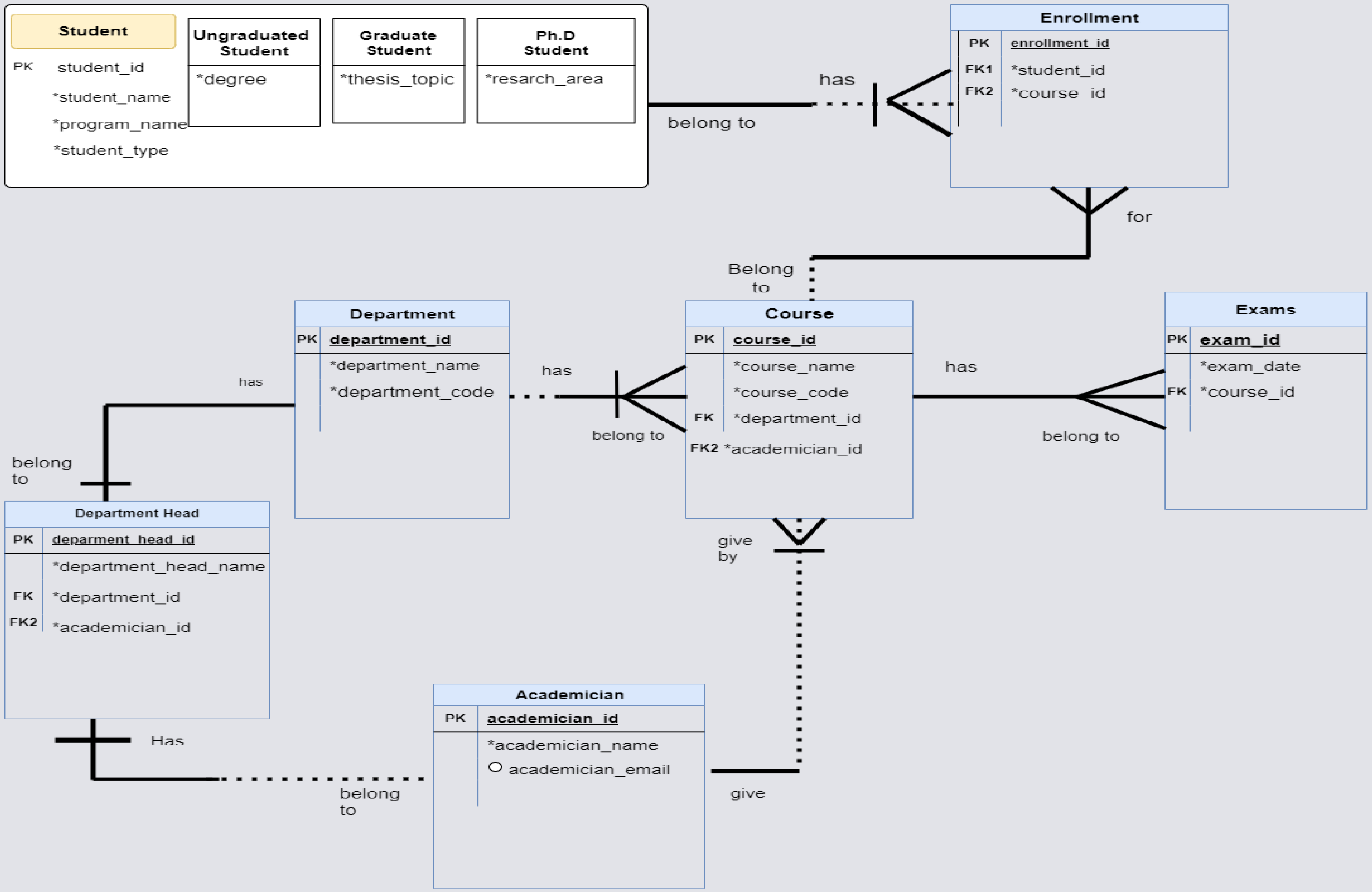
# COLLEGE DATABASE

ARİF TUNÇER 2110206029 AHMET KALE 2010206032 AHMET SEZER 2110206036

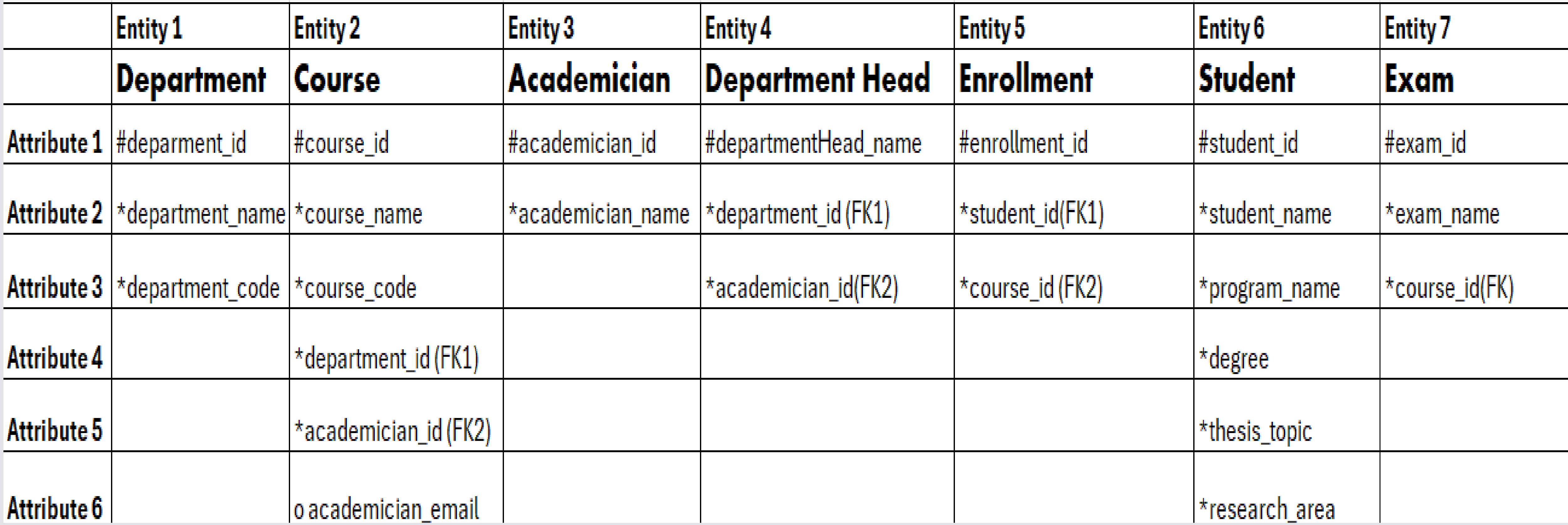
%30 2.Öğretim Group Number:4

## SCENARIO

Karabük University uses a comprehensive database system to manage its academic operations. This system meticulously records and tracks information related to students, academicians, departments, courses, and exams within the university. Students are a fundamental part of the university, and each student is registered with unique details such as student ID, name, program name, and student type. Students are categorized as undergraduate, graduate, and Ph.D. students. Undergraduate students have information on their "degree," graduate students have "thesis topic," and Ph.D. students have "research area." Students enroll in specific courses (Enrollment), and these courses are evaluated through exams (Exams) held on designated dates. Each course is offered under a specific department and has details such as course code, course name, and the ID of the academician teaching the course. Courses are taught by academicians, who are registered in the system with an academician ID and name. Academicians also work within departments and each department is managed by a department head. Department heads are identified with a department ID and an academician ID in the system. Departments are crucial organizational units within the university, each with a unique name and code. Each department is led by a head and staffed with academicians who offer various courses to students. Students register for courses, attend them, and participate in exams for these courses, thereby actively engaging in the university's educational process. Academicians, in addition to teaching courses, also play roles in the administrative operations of their respective departments, ensuring the academic and administrative structure of the university functions cohesively. This database system enables the university to manage all these components in an integrated manner, providing an effective educational environment for both students and academicians. The system is vital for tracking student achievements, evaluating academic performances, and continuously improving the educational processes.

