## *Project Description:*

*Our project centers on the development of a robust database system tailored for the Noon Academy application, an emerging educational platform designed for student engagement. With a commitment to providing interactive learning experiences, Noon Academy seeks to revolutionize education delivery. The proposed database system aims to effectively manage user data and content, ensuring optimal functionality and support for the platform. Our endeavor is to construct a dependable infrastructure that resonates with Noon Academy's dedication to fostering dynamic educational environments.*

## *View Description:*

*The student is an individual seeking educational support and benefits from the database to access interactive learning sessions and participate in educational activities, along with tracking their progress. Their responsibilities include actively engaging in learning sessions, completing assigned tasks, and participating actively in discussions. The important data for students includes personal information, learning materials utilized, updates to personal data, and the option to delete and update data as needed.*

## *Data Requirements:*

***Student:***
*The student entity represents individuals enrolled in courses within the platform. It is identified by a unique StudentID and includes attributes such as Name, Email, Phone Number, and Enrollment Date. Each student is enrolled in one or more courses.*

***Instructor:***
*The instructor entity represents educators responsible for teaching courses within the platform. It is identified by a unique InstructorID and includes attributes such as Name, Email, and TeachingField. Each instructor teaches one or more courses.*

***Section:***
*The Section entity represents distinct organizational units, each containing multiple courses. It is identified by a unique SectionCode and includes attributes such as SectionName, Description, StartDate, and EndDate. Each section contains multiple courses, providing students with a cohesive learning experience.*

***Assignment:***
*The Assignment entity represents tasks or projects assigned to students as part of their coursework. It is identified by a unique AssignmentKey and includes attributes such as Title, Description, start date, end date Due Date. Each assignment is associated with one course and is completed by one or more students.*

***Course:***
*The Course entity represents an educational program offered by the Noon Academy platform. It is identified by a unique CourseID and includes attributes such as CourseName, Description, TeacherName. Each course is taken by zero or multiball students and each course is organized into one section.*

