**Table Relationship**

**Diagram

Description automatically generated**

**Table Descriptions**

The tables are described in order of relationship described in the previous figure.

STUDENT

There is one row in the STUDENT table for each student registering for the course.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| SID | Varchar (10) | Student ID | No |
| Name | Varchar (45) | Last, First Name | Yes |
| Address | Varchar (45) | Address | Yes |

PROFESSOR

There is one row in the PROFESSOR table for each professor in the department

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| PID | Varchar (10) | Professor ID | No |
| Name | Varchar (45) | Last, First Name | Yes |
| Office | Varchar (10) | Office number | Yes |
| DateofBirth | Date | Age of Professor | Yes |

COURSE

There is one row in the COURSE table for each course offered that quarter.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| CourseNum | Integer | Course number | No |
| DeptName | Varchar(45) | Name of Department | No |
| CourseName | Varchar(45) | Course Name | Yes |
| ClassRoom | Varchar(45) | Room Number | Yes |
| Enrollment | Integer | Number of Students Enrolled | Yes |

DEPARTMENTS

There is one row in the DEPARTMENTS table for each department in the University.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| DeptName | Varchar (45) | Name of Department | No |
| ChairmanID | Varchar (45) | Name of Chairman | Yes |

PREREQ

There is one row in the PREREQ table for each pre-requite for any course.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| CourseNum | Integer | Course Number | No |
| DeptName | Varchar (45) | Name of Department | No |
| PreReqNumber | Integer | Pre requisite Number | Yes |
| PreReqDeptName | Varchar (45) | Pre requisite Dept Name | Yes |

TEACH

There is a row in the TEACH table for each course taught by a professor.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| PID | Varchar(10) | Professor who teaches the course | No |
| CourseNum | Integer | The course taught | No |
| DeptName | Varchar (45) | The department the course taught | No |

TAKE

There is a row in the TAKE table for each course enrolled by a student.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| SID | Varchar (10) | ID of student taking a course | No |
| CourseNum | Integer | The course take by student | No |
| DeptName | Varchar (45) | The department of course | No |
| Grade | Decimal(4,2) | The grade obtained | Yes |
| ProfessorEval | Decimal(4,2) | The professor evaluation | Yes |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**HANDSON**

**Example 1:** Using **INNER JOIN** and **ON**:

**Question:** Write an SQL query that joins the COURSES with the PREREQ relation with the condition that their corresponding *CourseNum* match.

SELECT \* FROM COURSES c

**JOIN** PREREQ p

**ON** c.CourseNum = p.CourseNum;

**Example 2: Using INNER JOIN and ON with additional conditions:**

**Question:** Write an SQL query that joins the COURSES with the PREREQ relation with a condition that their corresponding *CourseNum* match AND the enrollment > 25

SELECT \* FROM COURSES c

**JOIN** PREREQ p

**ON** c.CourseNum = p.CourseNum

**AND** c.enrollment > 25;

**Example 3: Natural Join:**

**Question:** Write an SQL for the NATURAL JOIN of PROFESSORS and TEACH relations

SELECT \* FROM PROFESSORS

**NATURAL JOIN** TEACH;

**Example 4: INNER JOIN and USING:**

**Question:** Write an SQL that matches COURSES and PREREQ relations USING *CourseNum* and *DeptName* attributes to join these relations.

SELECT \* FROM COURSES c

**JOIN** PREREQ p

**USING** (coursenum, deptname);

**Example 5: LEFT OUTER JOIN:**

**Question:** Write an SQL to return all details from the PROFESSORS relation and joins details from TEACH relation whether or not the Professor’s *PID* exists in the TEACH relation.

SELECT \* FROM PROFESSORS p

**LEFT JOIN** TEACH t

**ON** p.pid = t.pid;

**Example 6: AGGREGATION , GROUP BY, HAVING:**

**Question:** SELECT the *deptname* and their minimum and maximum *enrollments* from the COURSES relation and GROUP the result BY *deptname* and   
HAVING min(enrollments) > 20.

SELECT deptname, min(enrollment),max(enrollment)

FROM COURSES

**GROUP BY** deptname

**HAVING** min(enrollment) > 20;