# RESEARCH & PROJECT

**Program:** **Computer Engineering and Software Systems- CESS**

*Course Code: CSE 227*
*Course Name: Database Systems (1)*

*Examination Committee*
**Prof. Dr. Hoda Korashy Mohamed**

**Ain Shams University**
**Faculty of Engineering**
**Spring Semester – 2020**

## Student Personal Information for Group Work

| **Student Names:** | **Student Codes:** |
|---|---|
| Moaz Mohamed Abd Elaziz | 17P1023 |
| Abdelrahman Mahmoud Fangary | 17P6006 |

## Plagiarism Statement

I certify that this assignment / report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they are books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment / report has not been previously been submitted for assessment for another course. I certify that I have not copied in part or whole or otherwise plagiarized the work of other

| | | |
|---|---|---|
| **Signature/Student** | **Moaz Mohamed Abd Elaziz** | **Date: 5/6/2020** |
| **Signature/Student** | **Abdelrahman Mahmoud Fangary** | **Date 5/6/2020** |

## Submission Contents

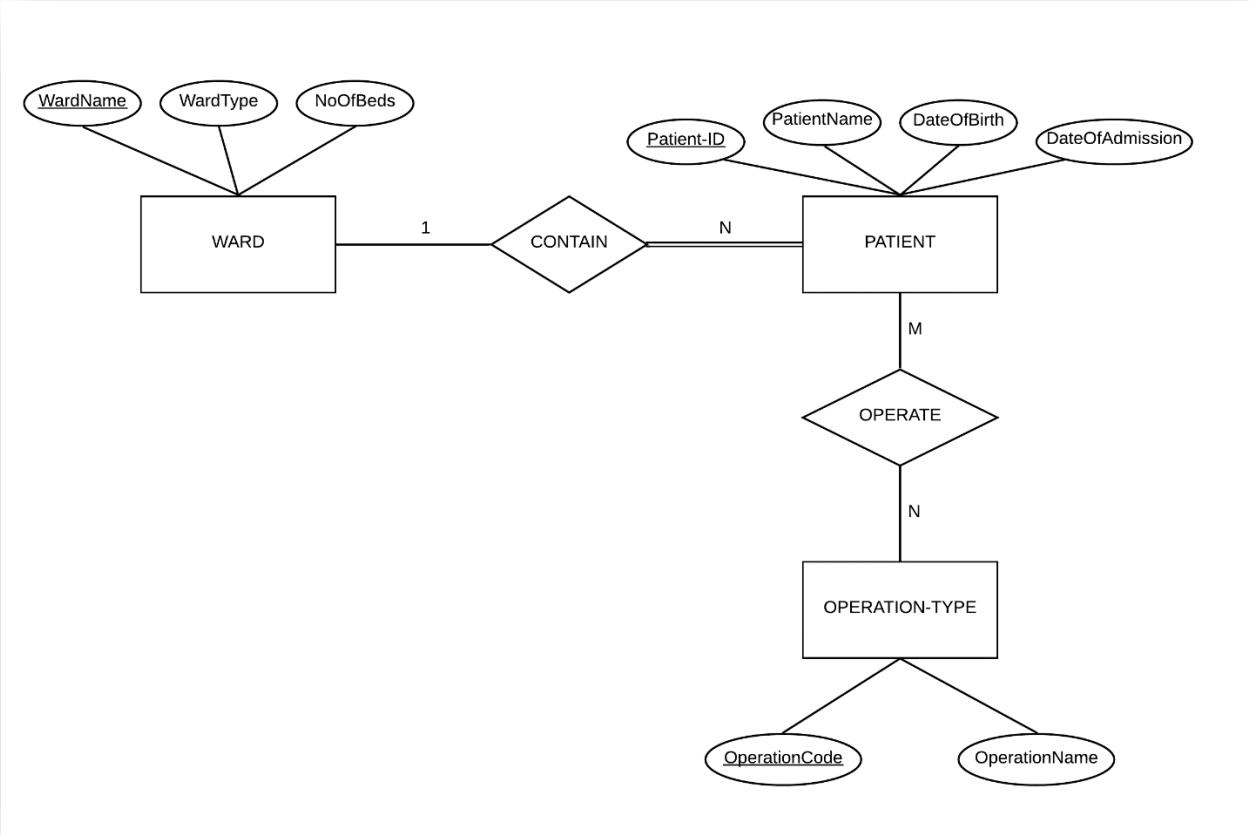

**01: First Section Title**
**02: Second Section Title**
**03: Third Section Title**
**04: Fourth Section Title**
**05: Fifth Section Title**

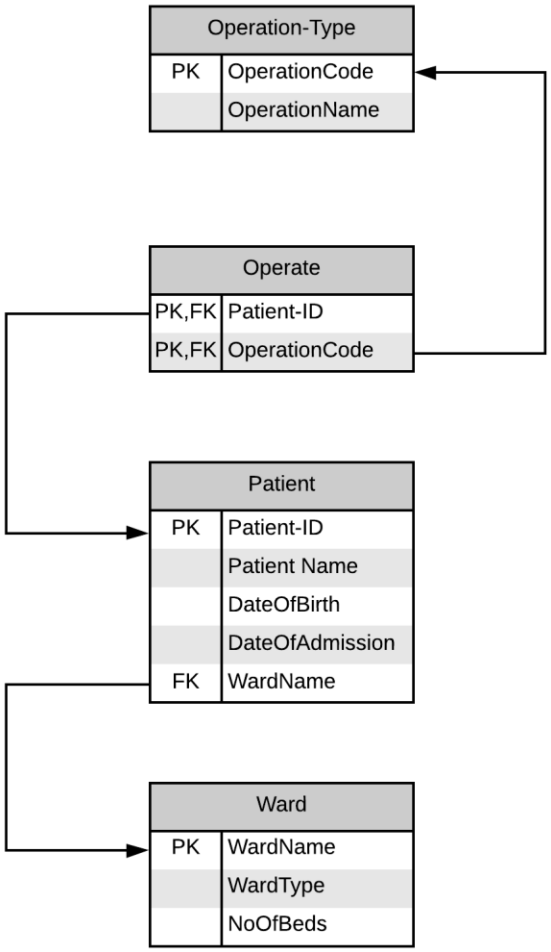
# Table of Contents

Part 1: Assignment Questions ..... 4
Part 2: University Database Design Project ..... 10
1. Introduction ..... 10
2. Important data and reports ..... 10
3. Assumptions ..... 15
4. EER Diagram ..... 16
5. Database Schema ..... 17
6. Sample of SQL ..... 18
7. Implementation ..... 25
8. References ..... 33

