key) |
| 2 | Name | VARCHAR (100) | Tên phòng ban |

### **6.4 Employee\_Role**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Column\_name | Data\_type | Description |
| 1 | EmployeeID | INT | Mã nhân viên (Foreign key) |
| 2 | RoleID | INT | Mã vai trò (Foreign key) |
| 3 | StartDate | DATE | Ngày bắt đầu |

### **6.5 Employee\_Department**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Column\_name | Data\_type | Description |
| 1 | EmployeeID | INT | Mã nhân viên (Foreign Key) |
| 2 | DepartmentID | INT | Mã phòng ban (Foreign Key) |
| 3 | StartDate | DATE | Ngày bắt đầu |
| 4 | EndDate | DATE | Ngày kết thúc |

## Thực hiện yêu cầu (SQL)

1. Tạo bảng, khởi tạo dữ liệu



Github: [snap\_tech\_son/script\_sql/structure\_data\_hrm.sql at main · CarsonIT/snap\_tech\_son (github.com)](https://github.com/CarsonIT/snap_tech_son/blob/main/script_sql/structure_data_hrm.sql)

2. Lập danh sách nhân viên, phòng ban, vị trí công việc của từng nhân viên ở thời điểm hiện tại

**SELECT**

e.Name **AS** EmployeeName,

d.Name **AS** Department,

r.RoleName **AS** JobPosition

**FROM**

Employee e

**JOIN**

Employee\_Department ed **ON** e.ID = ed.EmployeeID

**JOIN**

Department d **ON** ed.DepartmentID = d.ID

**JOIN**

Employee\_Role er **ON** e.ID = er.EmployeeID

**JOIN**

**Role** r **ON** er.RoleID = r.ID

**WHERE**

ed.EndDate **IS** **NULL**

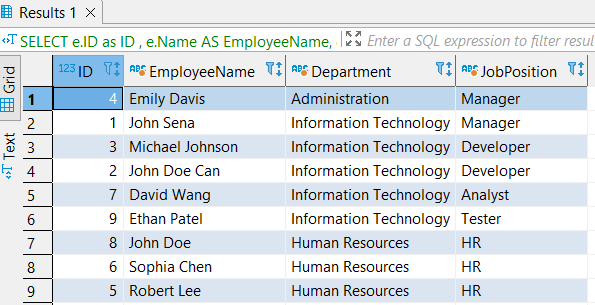
**AND** er.StartDate = (

**SELECT** **MAX**(StartDate)

**FROM** Employee\_Role

**WHERE** EmployeeID = e.ID

);



3. Tìm phòng ban có nhiều nhân viên nhất.

**SELECT**

**Top** 1

d.Name **AS** DepartmentName,

**COUNT**(e.ID) **AS** EmployeeCount

**FROM**

Department d

**JOIN**

Employee\_Department ed **ON** d.ID = ed.DepartmentID

**JOIN**

Employee e **ON** ed.EmployeeID = e.ID

**WHERE**

ed.EndDate **IS** **NULL**

**GROUP** **BY**

d.Name

**ORDER** **BY**

EmployeeCount **DESC**

A screenshot of a computer

Description automatically generated

4. Tìm nhân viên làm việc cho nhiều phòng ban nhất.

**SELECT**

e.ID **AS** EmployeeID,

e.Name **AS** EmployeeName,

**COUNT**(**DISTINCT** ed.DepartmentID) **AS** DepartmentCount

**FROM**

Employee e

**JOIN**

Employee\_Department ed **ON** e.ID = ed.EmployeeID

**GROUP** **BY**

e.ID, e.Name

**HAVING**

**COUNT**(**DISTINCT** ed.DepartmentID) = (

**SELECT**

**MAX**(DepartmentCount)

**FROM** (

**SELECT**

**COUNT**(**DISTINCT** ed.DepartmentID) **AS** DepartmentCount

**FROM**

Employee e

**JOIN**

Employee\_Department ed **ON** e.ID = ed.EmployeeID

**GROUP** **BY**

e.ID

) **AS** DepartmentCounts

);

A screenshot of a computer

Description automatically generated

5. Tìm tất cả những nhân viên mà làm chung ít nhất 1 phòng ban với nhân viên tên John Doe

**SELECT** **DISTINCT**

e.ID **AS** EmployeeID,

e.Name **AS** EmployeeName

**FROM**

Employee e

**JOIN**

Employee\_Department ed **ON** e.ID = ed.EmployeeID

**JOIN**

Employee johndoe **ON** johndoe.Name = 'John Doe'

**JOIN**

Employee\_Department johnDept **ON** johndoe.ID = johnDept.EmployeeID

**WHERE**

ed.DepartmentID = johnDept.DepartmentID

**AND** e.ID <> johndoe.ID;

A screenshot of a computer

Description automatically generated

6. Tính mức lương trung bình theo từng phòng ban.

**SELECT**

d.Name **AS** DepartmentName,

**AVG**(e.Salary) **AS** AverageSalary

**FROM**

Department d

**JOIN**

Employee\_Department ed **ON** d.ID = ed.DepartmentID

**JOIN**

Employee e **ON** ed.EmployeeID = e.ID

**GROUP** **BY**

d.Name;

A screenshot of a computer

Description automatically generated

7. Tìm tất cả những nhân viên có làm việc chung với nhân viên John Doe tại tất cả phòng ban.

**WITH** JohnDoeDepartments **AS** (

**SELECT** **DISTINCT** ED.DepartmentID

**FROM** Employee E

**JOIN** Employee\_Department ED **ON** E.ID = ED.EmployeeID

**WHERE** E.Name = 'John Doe'

)