## *Transaction Requirements:*

***Data Entry:***

*a) Register as a student.*
*b) Enroll in course.*

***Data update/deletion:***

*a) Update section.*
*b) Update the language.*
*c) Unregister course.*

***Data Queries:***

*a) List of available courses for registration.*

*b) List assignments along with their courses.*

*c) List all courses with their enrollment count*

*d) List all courses without any assignments*

*e) List of courses completed in the year 2020.*

*f) List all assignments due before a specific date ('2023-10-01')*

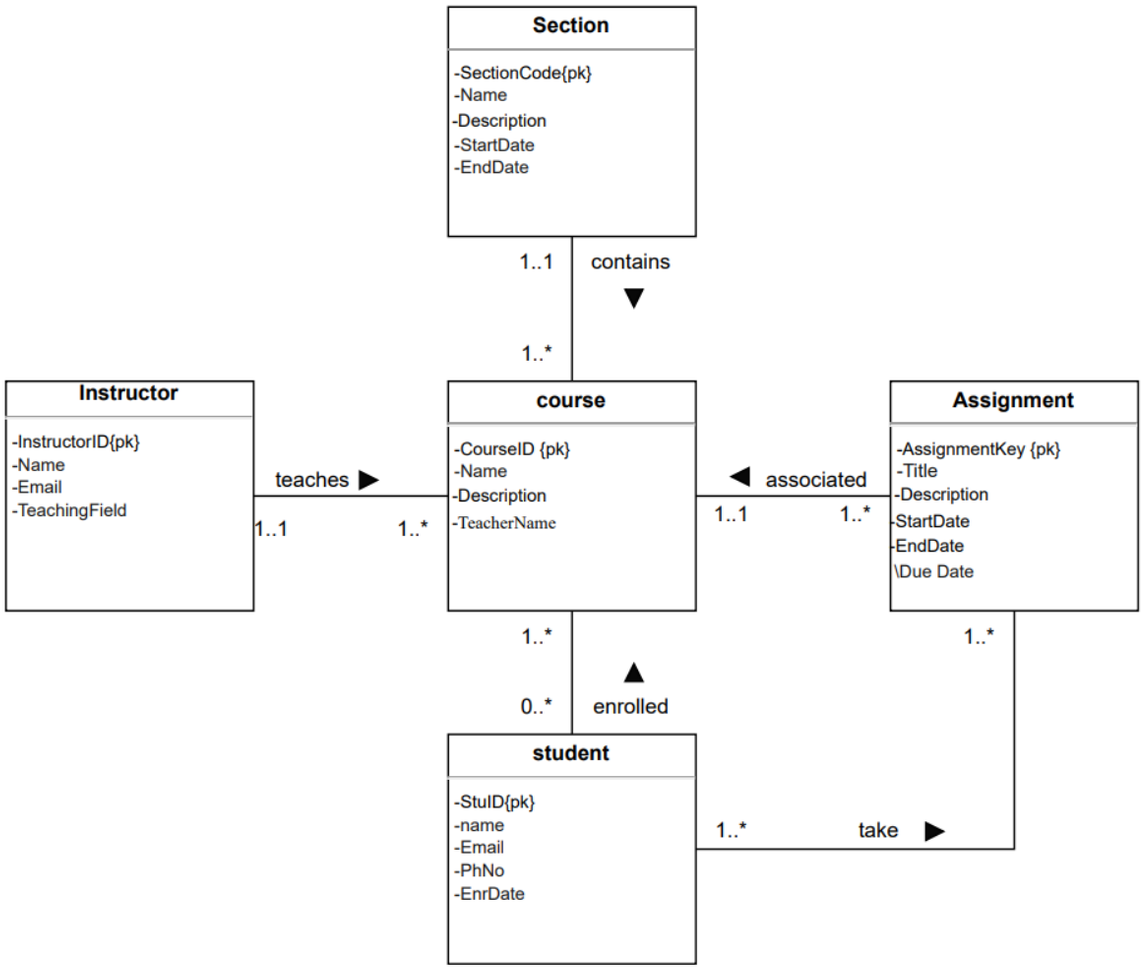
*g) List of courses completed in the Physics department.*

*h) List all courses taught by instructors in a specific teaching field ('Computer Science').*

*i) List of assignments worked on in October 2023.*

*j) List all sections with their corresponding courses*

## *Global enhanced entity relationship diagram (EER):*



***Relational Schema:***

Section (SectionCode, Name, Description, StartDate, EndDate)
**Primary key:** SectionCode.

Course (CourseID, Name, Description, TeacherName, SectionCode, InstructorID)
**Primary key:** CourseID.
**Foreign key:** SectionCode references Section.
**Foreign key:** InstructorID references Instructor.

Instructor (InstructorID, Name, Email, TeachingField)
**Primary key:** InstructorID.

Student (StuID, Name, Email, PhNo, EnrDate)
**Primary key:** StuID.

StudentEnrolledCourse (CourseID, StuID)
**Foreign key:** CourseID references Course.
**Foreign key:** StuID references Student.

Assignment (AssignmentKey, Title, Description, StartDate, EndDate, CourseID)
**Primary key:** AssignmentKey.
**Foreign key:** CourseID references Course.

StudentTakeAssignment (AssignmentKey, StuID)
**Foreign key:** AssignmentKey references Assignment.
**Foreign key:** StuID references Student.