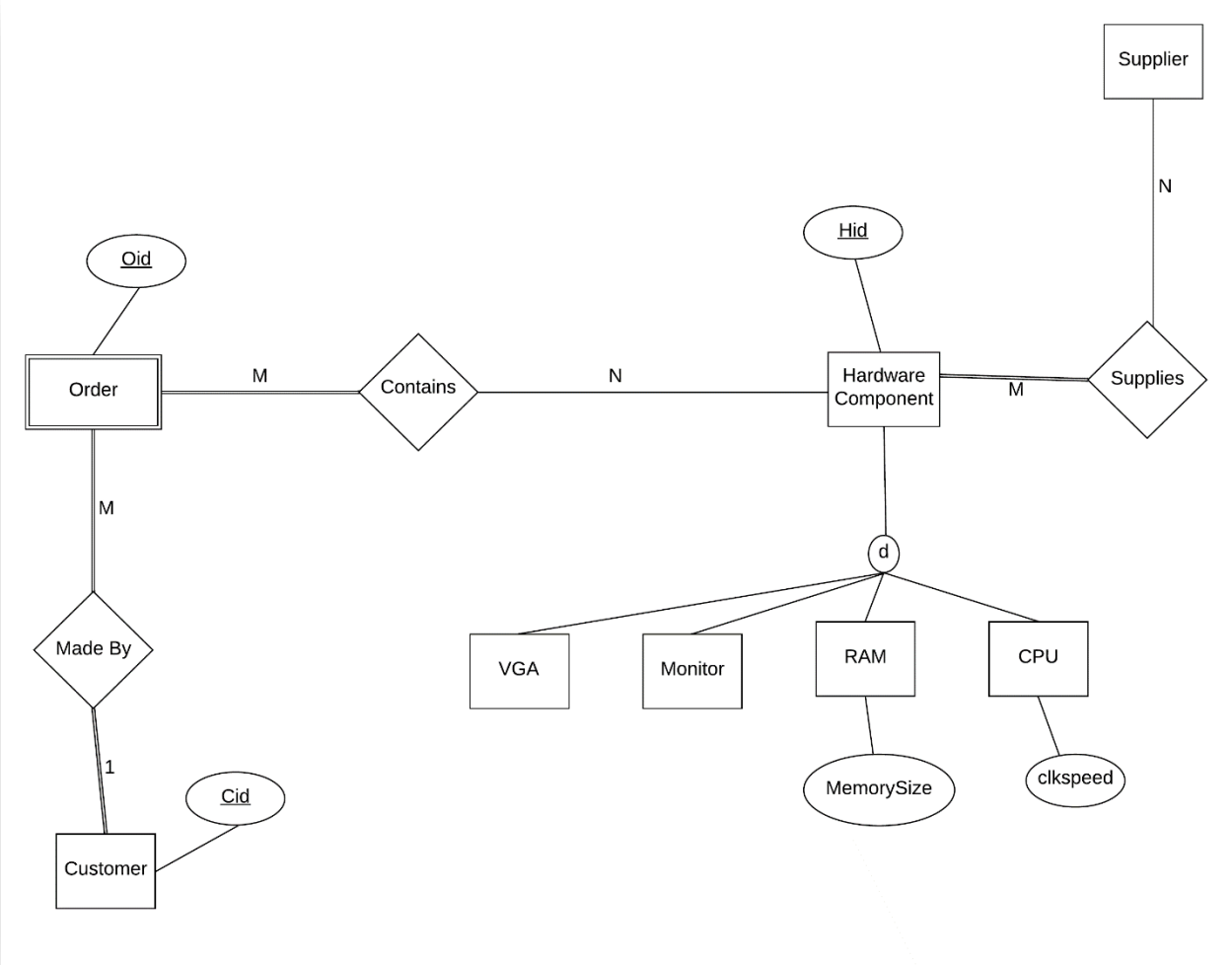
# Part 1: Assignment Questions

## Q1)



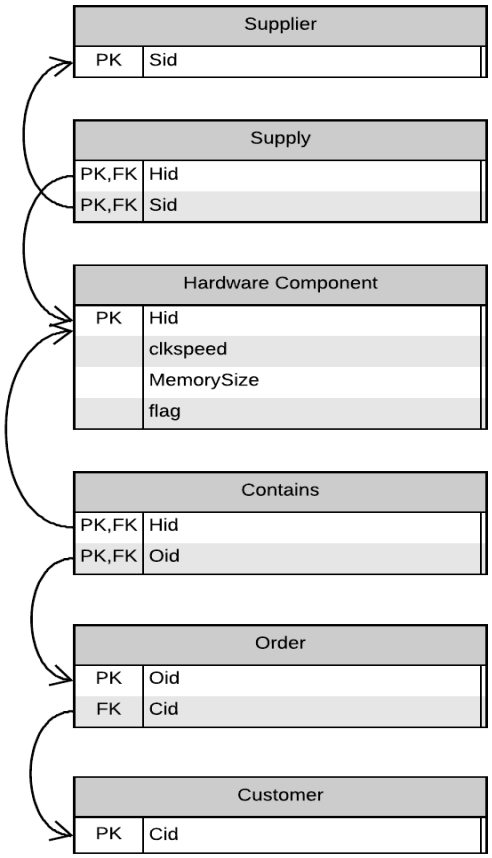

Operation-Type
PK
OperationCode
OperationName
Operate
PK,FK
Patient-ID
PK,FK
OperationCode
Patient
PK
Patient-ID
Patient Name
DateOfBirth
DateOfAdmission
FK
WardName
Ward
PK
WardName
WardType
NoOfBeds

**Q2)**




**Assumptions:**

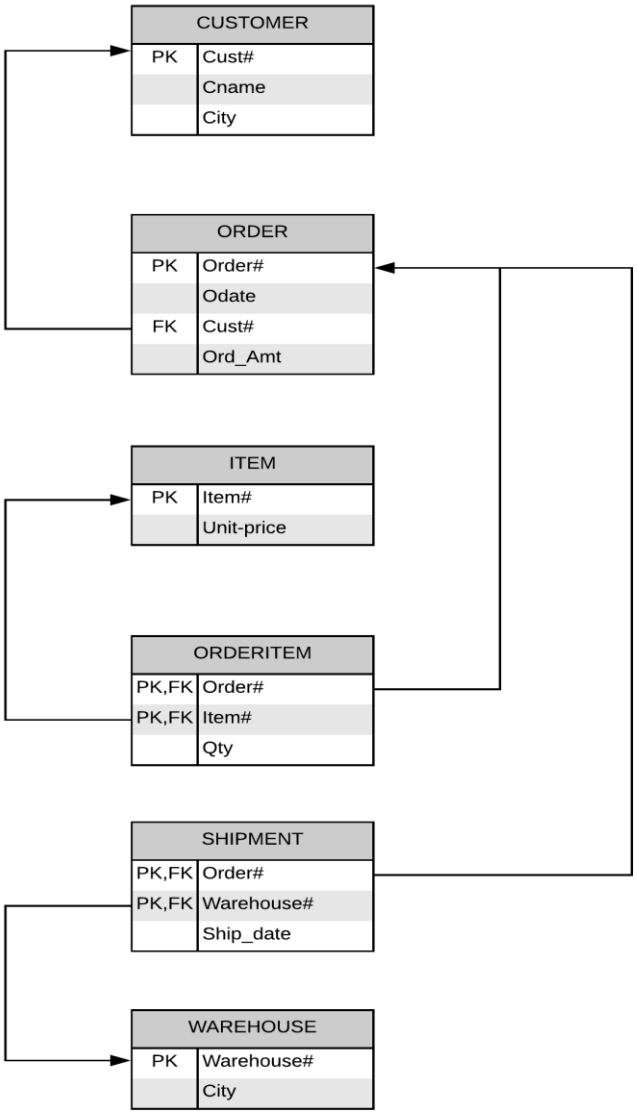
- Assume H.W components (VGA, Monitor, RAM, CPU) are entities with attributes
- Assume partial participation in Hardware Component inheritance.
- Assume supplier may supply many H.W Components.
- Assume that one H.W component must be supplied by one or more supplier.
- Assume that the customer who is stored in database, made orders.
- One order must be made by one customer, and customer must make at least one order


Supplier
PK Sid
Supply
PK,FK Hid
PK,FK Sid
Hardware Component
PK Hid
clkspeed
MemorySize
flag
Contains
PK,FK Hid
PK,FK Oid
Order
PK Oid
FK Cid
Customer
PK Cid

**Q3)**

**<u>Foreign Keys:</u>**

- The attribute Cust# in ORDER relation references CUSTOMER relation
- The attribute Oder# in ORDERITEM relation references ORDER relation
- The attribute Item# in ORDERITEM relation references ITEM relation
- The attribute Order# in SHIPMENT relation references ORDER relation
- The attribute Warehouse# in SHIPMENT relation references WAREHOUSE relation



**Q4)**

**a)**

```
select FNAME, LNAME, BDATE
from EMPLOYEE E, DEPARTMENT D
 where E.DNO = D.DNO AND DNAME = ‘Marketing’
```

**b)**

```
select DNAME
from DEPARTMENT
where BUDGET > 5000
```

**c)**

```
select E.FNAME, E.LNAME
 from EMPLOYEE E, PROJECT P, WORKS-ON W
 where E.ESSN = W.ESSN AND P.PNO = W.PNO
 AND W.HOURS > 10 AND P.PNAME = ‘PRODUCTX’ AND E.DNO = 5
```

## Part 2: University Database Design Project

### 1. Introduction

The chosen domain is ***"University Database"***.

The database stores information about students, university departments, employees, courses offered, courses' rooms, trainings, projects, and employees' offices. There are three types of employees: manager, advisor, and lecturer. Employees have offices.

Each department is managed by one manager, and each department offer some courses, these courses are taught by lecturers and held in rooms. Each department provide trainings.

Students takes courses and they also are advised by advisors. Students may work on projects that are supervised by lecturers. Students take trainings offered by departments.