**SELECT** **DISTINCT** E2.\*, ED1.DepartmentID

**FROM** Employee E1

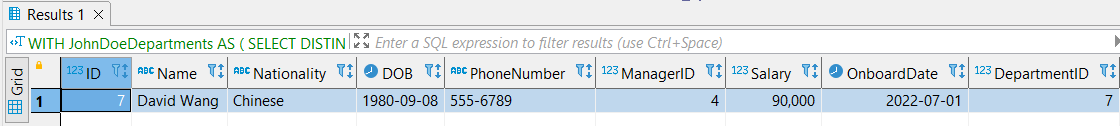
**JOIN** Employee\_Department ED1 **ON** E1.ID = ED1.EmployeeID

**JOIN** Employee E2 **ON** ED1.DepartmentID = E2.ID

**JOIN** Employee\_Department ED2 **ON** E2.ID = ED2.EmployeeID

**WHERE** E1.Name = 'John Doe'

**AND** ED2.DepartmentID **IN** (**SELECT** DepartmentID **FROM** JohnDoeDepartments);



8. Tìm nhân viên có nhiều cấp dưới nhất

**WITH** SubordinateCount **AS** (

**SELECT**

ManagerID,

**COUNT**(\*) **AS** SubordinateCount

**FROM**

Employee

**WHERE**

ManagerID **IS** **NOT** **NULL**

**GROUP** **BY**

ManagerID

)

**SELECT**

E.ID **AS** ManagerID,

E.Name **AS** ManagerName,

SC.SubordinateCount

**FROM**

SubordinateCount SC

**JOIN**

Employee E **ON** SC.ManagerID = E.ID

**WHERE**

SC.SubordinateCount = (

**SELECT**

**MAX**(SubordinateCount)

**FROM**

SubordinateCount

);

A screenshot of a computer

Description automatically generated

9. Lập danh sách nhân viên và vị trí công việc mà nhân viên đã làm ngay trước khi làm ở vị trí hiện tại (Ví dụ: nhân viên John Doe có quá trình làm việc qua các vị trí RoleA → RoleB → RoleC. Hiện tại đang làm RoleC. Kết quả cần trả về RoleB)

**WITH** EmployeeRolesRanked **AS** (

**SELECT**

ER.EmployeeID,

R.RoleName,

ER.StartDate,

**ROW\_NUMBER**() **OVER** (**PARTITION** **BY** ER.EmployeeID **ORDER** **BY** ER.StartDate **DESC**) **AS** RoleRank

**FROM**

Employee\_Role ER

**JOIN**

**Role** R **ON** ER.RoleID = R.ID

)

**SELECT**

E.ID **AS** EmployeeID,

E.Name **AS** EmployeeName,

PrevRole.RoleName **AS** PreviousRole,

CurrRole.RoleName **AS** CurrentRole

**FROM**

Employee E

**JOIN**

EmployeeRolesRanked CurrRole **ON** E.ID = CurrRole.EmployeeID **AND** CurrRole.RoleRank = 1

**LEFT** **JOIN**

EmployeeRolesRanked PrevRole **ON** E.ID = PrevRole.EmployeeID **AND** PrevRole.RoleRank = 2

**WHERE**

CurrRole.StartDate = (**SELECT** **MAX**(StartDate) **FROM** Employee\_Role **WHERE** EmployeeID = E.ID);

A screenshot of a computer

Description automatically generated

10. Xóa dup dữ liệu bảng Employee\_Department

**WITH** DuplicateCTE **AS** (

**SELECT**

EmployeeID,

DepartmentID,

StartDate,

EndDate,

**ROW\_NUMBER**() **OVER** (**PARTITION** **BY** EmployeeID, DepartmentID **ORDER** **BY** StartDate) **AS** RowNum

**FROM**

Employee\_Department

)

**DELETE** **FROM** DuplicateCTE

**WHERE** RowNum > 1;

11. Giả sử doanh nghiệp trên sử dụng cơ sở dữ liệu MySQL. Viết câu lệnh để tạo mới một user và cấp quyền read-all cho user đó, không cấp quyền chỉnh sửa dữ liệu.

- Tạo user hongson

**CREATE** **USER** 'hongson'@'localhost' IDENTIFIED **BY** 'Password@123';

- Gán quyền cho user hongson read-all, không cấp quyền chỉnh sửa

**GRANT** **SELECT** **ON** \*.\* **TO** 'hongson'@'localhost';

* Kiểm tra quyền

**SELECT** **user**, host, Select\_priv, Insert\_priv, Update\_priv, Delete\_priv, Create\_priv, Drop\_priv **FROM** mysql.**user** **where** **user** = 'hongson';

A computer screen with white text and purple text

Description automatically generated

A black screen with white text

Description automatically generated

## Link github dự án

Github: [CarsonIT/Youtube\_video (github.com)](https://github.com/CarsonIT/Youtube_video)

# Tài liệu tham khảo

[1]: [Trending YouTube Video Statistics | Kaggle](https://www.kaggle.com/datasets/datasnaek/youtube-new?select=CAvideos.csv)

[2]: [Power BI: Create a YouTube Channel Statistics Dashboard Using YouTube Feeds (No API Key Required) - YouTube](https://www.youtube.com/watch?v=DwzF-3M3tMs)