## *Data Dictionary showing description of all entities:*

| Entity Name | Description | Occurrence |
|---|---|---|
| ***Student*** | *The student entity represents individuals enrolled in courses within the platform.* | *Each student is enrolled in one or more courses.*<br>*Each student takes one or more assignments.* |
| ***Instructor*** | *The instructor entity represents educators responsible for teaching courses within the platform.* | *Each instructor teaches one or more courses.* |
| ***Section*** | *The Section entity represents distinct organizational units, each containing multiple courses.* | *Each section contains multiple courses.* |
| ***Assignment*** | *The Assignment entity represents tasks or projects assigned to students as part of their coursework.* | *Each assignment is associated with one course and is completed by one or more students.* |
| ***Course*** | *The Course entity represents an educational program offered by the Noon Academy platform.* | *Each course is taken by one or more students and organized into one section and associated with one or more assignments.*<br>*One or more courses are taught by one instructor.* |

## *Data Dictionary showing description of all relationships:*

| Entity Name | Multiplicity | Relationship | Entity Name | Multiplicity |
|---|---|---|---|---|
| ***Student*** | **0..***<br>**1..*** | ***Enrolled***<br>***Take*** | **Course**<br>**Assignment** | **1..***<br>**1..*** |
| ***Instructor*** | **1..1** | ***Teaches*** | **Course** | **1..*** |
| ***Section*** | **1..1** | ***Contains*** | **Course** | **1..*** |
| ***Assignment*** | **1.. *** | ***Associated*** | **Course** | **1..1** |

## *Data Dictionary showing description of all attributes:*

| Entity Name | Attribute | Description | Data Type | Length | Nulls | Multi-Valued | Default Value | Range | PK |
|---|---|---|---|---|---|---|---|---|---|
| ***Course*** | ***CourseID*** | *Uniquely identifies the course.* | *Char* | *4* | *No* | *No* | | *0000-9999* | *Y* |
| | ***Name*** | *Name of the Course.* | *Varchar* | *30* | *No* | *No* | | | |
| | ***Description*** | *Brief Description of the Course.* | *Varchar* | *120* | *Yes* | *No* | | | |
| | ***Teacher Name*** | *Name of the instructor that teaches the course.* | *Varchar* | *30* | *No* | *No* | | | |
| ***Section*** | ***SectionCode*** | *Uniquely identifies the section.* | *Char* | *7* | *No* | *No* | | *0000000-9999999* | *Y* |
| | ***Name*** | *Name of the Section.* | *Varchar* | *30* | *No* | *No* | | | |
| | ***Description*** | *Description of the Section.* | *Varchar* | *120* | *Yes* | *No* | | | |
| | ***StartDate*** | *Start Date of the Section.* | *Date* | | *No* | *No* | | | |
| | ***EndDate*** | *End Date of the Section.* | *Date* | | *No* | *No* | | | |
| ***Instructor*** | ***InstructorID*** | *Uniquely identifies the instructor.* | *Char* | *10* | *No* | *No* | | *0000000000-9999999999* | *Y* |
| | ***Name*** | *Name of the instructor.* | *Varchar* | *30* | *No* | *No* | | | |
| | ***Email*** | *Email to the instructor.* | *Varchar* | *30* | *Yes* | *No* | | | |
| | ***TeachingField*** | *Field of sections the instructor teaches.* | *Varchar* | *20* | *No* | *No* | | | |
| ***Student*** | ***StuID*** | *Uniquely identifies the student.* | *Char* | *10* | *No* | *No* | | *0000000000-9999999999* | *Y* |
| | ***Name*** | *Name of the student.* | *Varchar* | *30* | *No* | *No* | | | |
| | ***Email*** | *Email from the student.* | *Varchar* | *30* | *Yes* | *No* | | | |
| | ***PhNo*** | *Phone number of the student.* | *Char* | *10* | *Yes* | *No* | | | |
| | ***EnrDate*** | *Date in which student was enrolled.* | *Date* | | *No* | *No* | | | |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| ***Assignment*** | ***AssignmentKey*** | *Uniquely identifies the Assignment.* | *Char* | *7* | *No* | *No* | | *0000000-9999999* | *Y* |
| | ***Title*** | *Title of the assignment.* | *Varchar* | *30* | *No* | *No* | | | |
| | ***Description*** | *Brief Description of the assignment.* | *Varchar* | *120* | *No* | *No* | | | |
| | ***StartDate*** | *Start Date of the Assignment.* | *Date* | | *No* | *No* | | | |
| | ***EndDate*** | *End Date of the Assignment.* | *Date* | | *No* | *No* | | | |