### 2. Important data and reports

**Important data:**

Student: S_ID, FirstName, LastName, Email, Gender, Telephone, GPA

Employee: E_ID, FirstName, LastName, Gender, Email, Salary, Telephone

Lecturer: Ranking

Advisor: Sessions

Manager: Secretary

Course: C_ID, Name, CreditHours

Departement: D_ID, Name, NoOfEmployees, Telephone

Project: P_ID, Name,  StartingDate, EndingDate, Budget

**Reports:**

1. Get all students who took Programming 1 whose grade is greater than 3.3.

```
select s.S_ID,s.firstName,s.LastName ,t.grade
from Student s, Takes t ,course c
where t.Grade > 3.3 AND c.Name = "Programming 1" AND t.C_ID = c.C_ID AND t.S_ID =s.S_ID;
```

Result Grid

| S_ID | firstName | LastName | grade |
|---|---|---|---|
| 4 | Mohamed | Hashem | 3.5 |

2. Retrieve the names of students who took neural network (used nested queries)

```
select s.S_ID,s.firstName,s.LastName
  from Student s
  where s.S_ID IN ( select t.S_ID
                    from Takes t , Course c
                    where c.Name = "Neural Network" AND t.C_ID = c.C_ID
                  );
```

Result Grid

| S_ID | firstName | LastName |
|---|---|---|
| 2 | Abdelrahman | Fangary |
| 4 | Mohamed | Hashem |
| NULL | NULL | NULL |

3. For each department count all course having credit hours = 3

```
select d.Name, COUNT(*)
FROM Department d, Course c
WHERE c.DPRT_ID = d.D_ID AND
c.CreditHours =3
GROUP BY c.DPRT_ID
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Name | COUNT(*) |
|---|---|
| Building Engineering | 1 |
| Computer Engineering | 2 |

4. Show each advisor and all students he advises

```
select e.FirstName AS AdvisorFirstName, e.LastName AS AdvisorLasttName, s.FirstName AS StudentFirstName, s.LastName AS StudentLastName
From Advisor a, Employee e, Student s
where a.Advisor_ID = e.E_ID AND s.Advisor_ID = a.Advisor_ID
order by a.Advisor_ID
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| AdvisorFirstName | AdvisorLasttName | StudentFirstName | StudentLastName |
|---|---|---|---|
| Ahmed | Khalid | Abdelrahman | Fangary |
| Ahmed | Khalid | Mohamed | Hashem |
| Ahmed | Khalid | Ahmed | Tarek |
| Ahmed | Shaker | Moaz | Mohamed |
| Ahmed | Shaker | Mahmoud | Mohamed |

5. Show all the projects with the students work on and lecturers supervise

```
Select p.P_ID , p.Name , p.Budget, e.FirstName , e.LastName , st.FirstName , st.LastName
From Project p , Supervise s, Works_On w, Lecturer l , Student st , Employee e
where p.P_ID = w.P_ID AND  p.P_ID = s.P_ID AND st.S_ID = w.S_ID AND l.Lecturer_ID = s.Lecturer_ID  AND l.Lecturer_ID = e.E_ID
```

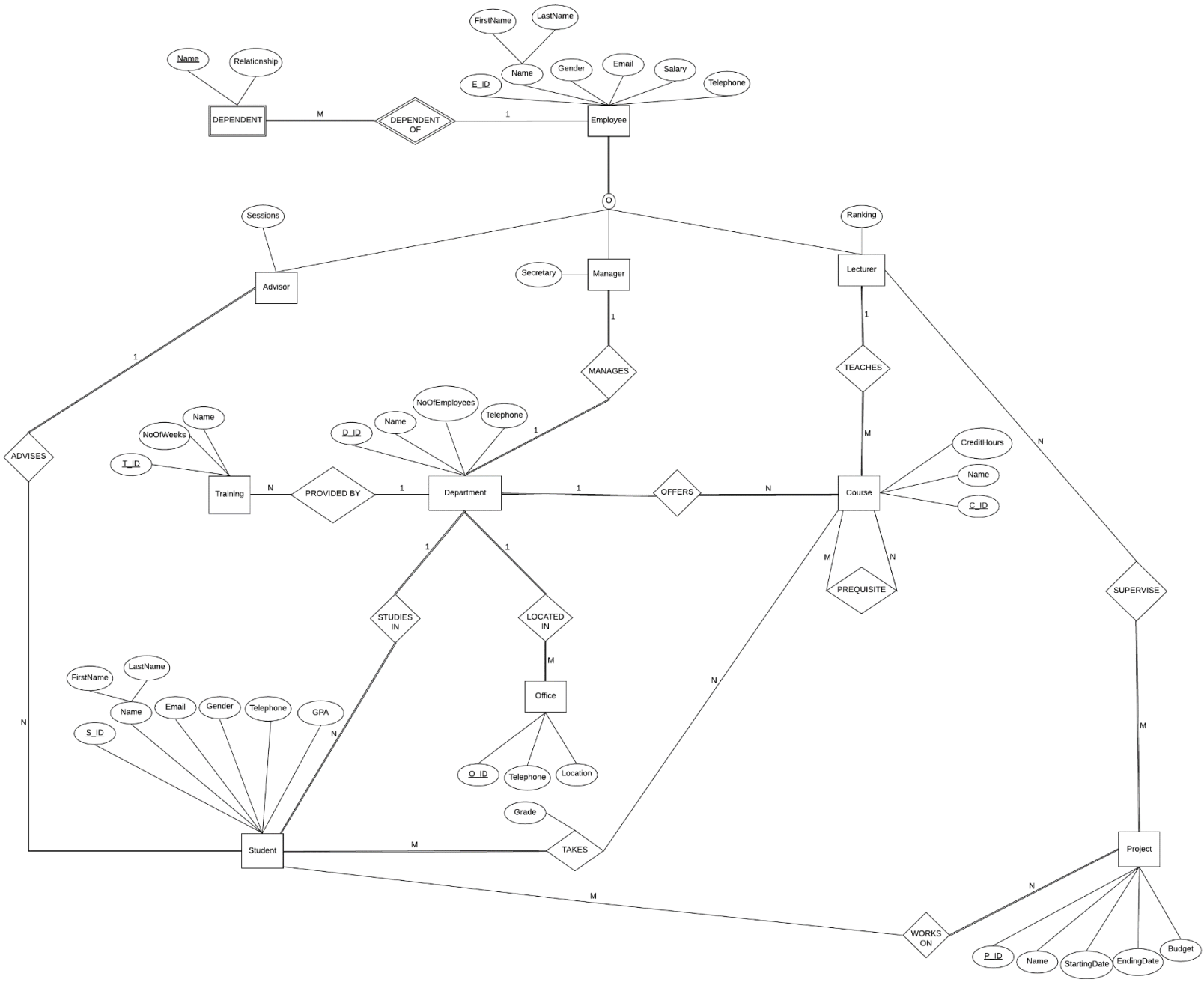
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| P_ID | Name | Budget | FirstName | LastName | FirstName | LastName |
|---|---|---|---|---|---|---|
| 1 | DB | 200 | Ahmed | Taher | Abdelrahman | Fangary |
| 1 | DB | 200 | Ahmed | Taher | Mahmoud | Mohamed |
| 2 | Concrete Design | 1000 | Ziad | Tarek | Moaz | Mohamed |