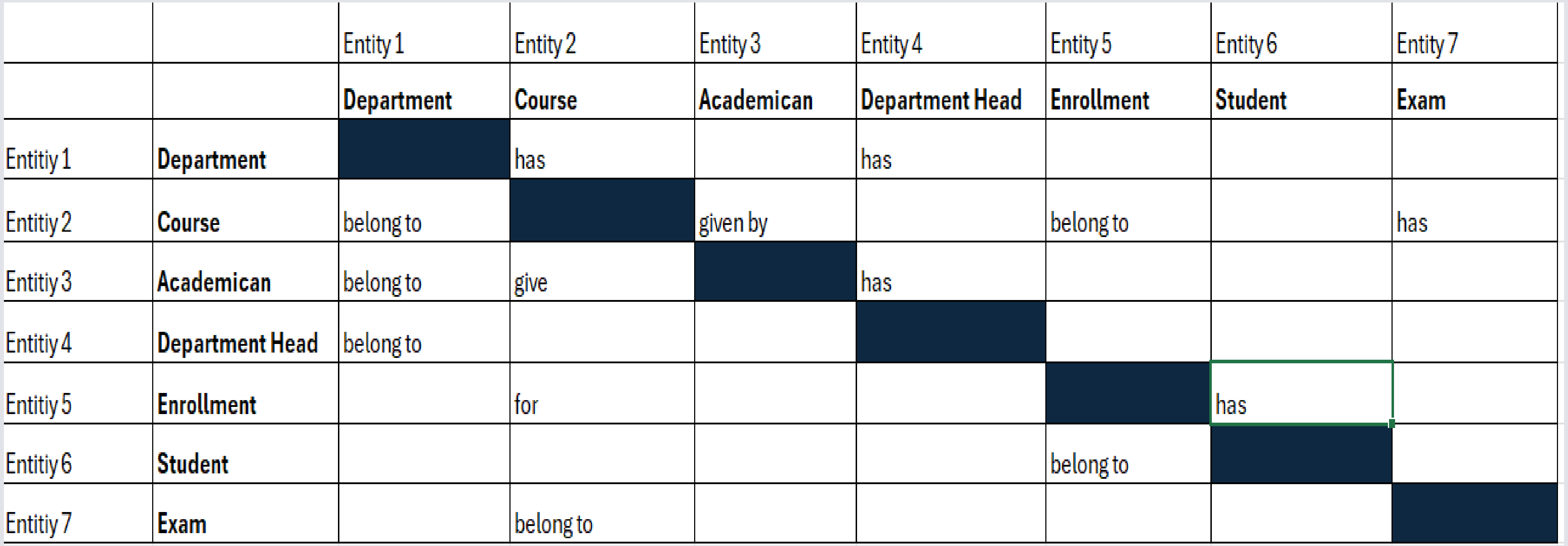

Student
PK student_id
*student_name
*program_name
*student_type
Ungraduated Student
*degree
Graduate Student
*thesis_topic
Ph.D Student
*resarch_area
has
belong to
Enrollment
PK enrollment_id
FK1 *student_id
FK2 *course_id
for
Belong to
Department
PK department_id
*department_name
*department_code
has
belong to
Course
PK course_id
*course_name
*course_code
FK *department_id
FK2 *academician_id
has
belong to
Exams
PK exam_id
*exam_date
FK *course_id
has
belong to
Department Head
PK deparment_head_id
*department_head_name
FK *department_id
FK2 *academician_id
give by
Has
belong to
Academician
PK academician_id
*academician_name
○ academician_email
give

# ENTITY- ATTRIBUTE DİAGRAM



| | Entity 1 | Entity 2 | Entity 3 | Entity 4 | Entity 5 | Entity 6 | Entity 7 |
|---|---|---|---|---|---|---|---|
| | **Department** | **Course** | **Academician** | **Department Head** | **Enrollment** | **Student** | **Exam** |
| **Attribute 1** | #deparment_id | #course_id | #academician_id | #departmentHead_name | #enrollment_id | #student_id | #exam_id |
| **Attribute 2** | *department_name | *course_name | *academician_name | *department_id (FK1) | *student_id(FK1) | *student_name | *exam_name |
| **Attribute 3** | *department_code | *course_code | | *academician_id(FK2) | *course_id (FK2) | *program_name | *course_id(FK) |
| **Attribute 4** | | *department_id (FK1) | | | | *degree | |
| **Attribute 5** | | *academician_id (FK2) | | | | *thesis_topic | |
| **Attribute 6** | | o academician_email | | | | *research_area | |

# MATRİX DİAGRAM



| | | Entity 1 | Entity 2 | Entity 3 | Entity 4 | Entity 5 | Entity 6 | Entity 7 |
|---|---|---|---|---|---|---|---|---|
| | | **Department** | **Course** | **Academican** | **Department Head** | **Enrollment** | **Student** | **Exam** |
| Entitiy 1 | **Department** | ■ | has | | has | | | |
| Entitiy 2 | **Course** | belong to | ■ | given by | | belong to | | has |
| Entitiy 3 | **Academican** | belong to | give | ■ | has | | | |
| Entitiy 4 | **Department Head** | belong to | | | ■ | | | |
| Entitiy 5 | **Enrollment** | | for | | | ■ | has | |
| Entitiy 6 | **Student** | | | | | belong to | ■ | |
| Entitiy 7 | **Exam** | | belong to | | | | | ■ |

## TABLES

# STUDENT

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| STUDENT_ID | PK | NUMBER(10,0) | | NO | 1 |
| STUDENT_NAME | | VARCHAR2(50) | | NO | Arif Tunçer |
| PROGRAM_NAME | | VARCHAR2(50) | | NO | Computer Engineering |
| STUDENT_TYPE | | VARCHAR2(50) | | NO | Ungrauated |
| | | | | | |
| | | | | | |

**DDL Statements:**

```
CREATE TYPE COLLEGE_STUDENT AS
OBJECT (
    student_id INT,
    student_name VARCHAR2(50),
    program_name VARCHAR2(50),
    student_type VARCHAR2(50),
);
```

```
CREATE TYPE Ungraduate UNDER Student (
    degree VARCHAR2(50)
);
CREATE TYPE Graduate UNDER Student (
    thesis_topic VARCHAR2(50)
);
CREATE TYPE Phd UNDER Student (
    research_area VARCHAR2(50)
```

# ACADEMICIAN

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| ACADEMICIAN_ID | PK | NUMBER(10,0) | | NO | 6 |
| ACADEMICIAN_NAME | | VARCHAR2(50) | | NO | Oğuz Fındık |

**DDL Statements:**

```
CREATE TABLE COLLEGE_Academician (
    Academician_id INT PRIMARY KEY,
    Academician_name VARCHAR(50)
);
```

```
INSERT INTO COLLEGE_Academician
(Academician_id, Academician_name) VALUES
(1, 'İlhami Muharrem Orak'),
(2, 'Kürşat Mustafa Karaoğlan'),
(3, 'Ferhat Atasoy'),
(4, 'Hüseyin Altınkaya'),
(5, 'Sezer Pıçak'),
(6, 'Oğuz Fındık'),
(7, 'Hasan Gökkaya'),
(8, 'Ozan Gülbudak'),
```