### *DB tables creation commands:*

```
CREATE TABLE Student (
   StuID CHAR(10) NOT NULL,
   Name VARCHAR(30) NOT NULL,
   Email VARCHAR(30),
   PhNo VARCHAR (10),
   EnrDate DATE NOT NULL,
   PRIMARY KEY (StuID)
);

CREATE TABLE Instructor (
   InstructorID CHAR(10) NOT NULL,
   Name VARCHAR(30) NOT NULL,
   Email VARCHAR(30),
   TeachingField VARCHAR(20) NOT NULL,
   PRIMARY KEY (InstructorID)
);

CREATE TABLE Section (
   SectionCode CHAR(7) NOT NULL,
   Name VARCHAR(30) NOT NULL,
   Description VARCHAR(120),
   StartDate DATE NOT NULL,
   EndDate DATE NOT NULL,
   PRIMARY KEY (SectionCode)
);

CREATE TABLE Course (
   CourseID CHAR(4) NOT NULL,
   Name VARCHAR(30) NOT NULL,
   Description VARCHAR(120),
   TeacherName VARCHAR(30) NOT NULL,
   SectionCode CHAR(7) NOT NULL,
   InstructorID CHAR(10) NOT NULL,
   PRIMARY KEY (CourseID),
   FOREIGN KEY (SectionCode) REFERENCES Section(SectionCode)
    ON UPDATE CASCADE,
   FOREIGN KEY (InstructorID) REFERENCES Instructor(InstructorID)
    ON UPDATE SET NULL
);


CREATE TABLE Assignment (
   AssignmentKey CHAR(7) NOT NULL,
   Title VARCHAR(30) NOT NULL,
   Description VARCHAR(120) NOT NULL,
   StartDate DATE NOT NULL,
   EndDate DATE NOT NULL,
   CourseID CHAR(4) NOT NULL,
   PRIMARY KEY (AssignmentKey),
   FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);

CREATE TABLE StudentEnrolledCourse (
   CourseID CHAR(4),
   StuID CHAR(10),
   FOREIGN KEY (CourseID) REFERENCES Course(CourseID),
   FOREIGN KEY (StuID) REFERENCES Student(StuID)
);

CREATE TABLE StudentTakeAssignment (
   AssignmentKey CHAR(7),
   StuID CHAR(10),
   FOREIGN KEY (AssignmentKey) REFERENCES Assignment(AssignmentKey),
   FOREIGN KEY (StuID) REFERENCES Student(StuID)
);
```

