GEN-AI ASSIGNMENT-3

MySQL Tables Creation

Create the `students` table with relationships to both `department` and `year`:

```
CREATE TABLE department (
    dept_id INT AUTO_INCREMENT PRIMARY KEY,
    dept_name VARCHAR(50) NOT NULL
);

CREATE TABLE year (
    year_id INT AUTO_INCREMENT PRIMARY KEY,
    year_name VARCHAR(10) NOT NULL
);

CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    dept_id INT,
    year_id INT,
    FOREIGN KEY (dept_id) REFERENCES department(dept_id),
    FOREIGN KEY (year_id) REFERENCES year(year_id)
);
```

Inserting sample data:

```
INSERT INTO department (dept_name) VALUES ('CSE'), ('ECE'), ('ME'), ('CE'), ('EE');
```

```
INSERT INTO year (year_name) VALUES ('First'), ('Second'), ('Third'), ('Fourth');


INSERT INTO students (first_name, last_name, dept_id, year_id) VALUES
('Srikanth', 'Thirumani', 1, 1),
('Rohith', 'Thumma', 1, 2),
('Vishnu', 'Andhe', 1, 3),
('Sammad', 'Mohammed', 1, 4),
('Nikhil', 'Dadige', 1, 1),
('Praveen', 'Chigurla', 2, 2),
('Shiva teja', 'Kandlapally', 2, 3),
('Uday kiran’, 'Uppu', 2, 4),
('Saivarun', 'Somishetty', 2, 1),
('Koushik reddy', 'Jai', 2, 2),
('Geethsai', 'Taylor', 3, 3),
('Yashwanth', 'Goti', 3, 4),
('Keerthi', 'Kotha', 3, 1),
('Sreshta', 'Soma', 3, 2),
('Rakshitha', 'Sanda', 3, 3),
('Harsha vardhini', 'Pendyala', 4, 4),
('Anna', 'Harris', 4, 1),
('Jake', 'Nelson', 4, 2),
('Lucy', 'Carter', 4, 3),
('Ryan', 'Mitchell', 4, 4),
('Sam', 'Perez', 5, 1),
('Nina', 'Roberts', 5, 2),
('Matt', 'Evans', 5, 3),
('Sara', 'Edwards', 5, 4),
('Paul', 'Collins', 5, 1);
```

Queries:

.Display students from the CSE department:

SELECT * FROM students WHERE dept_id = (SELECT dept_id FROM department WHERE dept_name = 'CSE');

Display only `dept_name` using the `students` table:

SELECT DISTINCT d.dept_name

FROM students s

JOIN department d ON s.dept_id = d.dept_id;

Display students sorted by department and first name:

SELECT s.first_name, s.last_name, d.dept_name

FROM students s

JOIN department d ON s.dept_id = d.dept_id

ORDER BY d.dept_name, s.first_name;

Translate MySQL to MongoDB

```
CREATE TABLE department (
    dept_id INT AUTO_INCREMENT PRIMARY KEY,
    dept_name VARCHAR(50) NOT NULL
);

CREATE TABLE year (
    year_id INT AUTO_INCREMENT PRIMARY KEY,
    year_name VARCHAR(10) NOT NULL
);

CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    dept_id INT,
```

```
    year_id INT,
    FOREIGN KEY (dept_id) REFERENCES department(dept_id),
    FOREIGN KEY (year_id) REFERENCES year(year_id)
);
```

To create a similar structure in MongoDB, you can embed the related documents or use references.

1.Using Embedding (not the best for normalized data but can be simpler):

```
{
    "_id": ObjectId(),
    "first_name": "Srikanth",
    "last_name": "Thirumani",
    "department": {
        "dept_id": 1,
        "dept_name": "CSE"
    },
    "year": {
        "year_id": 1,
        "year_name": "First"
    }
}
```

2.Using References (more similar to normalized SQL structure):

Department Collection

```
{
    "_id": ObjectId(),
    "dept_id": 1,
    "dept_name": "CSE"
```

```
}
```

Year Collection

```
{
    "_id": ObjectId(),
    "year_id": 1,
    "year_name": "First"
}
```

Students Collection

```
{
    "_id": ObjectId(),
    "first_name": "Srikanth",
    "last_name": "Thirumani",
    "dept_id": 1,
    "year_id": 1
}
```

Insert 5 Students for Each Department

This can be done similarly by inserting documents into the `students` collection with references to `dept_id` and `year_id`.

MongoDB Queries

1.Display students from the CSE department:

```
db.students.find({ dept_id: db.department.findOne({ dept_name: "CSE" }).dept_id });
```

2.Display only `dept_name` using `students` table

```
db.students.aggregate([
    {
        $lookup: {
            from: "department",
            localField: "dept_id",
            foreignField: "dept_id",
            as: "department"
        }
    },
    {
        $unwind: "$department"
    },
    {
        $group: {
            _id: "$department.dept_name"
        }
    },
    {
        $project: {
            _id: 0,
            dept_name: "$_id"
        }
    }
```

```
]);
```

3.Display students sorted by department and first name:

```
db.students.aggregate([
    {
        $lookup: {
            from: "department",
            localField: "dept_id",
            foreignField: "dept_id",
            as: "department"
        }
    },
    {
        $unwind: "$department"
    },
    {
        $sort: {
            "department.dept_name": 1,
            "first_name": 1
        }
    },
    {
        $project: {
            _id: 0,
```

```
            first_name: 1,
            last_name: 1,
            dept_name: "$department.dept_name"
        }
    }
]);
```

This completes the process of translating the MySQL schema and queries to MongoDB equivalents.