# COURSES

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| COURSE_ID | PK | | | NO | 2 |
| COURSE_NAME | | | | NO | DATABASE SYSTEM |
| COURSE_CODE | | | | NO | DS |
| DEPARTMENT_ID | FK | | DEPARTMENTS | NO | 1 |
| ACADEMICIAN_ID | FK | | ACADEMİCİANS | NO | 1 |

**DDL Statements:**

```
CREATE TABLE COLLEGE_Courses (
   Course_id INT PRIMARY KEY,
   Course_name VARCHAR(50),
   Course_code VARCHAR(50),
   Department_id INT,
   Academician_id INT,
) CONSTRAINT fk_department
      FOREIGN KEY (Department_id)
      REFERENCES Department(Department_id),
   CONSTRAINT fk_academician
      FOREIGN KEY (Academician_id)
      REFERENCES Academician(Academician_id)
);
```

# DEPARTMENTS

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| DEPARTMENT_ID | PK | NUMBER(10,0) | | NO | 2 |
| DEPARTMENT_NAME | | VARCHAR(50) | | NO | ELECTRİCS_ELECTRONİCS E. |
| DEPARTMENT_CODE | | VARCHAR2(50) | | NO | EE |

**DDL Statements:**

```
CREATE TABLE COLLEGE_Department (
    Department_id INT PRIMARY KEY,
    Department_name VARCHAR(50),
    Department_code VARCHAR(50)
);
```

```
INSERT INTO COLLEGE_Department (Department_id,
Department_name, Department_code) VALUES
(3, 'Mechanical Engineering', 'ME'),
(1, 'Computer Engineering', 'CE'),
(2, 'Electrics and Electronics Engineering', 'EE');
```

# DEPARTMENT_HEADS

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| DEPARTMENT_HEAD_ID | PK | NUMBER(10,0) | | NO | 2 |
| DEPARTMENT_HEAD_NAME | | VARCHAR2(50) | | NO | CE Head of department |
| DEPARTMENT_ID | FK | NUMBER(10,0) | DEPARTMENTS | NO | 1 |
| ACADEMICIAN_ID | FK | NUMBER(10,0) | ACADEMİCİAN | NO | 6 |

```
CREATE TABLE COLLEGE_DepartmentHead (
    Department_head_id INT PRIMARY KEY,
    Department_head_name VARCHAR(50),
    Department_id INT,
    Academician_id INT,
    CONSTRAINT fk_department_head_department
        FOREIGN KEY (Department_id)
        REFERENCES Department(Department_id),
    CONSTRAINT fk_department_head_academician
        FOREIGN KEY (Academician_id)
        REFERENCES Academician(Academician_id)
);
```

**DDL Statements:**

```
INSERT INTO  COLLEGE_DepartmentHead
(Department_head_id, Department_head_name,
Department_id, Academician_id) VALUES
(1, 'CE head of department', 1, 6),
(2, 'EE head of department', 2, 8),
(3, 'ME head of department', 3, 7);
```

# ENROLLMENTS

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| ENROLLMENT_ID | PK | NUMBER(10,0) | | NO | 3 |
| STUDENT_ID | FK | NUMBER(10,0) | STUDENT | NO | 2 |
| COURSE_ID | FK | NUMBER(10,0) | COURSES | NO | 1 |

```
CREATE TABLE COLLEGE_Enrollment (
    enrollment_id INT PRIMARY KEY,
    student_id INT,
    course_id INT,
    CONSTRAINT fk_enrollment_student
        FOREIGN KEY (student_id)
        REFERENCES Student(student_id),
    CONSTRAINT fk_enrollment_course
        FOREIGN KEY (course_id)
        REFERENCES Course(course_id)
);
```

**DDL Statements:**

```
INSERT INTO  COLLEGE_Enrollment (enrollment_id,
student_id, course_id) VALUES
(1, 1, 1),
(2, 1, 2),
(3, 2, 1),
(4, 2, 2);
```

# EXAMS

| COLUMN NAME | KEY TYPE | DATA TYPE | FK TABLE | NULLABLE | İNSTANCE |
|---|---|---|---|---|---|
| EXAM_ID | PK | NUMBER(10,0) | | NO | 3 |
| EXAM_DATE | | DATE | | NO | 05-Jun-2024 |
| COURSE_ID | FK | NUMBER(10,0) | COURSES | NO | 2 |