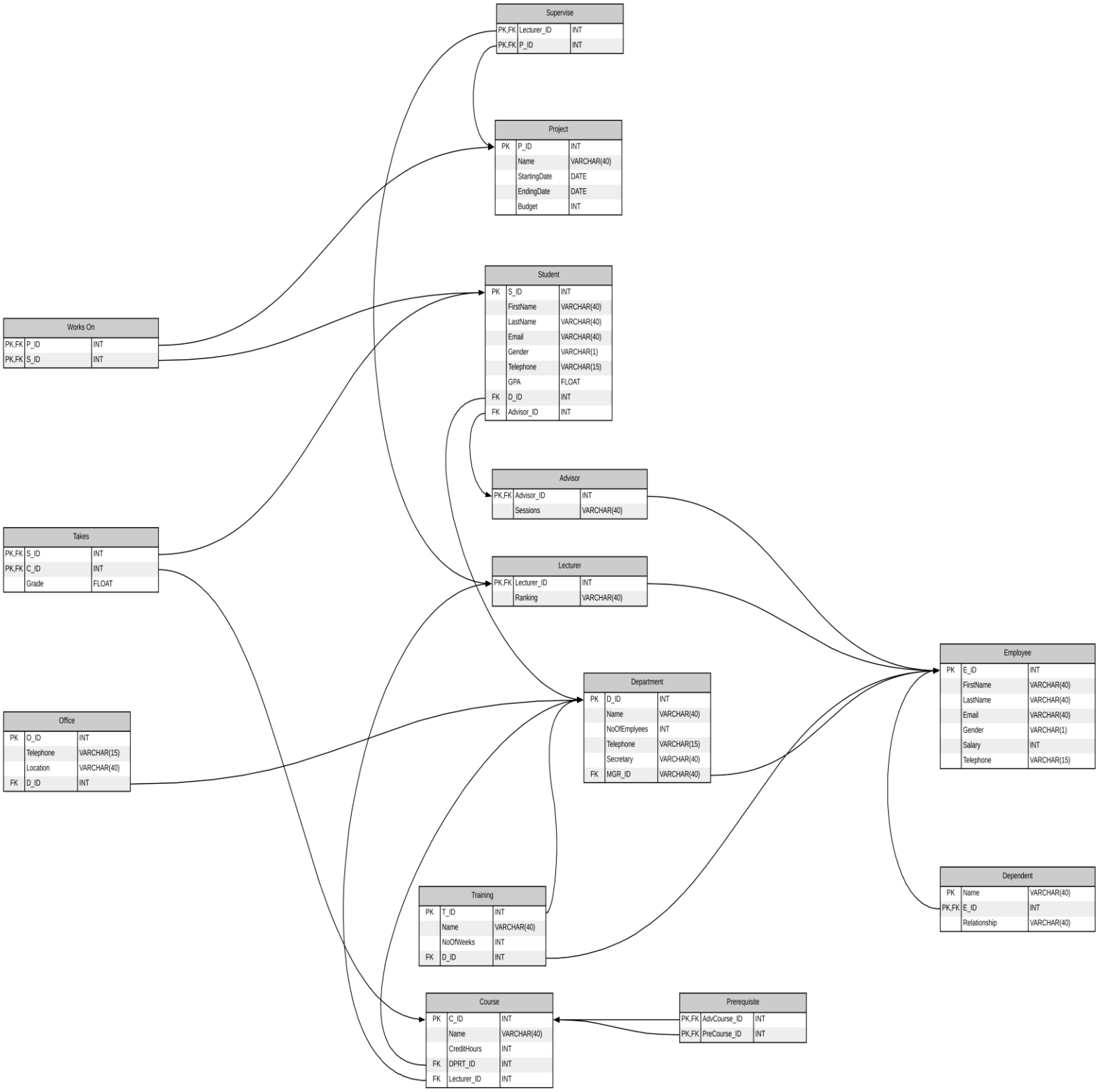
## 3. Assumptions

- Student must take at least one course, course may be taken by one or more students.
- Department must offer many courses, courses must be offered by only one department.
- Department must provide many trainings, training must be provided by one department.
- Student must study in one department, one department must be studied in by many students.
- Manager must manage one department, department is managed by one manager.
- Employee must be Manager or advisor or lecturer or mix of them.
- Instructor must teach many courses, course must be taught by one.
- lecturer may supervise zero or many projects, project must be supervised by one or many.
- Advisor must advise one or more students, student must be advised by one advisor.
- Student may work on zero or more projects, project must be worked by one or more students.
- Department must be located in one office, office must be allocated to one department.
- course may prerequisite zero or many courses, prerequisite courses may be prerequisite by zero or many.

# 4. EER Diagram



# 5. Database Schema



## 6. Sample of SQL

**Creation**:

```
CREATE TABLE Employee (
  E_ID  INT NOT NULL,
  FirstName VARCHAR(40) NOT NULL,
  LastName VARCHAR(40) NOT NULL,
  Email VARCHAR(40) NOT NULL,
  Gender VARCHAR(1) NOT NULL,
  Salary  INT ,
  Telephone VARCHAR(15) NOT NULL,
  PRIMARY KEY (E_ID)

);
```

---

```
CREATE TABLE Advisor (
  Advisor_ID  INT NOT NULL,
  Sessions VARCHAR(40) ,
  PRIMARY KEY (Advisor_ID),
  FOREIGN KEY (Advisor_ID) REFERENCES Employee (E_ID)

  ON DELETE CASCADE
  ON UPDATE CASCADE

);
```

---

```
CREATE TABLE Lecturer (
  Lecturer_ID INT,
  Ranking VARCHAR(40),
  PRIMARY KEY (Lecturer_ID),
  FOREIGN KEY (Lecturer_ID) REFERENCES Employee(E_ID)
  ON DELETE CASCADE
  ON UPDATE CASCADE
);
```

---

```
CREATE TABLE Department (
  D_ID INT,
  Name VARCHAR(40),
  NoOfEmplyees INT,
  Telephone VARCHAR(15),
  Secretary VARCHAR(40),
  MGR_ID INT,
  PRIMARY KEY (D_ID),
  FOREIGN KEY (MGR_ID) REFERENCES Employee(E_ID)
  ON DELETE SET NULL
  ON UPDATE RESTRICT
);
```

```sql
CREATE TABLE Student (
  S_ID INT,
  FirstName VARCHAR(40) NOT NULL,
  LastName VARCHAR(40) NOT NULL,
  Email VARCHAR(40),
  Gender VARCHAR(1),
  Telephone VARCHAR(15),
  GPA FLOAT,
  D_ID INT,
  Advisor_ID INT,
  PRIMARY KEY (S_ID),
  FOREIGN KEY (D_ID) REFERENCES Department(D_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT,
  FOREIGN KEY (Advisor_ID) REFERENCES Advisor(Advisor_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

---

```sql
 CREATE TABLE Course (
  C_ID INT,
  Name VARCHAR(40) NOT NULL,
  CreditHours INT NOT NULL,
  DPRT_ID INT ,
  Lecturer_ID INT,
  PRIMARY KEY (C_ID),
  FOREIGN KEY (DPRT_ID) REFERENCES Department(D_ID)
  ON DELETE CASCADE
  ON UPDATE CASCADE,
  FOREIGN KEY (Lecturer_ID) REFERENCES Lecturer(Lecturer_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
CREATE TABLE Takes (
  S_ID INT,
  C_ID INT,
  Grade FLOAT NOT NULL,
  PRIMARY KEY (S_ID, C_ID),
  FOREIGN KEY (S_ID) REFERENCES Student(S_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT,
  FOREIGN KEY (C_ID) REFERENCES Course(C_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

---

```sql
CREATE TABLE Project (
  P_ID  INT NOT NULL,
  Name VARCHAR(40) NOT NULL,
  StartingDate DATE NOT NULL,
  EndingDate DATE NOT NULL,
  Budget  INT ,
  PRIMARY KEY (P_ID)
);
```

---

```sql
CREATE TABLE Office (
  O_ID  INT NOT NULL,
  Telephone VARCHAR(15) NOT NULL,
  Location VARCHAR(40)NOT NULL,
  DPRT_ID  INT NOT NULL,
  PRIMARY KEY (O_ID),
  FOREIGN KEY (DPRT_ID) REFERENCES Department (D_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

---

```sql
CREATE TABLE Dependent (
  Name VARCHAR(40) NOT NULL,
  E_ID  INT NOT NULL,
  Relationship VARCHAR(40),
  PRIMARY KEY (Name,E_ID),
  FOREIGN KEY (E_ID) REFERENCES Employee (E_ID)
  ON DELETE CASCADE
  ON UPDATE CASCADE
);
```

---

```sql
CREATE TABLE Training (
  T_ID  INT NOT NULL,
  Name VARCHAR(40)NOT NULL,
  NoOfWeeks  INT,
  DPRT_ID  INT NOT NULL,
  PRIMARY KEY (T_ID),
  FOREIGN KEY (DPRT_ID) REFERENCES Department (D_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

---

```sql
 CREATE TABLE Supervise (
  Lecturer_ID INT,
  P_ID INT,
  PRIMARY KEY (Lecturer_ID, P_ID),
  FOREIGN KEY (Lecturer_ID) REFERENCES Lecturer(Lecturer_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT,
  FOREIGN KEY (P_ID) REFERENCES Project(P_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

```sql
CREATE TABLE Prerequisite (
  AdvCourse_ID INT,
  PreCourse_ID INT,
  PRIMARY KEY (AdvCourse_ID, PreCourse_ID),
  FOREIGN KEY (AdvCourse_ID) REFERENCES Course(C_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT,
  FOREIGN KEY (PreCourse_ID) REFERENCES Course(C_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

---

```sql
 CREATE TABLE WorksOn (
  P_ID INT,
  S_ID INT,
  PRIMARY KEY (P_ID, S_ID),
  FOREIGN KEY (P_ID) REFERENCES Project(P_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT,
  FOREIGN KEY (S_ID) REFERENCES Student(S_ID)
  ON DELETE RESTRICT
  ON UPDATE RESTRICT
);
```