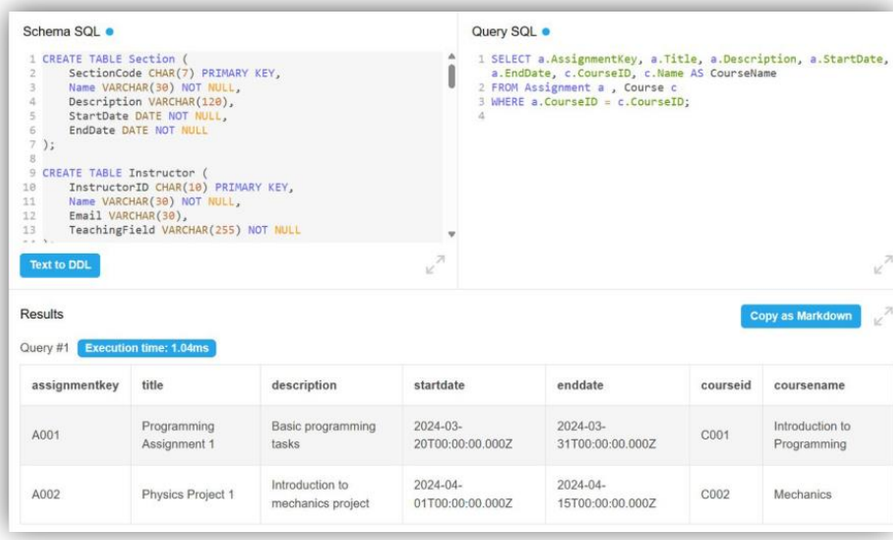
***Data insertion commands:***

```
INSERT INTO Student VALUES
('S000000001', 'Sara Ibrahim', 'sara89@gmail.com', '0501234567', '2023-09-06');

INSERT INTO Student VALUES
('S000000002', 'Faisal Ahmed', 'faisallahmed@outlook.com', '0559876543', '2023-10-19');

INSERT INTO Student VALUES
('S000000003', 'Maha Muhammed', 'mmuhammed@gmail.com', '0532468109', '2023-09-01');

INSERT INTO Student VALUES
('S000000004', 'Sultan Ali', 'sulali@outlook.com', '0541357924', '2023-12-09');

INSERT INTO Instructor VALUES
('I000000001', 'Dr. Michael Lee', 'michael@gmail.com', 'Computer Science');

INSERT INTO Instructor VALUES
('I000000002', 'Prof. Alia Suleiman', 'aliasul@gmail.com', 'Mathematics');

INSERT INTO Instructor VALUES
('I000000003', 'Dr. Jennifer Wang', 'jennifer@gmail.com', 'Physics');

INSERT INTO Section VALUES
('SEC0001', 'Programming', 'A section on programming concepts and instructions', '2023-09-01', '2024-01-15');

INSERT INTO Section VALUES
('SEC0002', 'Calculus', 'A section on quantities and their differentiation', '2023-09-01', '2024-01-15');

INSERT INTO Course VALUES
('C001', 'Introduction to Python', 'Introductory course on Python concepts and programming techniques', 'Dr. Michael Lee', 'SEC0001',
'I000000001');

INSERT INTO Course VALUES
('C002', 'Swift', 'Course on programming with Swift language for intermediate learners.', 'Dr. Michael Lee', 'SEC0001', 'I000000001');

INSERT INTO Course VALUES
('C003', 'Discrete Mathematics', 'Study of mathematical distinct structures for computing.', 'Prof. Alia Suleiman', 'SEC0002',
'I000000002');

INSERT INTO Assignment VALUES
('A000001', 'Python Assignment 1', 'Write a program to calculate factorial', '2023-9-10', '2023-09-20', 'C001');

INSERT INTO Assignment VALUES
('A000002', 'Math Homework 1', 'Solve exercises 1-10 from chapter 1', '2023-10-10', '2023-10-20', 'C002');

INSERT INTO StudentEnrolledCourse (CourseID, StuID) VALUES
('C001', 'S000000001'), ('C002', 'S000000002');

INSERT INTO StudentTakeAssignment (AssignmentKey, StuID) VALUES
('A000001', 'S000000001'), ('A000002', 'S000000002');
```