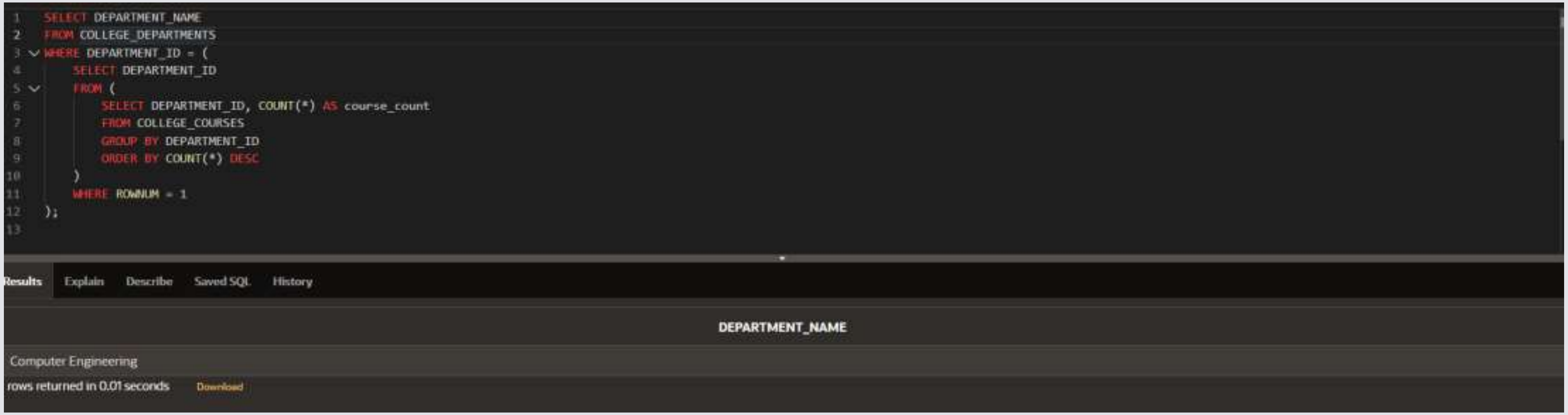
**DDL Statements:**

```
CREATE TABLE COLLEGE_Department (
    Department_id INT PRIMARY,
    Department_name VARCHAR(50),
    Department_code VARCHAR(50)
);
```

```
INSERT INTO Exams (exam_id, exam_date, course_id)
VALUES
(1, '2024-06-01', 1),
(2, '2024-06-02', 3),
(3, '2024-06-05', 2),
(4, '2024-05-29', 4),
(5, '2024-05-28', 5),
(6, '2024-06-09', 6);
```

## DML STATEMENTS

## SUBQUERY

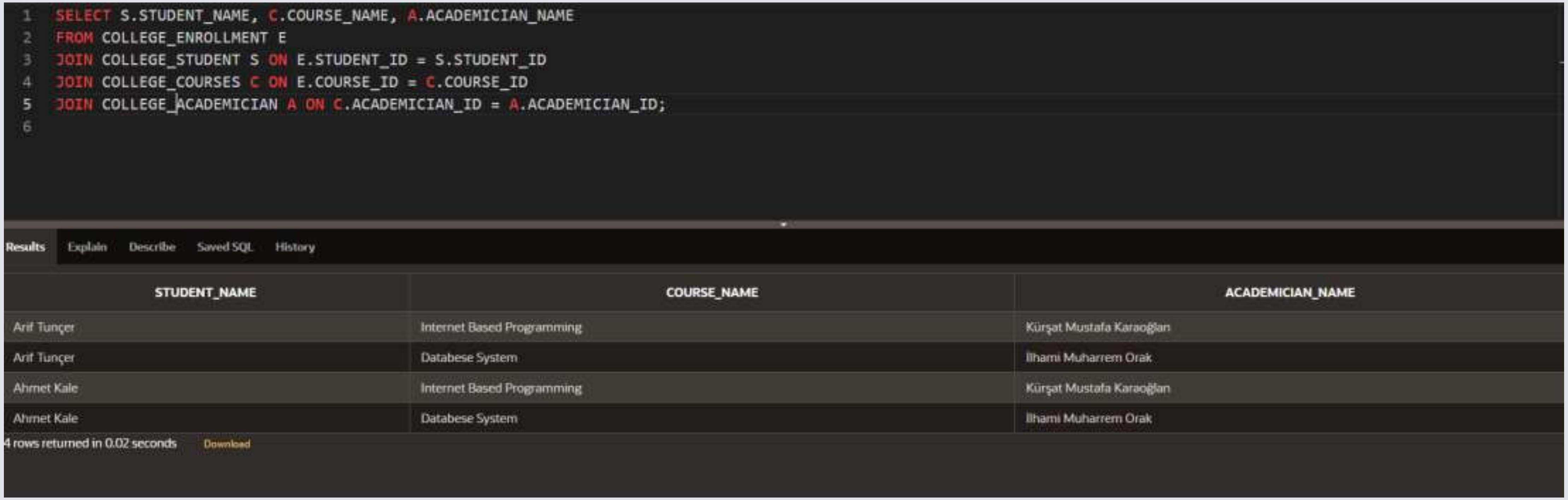


```
SELECT DEPARTMENT_NAME
FROM COLLEGE_DEPARTMENTS
WHERE DEPARTMENT_ID = (
    SELECT DEPARTMENT_ID
    FROM (
        SELECT DEPARTMENT_ID, COUNT(*) AS course_count
        FROM COLLEGE_COURSES
        GROUP BY DEPARTMENT_ID
        ORDER BY COUNT(*) DESC
    )
    WHERE ROWNUM = 1
);
```