---

## Insertion:

```
insert into Student
values(1,"Moaz","Mohamed","moaz@student.com","M","0123456789","3.1",2,6);
```

---

```
insert into Department
values(1,"Computer Engineering",40,"01153422356","Yasmin Mohamed",4);
```

---

```
insert into course
values(1,"Neural Network",4,1,1);
```

---

```
insert into training
values(100,"CCNA Training",5,1);
```

---

```
insert into Employee
values(2, "Ziad","Tarek","ziadtarek@gmail.com","M",2500,"01342234544");
```

---

```
insert into lecturer
values(2," Associate Professor");
```

---

```
insert into prerequisite
values(4,3);
```

---

## Update:

```
update Student
set Advisor_ID = 3
where S_ID =3;
```

---

```
update Office
set Location = "Building 2 Room 500"
where O_ID =1001;
```

---

## Delete:

```
Delete from Student
where S_ID = 3;
```

---

```
Delete from course
where C_ID=5;
```

---

**Retrieve:**

The retrieval reports are screenshotted above.

```
Select s.S_ID,s.firstName,s.LastName ,t.grade
From Student s, Takes t ,course c
where t.Grade > 3.3 AND c.Name = "Programming 1" AND t.C_ID = c.C_ID AND
t.S_ID =s.S_ID;
```

---

```
select s.S_ID,s.firstName,s.LastName
 From Student s
 where s.S_ID IN ( select t.S_ID
                   from Takes t , Course c
                   where c.Name = "Neural Network" AND t.C_ID = c.C_ID
                  );
```

---

```
Select d.Name, COUNT(*)
From Department d, Course c
Where c.DPRT_ID = d.D_ID AND
c.CreditHours =3
GROUP BY c.DPRT_ID
```

---

```
Select e.FirstName AS AdvisorFirstName, e.LastName AS AdvisorLasttName,
s.FirstName AS StudentFirstName, s.LastName AS StudentLastName
From Advisor a, Employee e, Student s
where a.Advisor_ID = e.E_ID AND s.Advisor_ID = a.Advisor_ID
order by a.Advisor_ID
```

---

```
Select p.P_ID , p.Name , p.Budget, e.FirstName , e.LastName , st.FirstName ,
st.LastName
From Project p , Supervise s, Works_On w, Lecturer l , Student st , Employee
e
where p.P_ID = w.P_ID AND  p.P_ID = s.P_ID AND st.S_ID = w.S_ID AND
l.Lecturer_ID = s.Lecturer_ID  AND l.Lecturer_ID = e.E_ID
```

---

# 7. Implementation

## 7.1 Part1: ERD Tool

We used **ERDPlus** tool: an online database modeling tool to quickly and easily create Entity Relationship Diagrams, Relational Schemas.