## *Data Queries commands and outputs:*

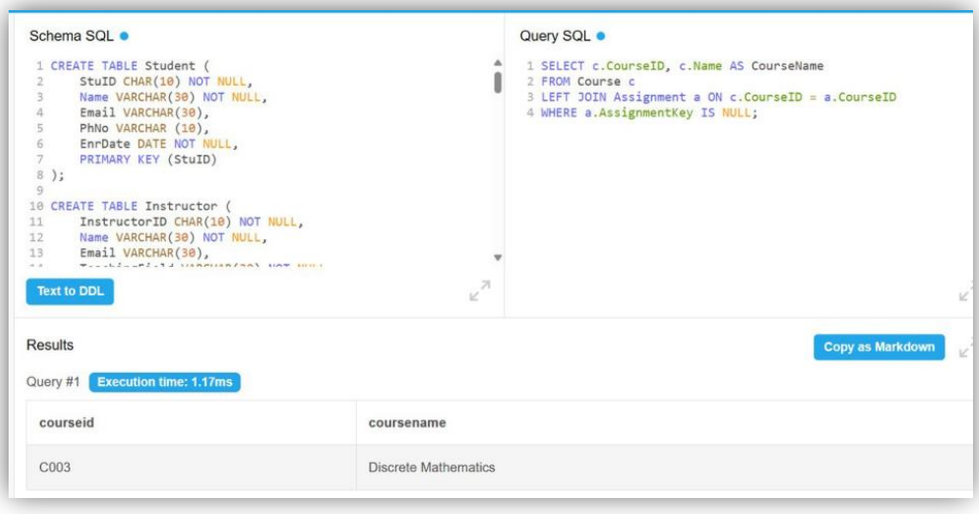
- ***List assignments along with their courses:***

```
SELECT a.AssignmentKey, a.Title, a.Description, a.StartDate , a.EndDate ,c.courseID,
c.Name AS CourseName
FROM Assignment a ,Course c
WHERE A.CourseID = C.CourseID;
```



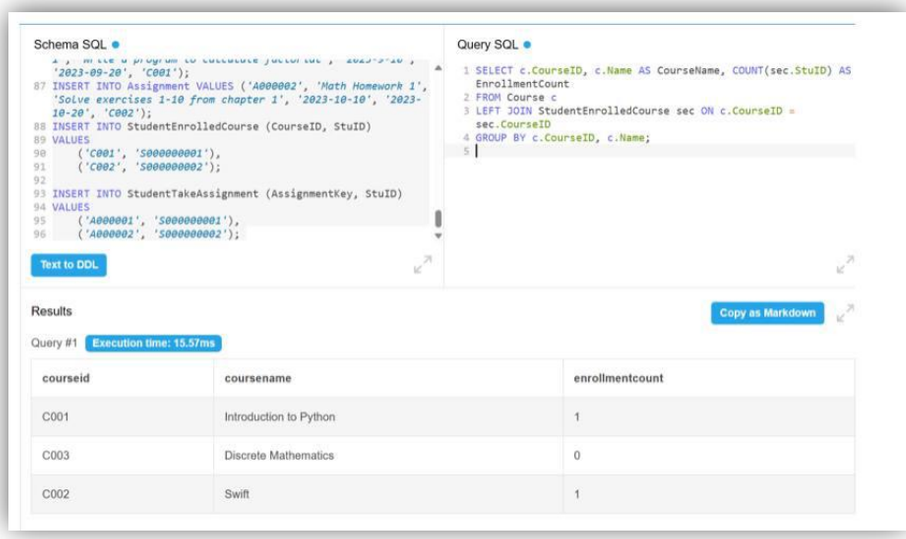

- ***List all courses without any assignments:***

```
SELECT c.CourseID, c.Name AS CourseName
FROM Course c
LEFT JOIN Assignment a ON c.CourseID = a.CourseID
WHERE a.AssignmentKey IS NULL;
```