| DEPARTMENT_NAME |
|---|
| Computer Engineering |

rows returned in 0.01 seconds

# JOİN

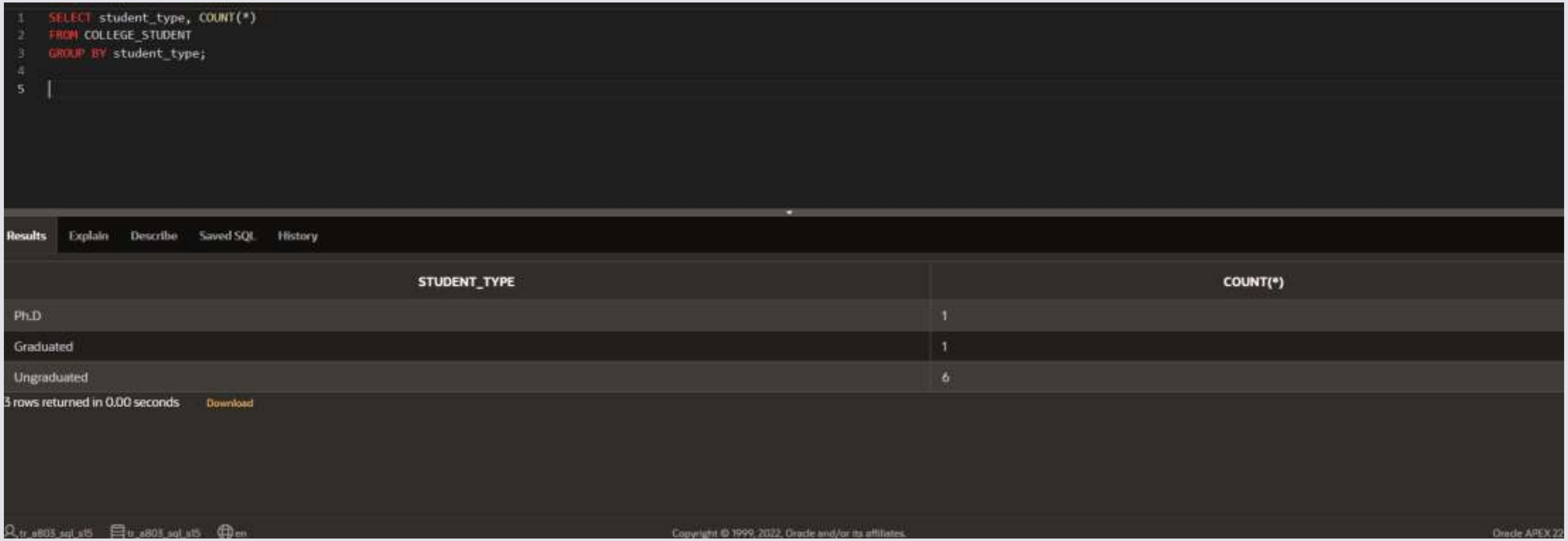


```
SELECT S.STUDENT_NAME, C.COURSE_NAME, A.ACADEMICIAN_NAME
FROM COLLEGE_ENROLLMENT E
JOIN COLLEGE_STUDENT S ON E.STUDENT_ID = S.STUDENT_ID
JOIN COLLEGE_COURSES C ON E.COURSE_ID = C.COURSE_ID
JOIN COLLEGE_ACADEMICIAN A ON C.ACADEMICIAN_ID = A.ACADEMICIAN_ID;
```

| STUDENT_NAME | COURSE_NAME | ACADEMICIAN_NAME |
|---|---|---|
| Arif Tunçer | Internet Based Programming | Kürşat Mustafa Karaoğlan |
| Arif Tunçer | Databese System | İlhami Muharrem Orak |
| Ahmet Kale | Internet Based Programming | Kürşat Mustafa Karaoğlan |
| Ahmet Kale | Databese System | İlhami Muharrem Orak |

4 rows returned in 0.02 seconds

## GROUP BY

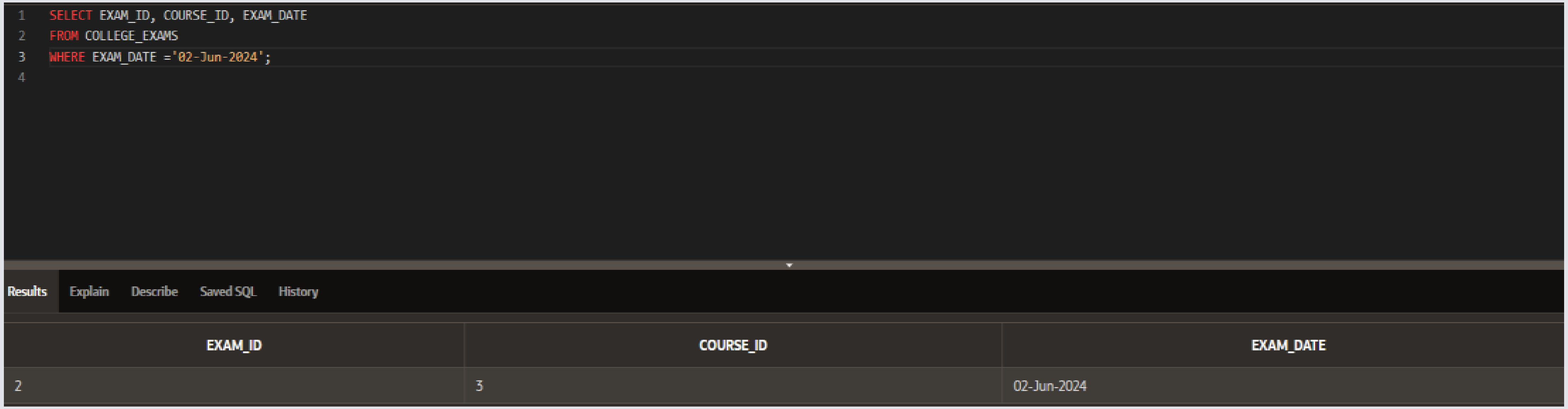


```sql
SELECT student_type, COUNT(*)
FROM COLLEGE_STUDENT
GROUP BY student_type;
```