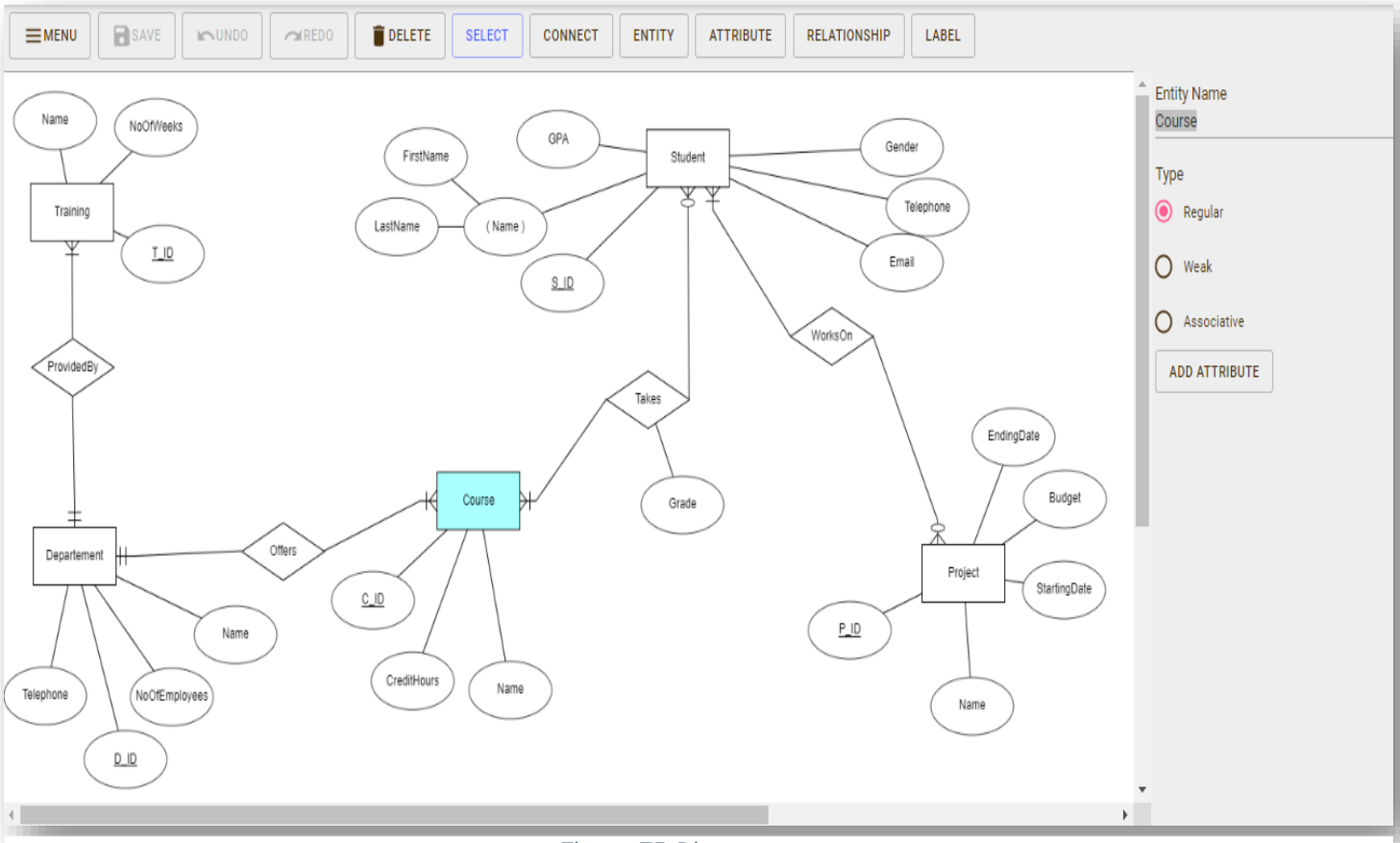



*Figure: ER Diagram*

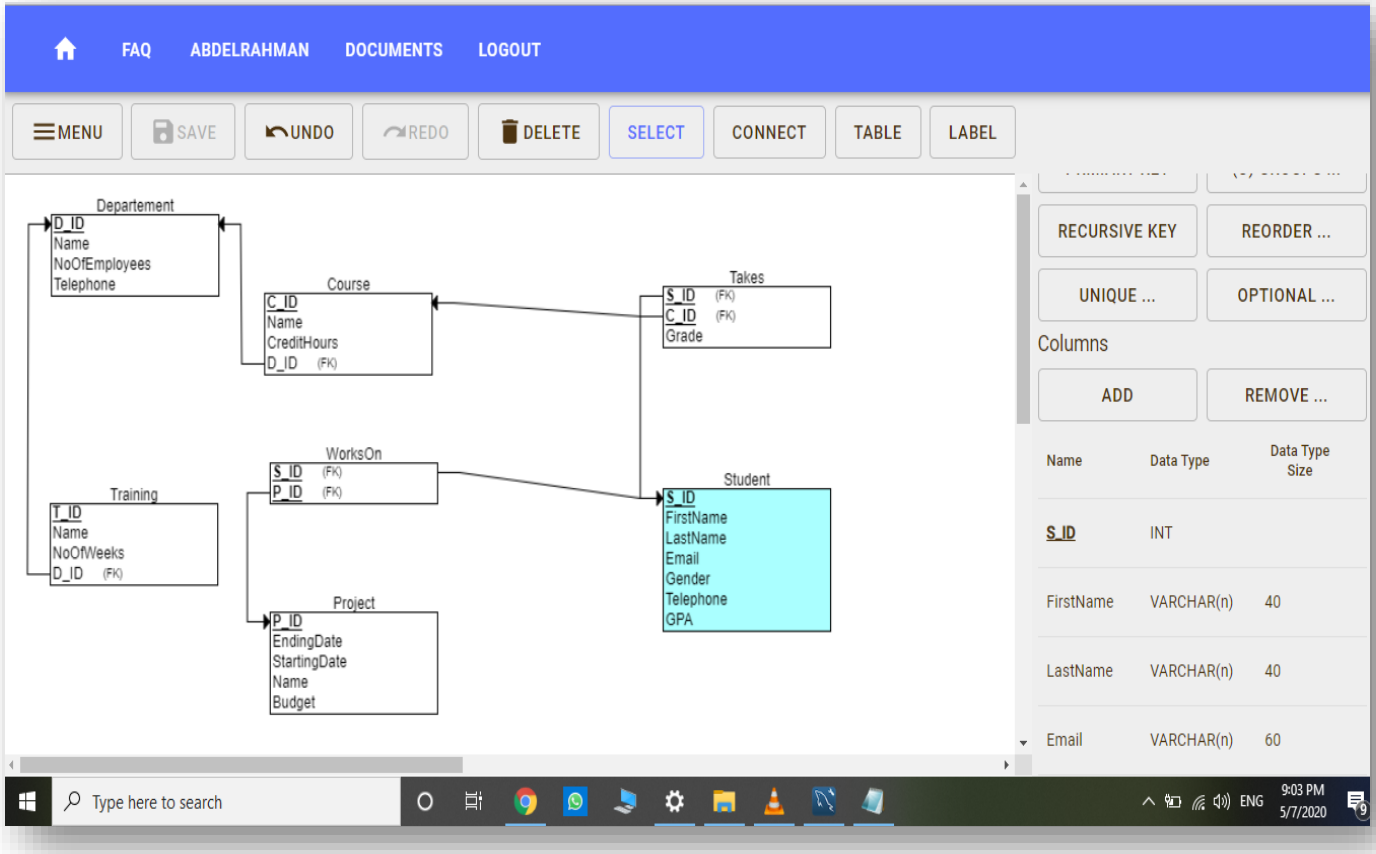



*Figure: Relational Schema*

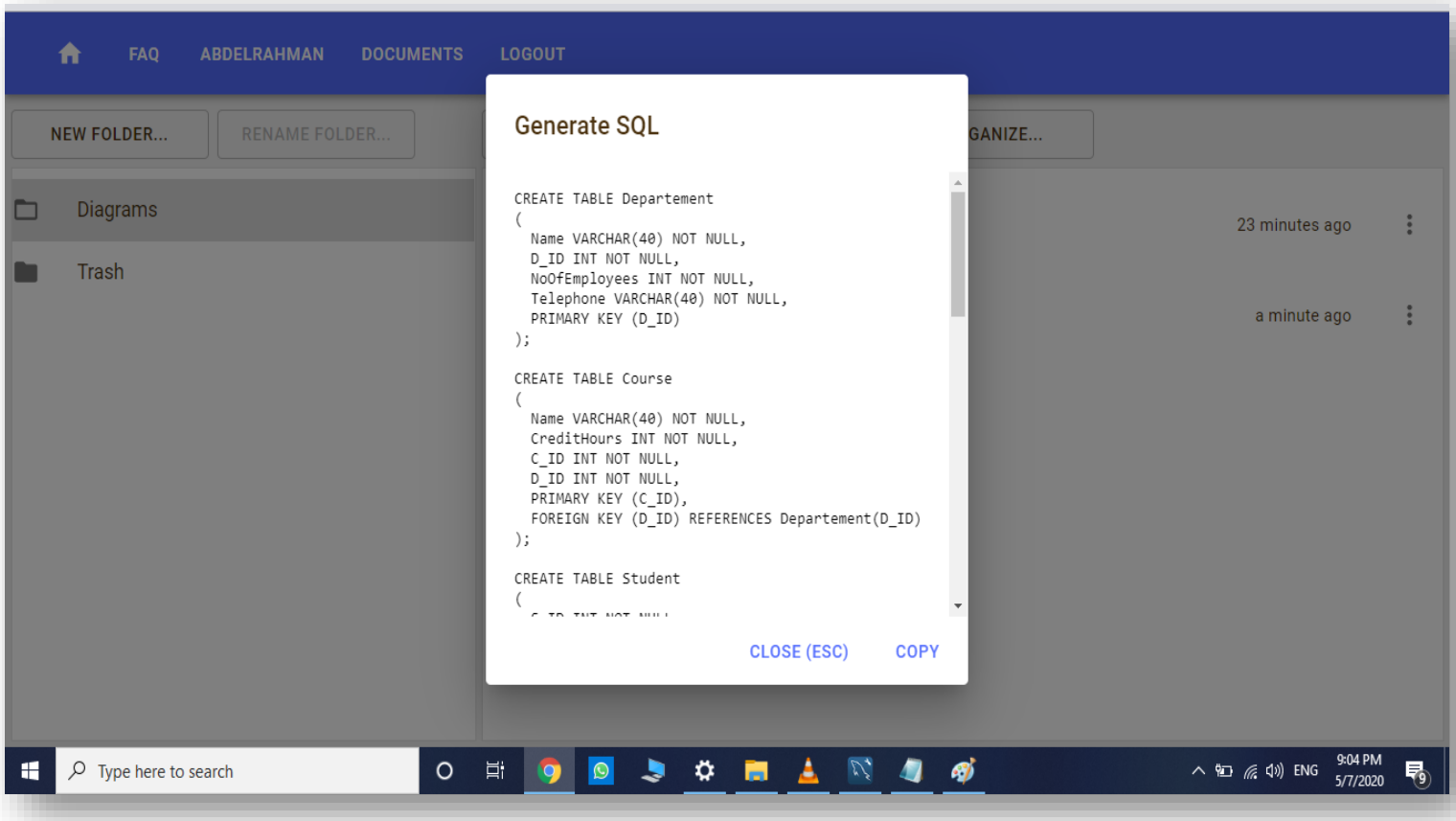
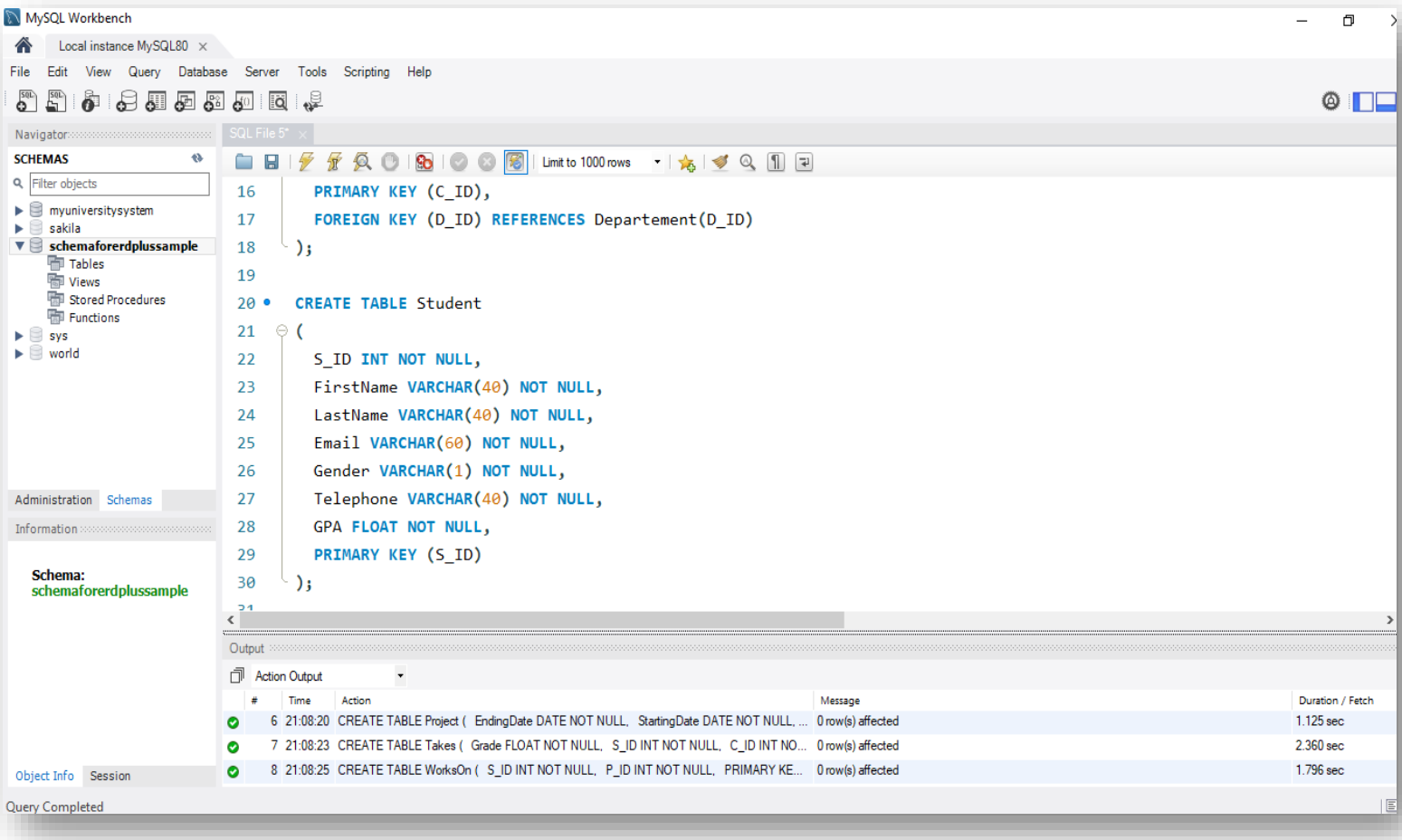