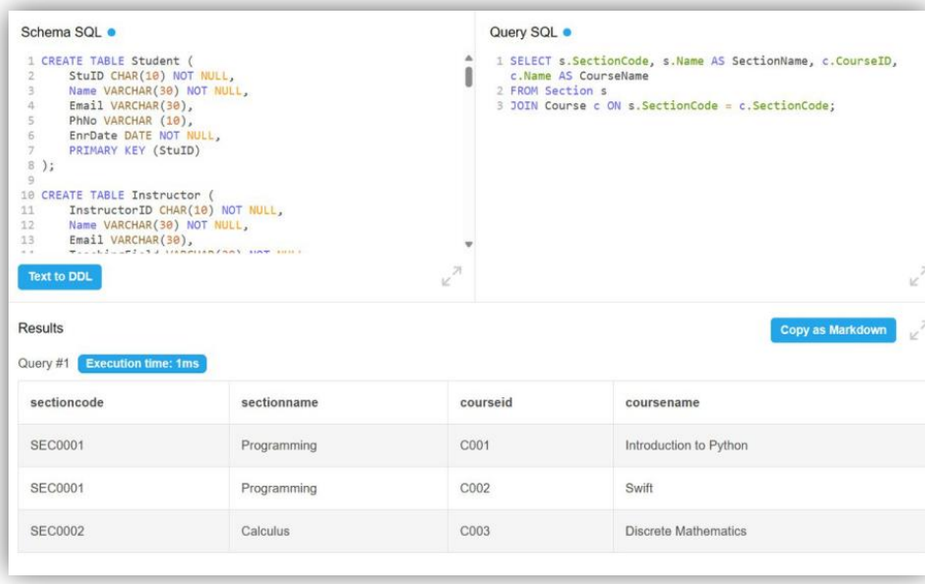
- ***List all courses with their enrollment count:***

```
SELECT c.CourseID, c.Name AS CourseName, COUNT(sec.StuID) AS EnrollmentCount
FROM Course c
LEFT JOIN StudentEnrolledCourse sec ON c.CourseID = sec.CourseID
GROUP BY c.CourseID, c.Name;
```




- ***List all sections with their corresponding courses:***

```
SELECT s.SectionCode, s.Name AS SectionName, c.CourseID, c.Name AS CourseName
FROM Section s
JOIN Course c ON s.SectionCode = c.SectionCode;
```

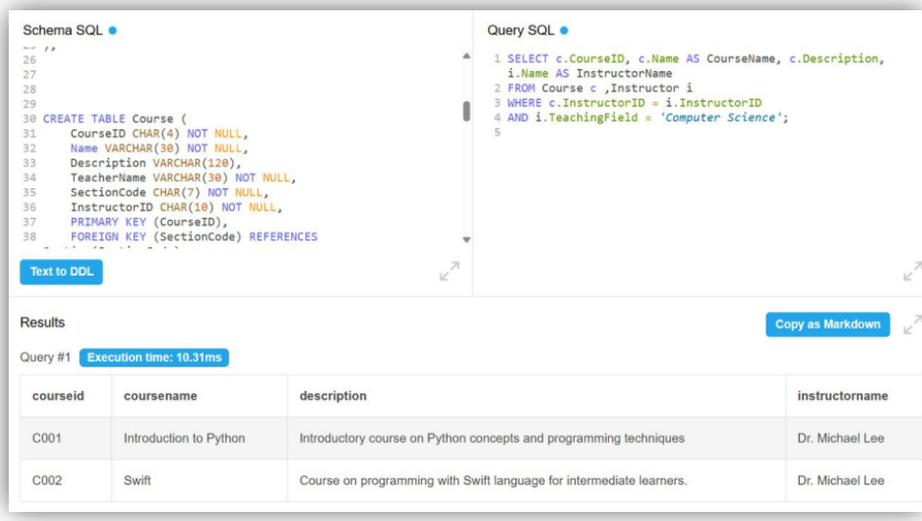


- ***List all assignments due before a specific date ('2023-10-01'):***

```
SELECT *
FROM Assignment
WHERE EndDate < '2023-10-01';
```


- ***List all courses taught by instructors in a specific teaching field ('Computer Science'):***

```
SELECT c.CourseID, c.Name AS CourseName, c.Description,  i.Name AS InstructorName
FROM Course c ,Instructor i
WHERE c.InstructorID = i.InstructorID
AND i.TeachingField = 'Computer Science';
```



- ***List of assignments worked on in October 2023:***

```
SELECT *
FROM Assignment
WHERE StartDate >= '2023-10-01' AND EndDate <= '2023-10-31';
```