| STUDENT_TYPE | COUNT(*) |
|---|---|
| Ph.D | 1 |
| Graduated | 1 |
| Ungraduated | 6 |

3 rows returned in 0.00 seconds

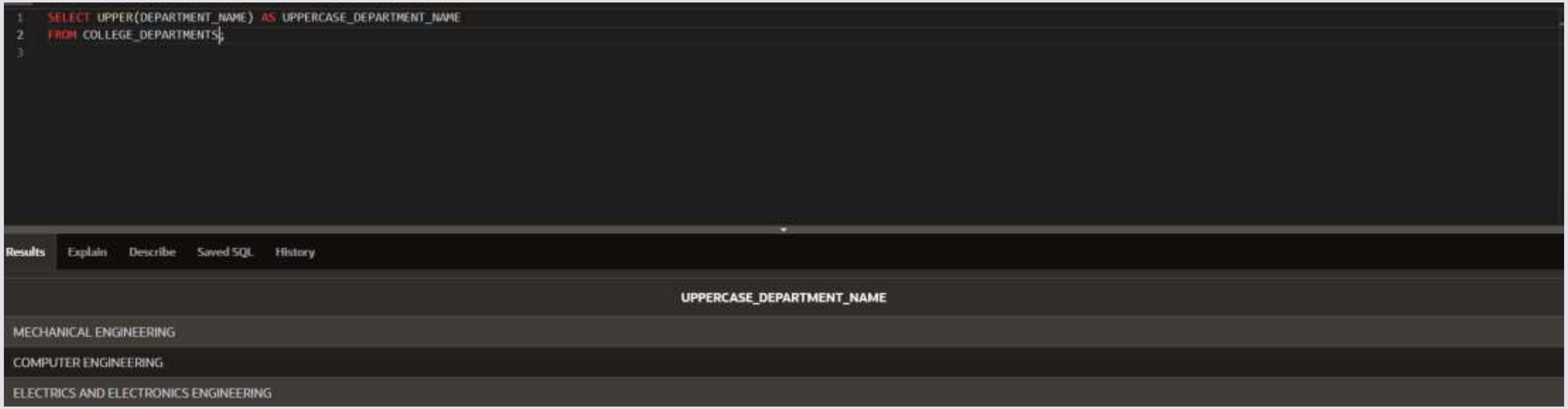
## DATE



```
SELECT EXAM_ID, COURSE_ID, EXAM_DATE
FROM COLLEGE_EXAMS
WHERE EXAM_DATE ='02-Jun-2024';
```