*Figure: SQL generated*

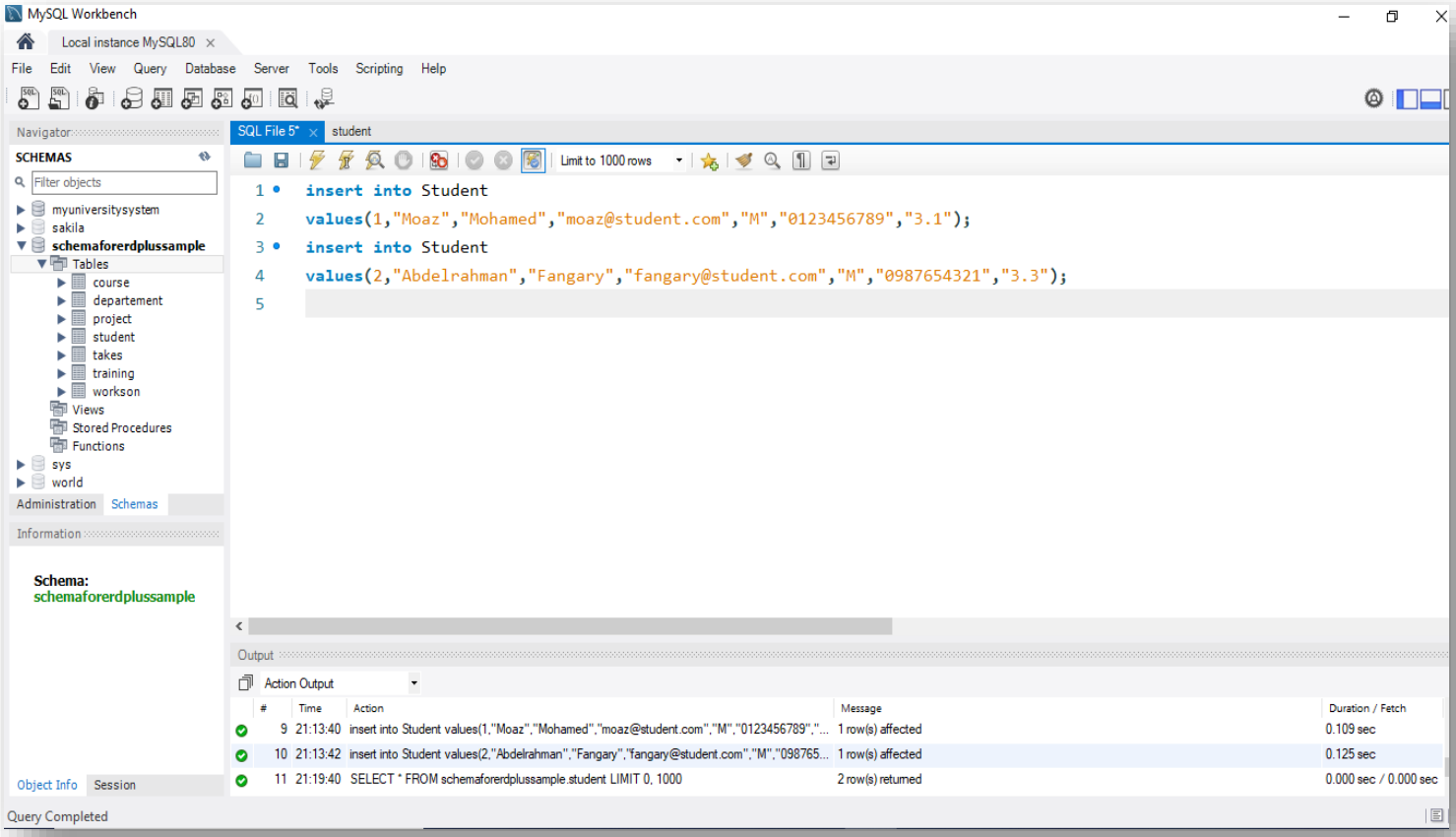
## 7.2 Part2: SQL Tool

We used **MySQL workbench** tool: MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system.

