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),

('Ram', 'Abhi', 4, 1),

('Reddy', 'Deekshith', 4, 2),

('', '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": "Jai Koushik",

"last\_name": "Reddy",

"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.