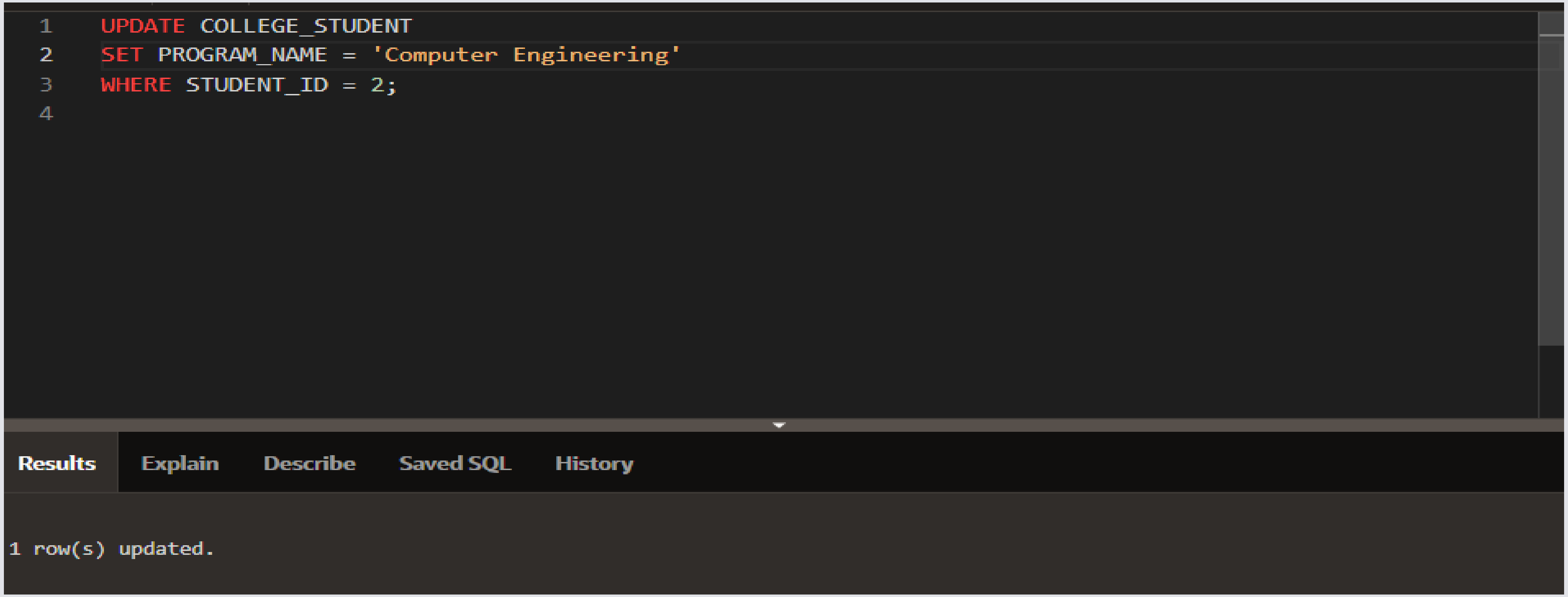
| EXAM_ID | COURSE_ID | EXAM_DATE |
|---|---|---|
| 2 | 3 | 02-Jun-2024 |

# CHARACTER


```
SELECT UPPER(DEPARTMENT_NAME) AS UPPERCASE_DEPARTMENT_NAME
FROM COLLEGE_DEPARTMENTS;
```

```
UPPERCASE_DEPARTMENT_NAME
MECHANICAL ENGINEERING
COMPUTER ENGINEERING
ELECTRICS AND ELECTRONICS ENGINEERING
```

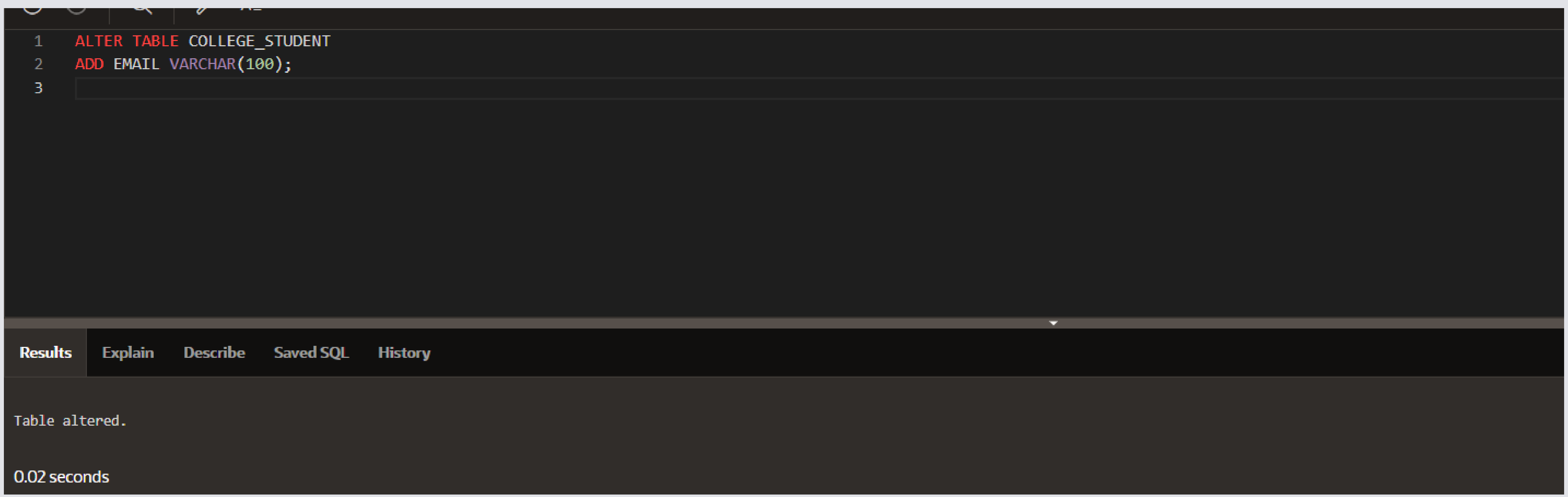
## UPDATE



```sql
UPDATE COLLEGE_STUDENT
SET PROGRAM_NAME = 'Computer Engineering'
WHERE STUDENT_ID = 2;
```

```
1 row(s) updated.
```

# ALTER



```
ALTER TABLE COLLEGE_STUDENT
ADD EMAIL VARCHAR(100);
```

```
Table altered.
```

0.02 seconds