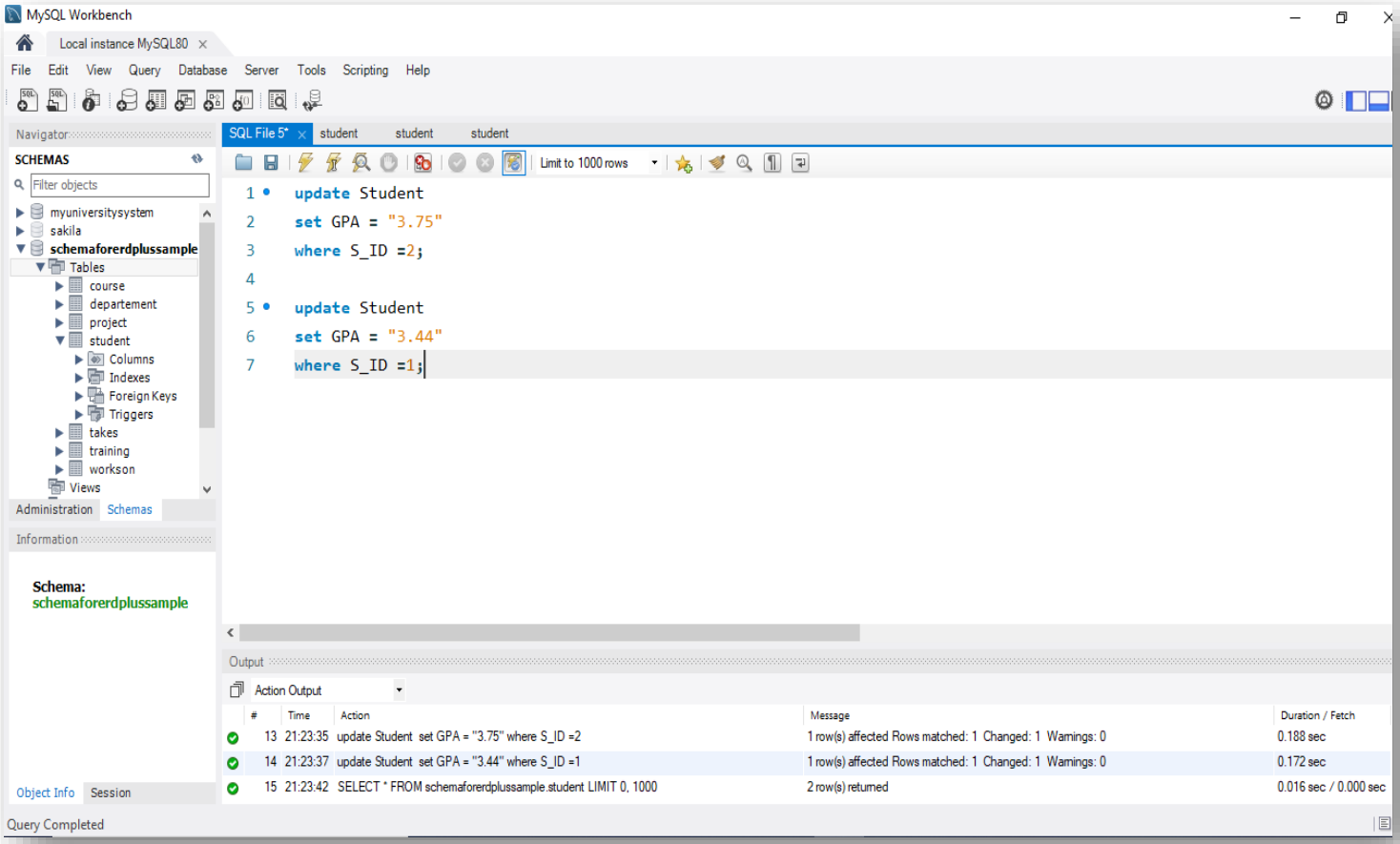
**Creation Sample:**



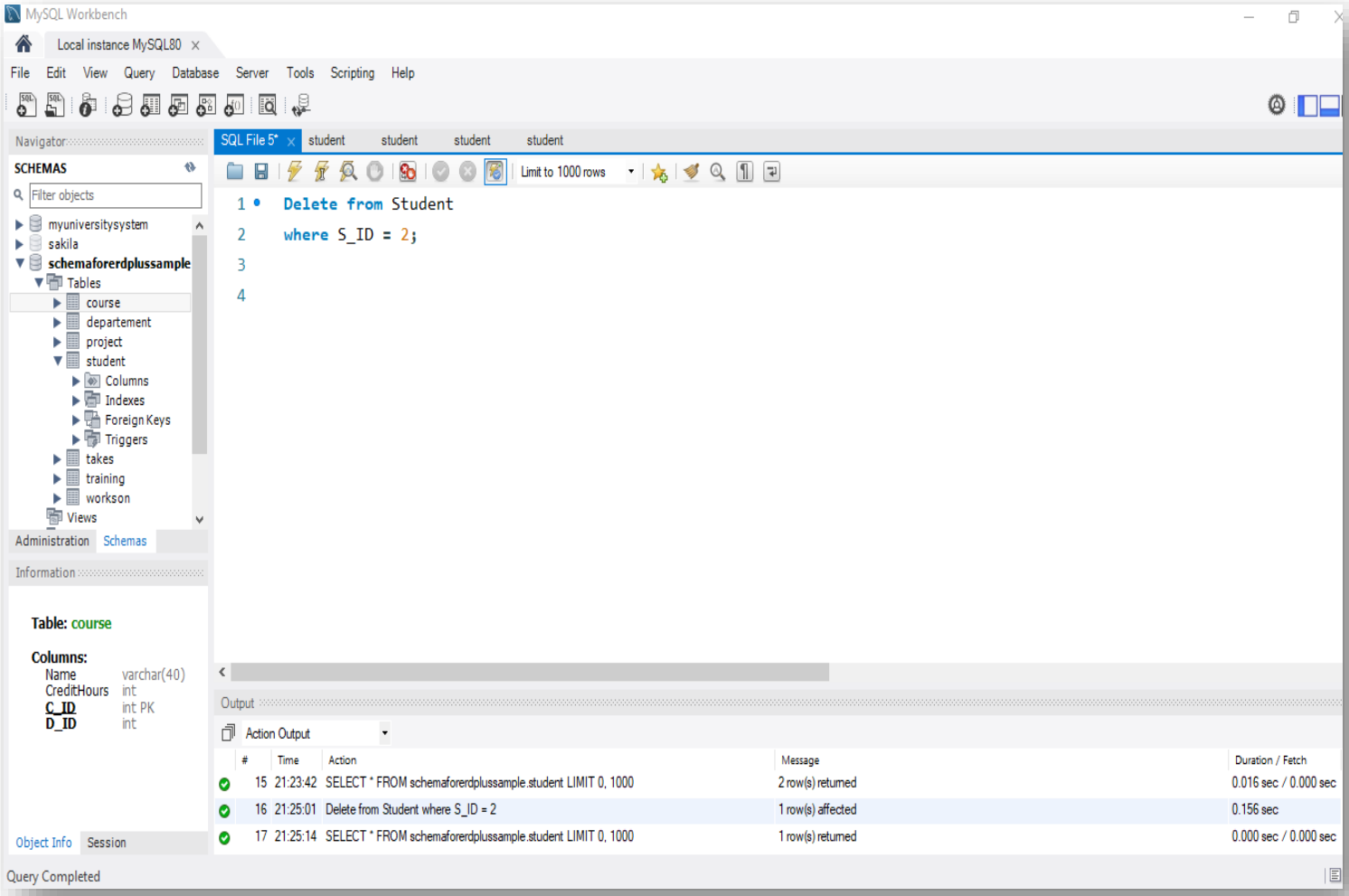
## Insertion Sample:



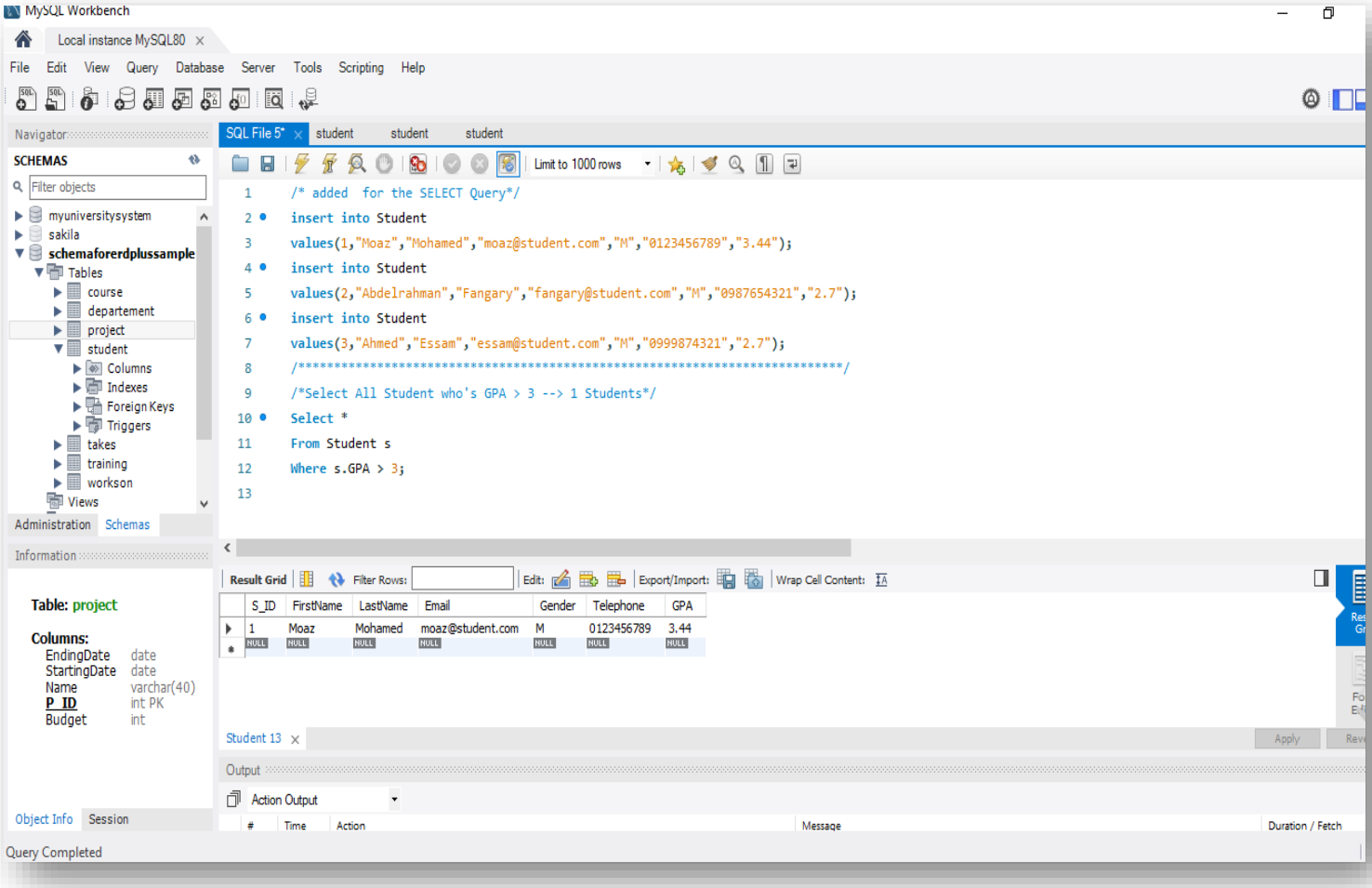
## Update Sample:



## Deletion Sample:



## Selection Sample:



## 8. References


1. Fundamentals of database systems ( 7th edition, 2015) By Ramez Elmasri, Shamkant B. Navathe.
2. MySQL Workbench: https://www.mysql.com/products/workbench/
3. ERDPlus: https://erdplus.com/