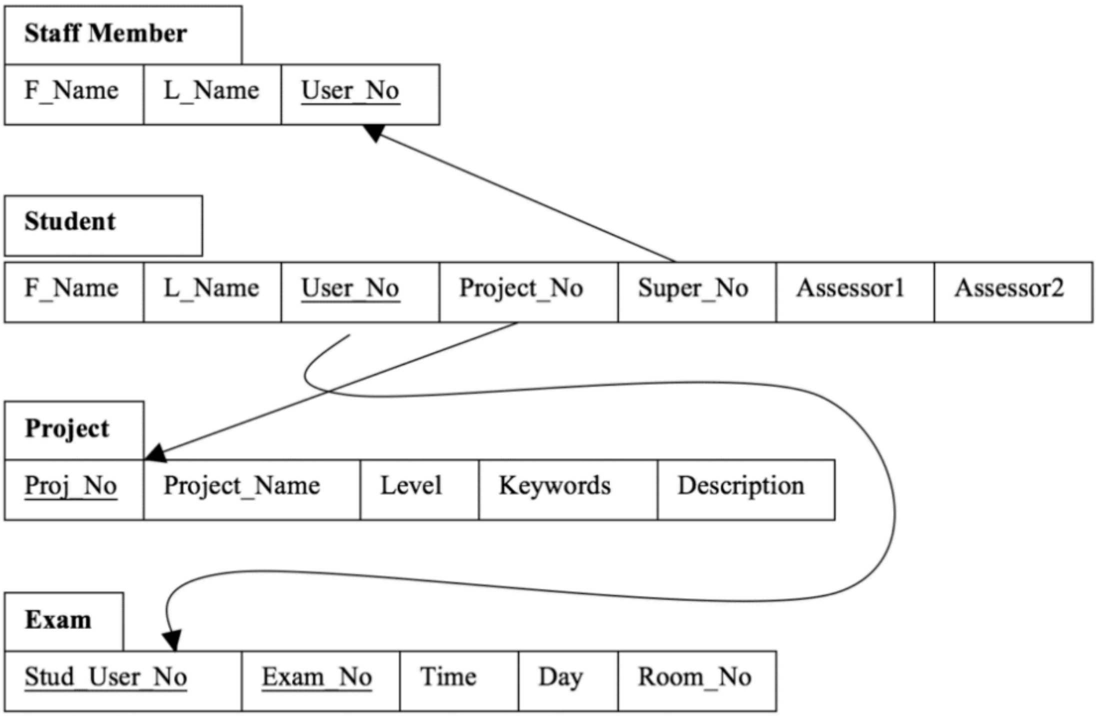
# Question: 02



There is a problem in the above table: the Arrow direction between Student table and Exam table is supposed to be opposite. The Stud_User_No is a Foreign Key in the exam table. I used the fixed Arrow Direction for this Schema (ie, this Solution is correct).

1. **Populate sensible data in the following database. Using Insert statement to insert 3 rows per table.**

Writing SQL Code to CREATE and INSERT values into tables:

```
-- Create the Database
use master
go
create database Exam_Project_System;

-- Use the Database
use [Exam_Project_System];

-- Drop all tables if exist
DROP TABLE Staff_Member;
DROP TABLE Project;
DROP TABLE Student;
DROP TABLE Exam;
```

```sql
-- Select all values in tables
SELECT * FROM Staff_Member;
SELECT * FROM Project;
SELECT * FROM Student;
SELECT * FROM Exam;

-- Create Table Staff_Member
create table Staff_Member
(
    F_Name varchar(100) NOT NULL,
    L_Name varchar(100) NOT NULL,
    User_No int PRIMARY KEY NOT NULL,
);

-- Create table Project
create table Project
(
    Proj_No INT NOT NULL PRIMARY KEY,
    Project_Name VARCHAR(100) NOT NULL,
    Level INT NOT NULL,
    Keywords VARCHAR(100) NOT NULL,
    Description VARCHAR(100) NOT NULL
);

-- Create Table Student
create table Student
(
    F_Name VARCHAR(100) NOT NULL,
    L_Name VARCHAR(100) NOT NULL,
    User_No int PRIMARY KEY NOT NULL,
    Project_No int NOT NULL,
    Super_No int NOT NULL,
    FOREIGN KEY (Super_No) REFERENCES Staff_Member(User_No),
    Assessor1 VARCHAR NOT NULL,
    Assessor2 VARCHAR NOT NULL,
    FOREIGN KEY (Project_No) REFERENCES Project(Proj_No)
);

-- Create Table Exam
create table Exam
(
    Stud_User_No INT NOT NULL,
    Exam_No INT NOT NULL,
    Time Time NOT NULL,
    Day INT NOT NULL,
    Room_No INT NOT NULL,
    -- stud_user_no refrences user no in student
    FOREIGN KEY (Stud_User_No) REFERENCES Student(User_No),
    PRIMARY KEY (Stud_User_No, Exam_No)
);

-- Insert values into Staff_Member table
INSERT INTO Staff_Member (F_Name, L_Name, User_No)
VALUES ('Hasan', 'Yahya', 1001),
       ('Wasee', 'Rehman', 1002),
       ('Zayed', 'Abdullah', 1003);

-- Insert values into Project table
INSERT INTO Project (Proj_No, Project_Name, Level, Keywords, Description)
VALUES (1, 'Weather', 1, 'Weather, App', 'Made using Javascript'),
       (2, 'Pacman', 2, 'Pacman, SFML', 'Made using C++ and SFML Library'),
       (3, 'Sudoku', 3, 'Sudoku, SFML', 'Made using C++ and SFML Library');

-- Alter Table Student
ALTER TABLE Student
ALTER COLUMN Assessor1 VARCHAR(100) NOT NULL;
ALTER TABLE Student
ALTER COLUMN Assessor2 VARCHAR(100) NOT NULL;

-- Insert values into Student table
INSERT INTO Student (F_Name, L_Name, User_No, Project_No, Super_No,
Assessor1, Assessor2)
VALUES ('Ali', 'Sahab', 1010, 1, 1001, 'Assessor1A', 'Assessor2A'),
       ('Michael', 'Johnson', 1011, 2, 1002, 'Assessor1B', 'Assessor2B'),
       ('Mani', 'Hasan', 1012, 2, 1002, 'Assessor1C', 'Assessor2C');

-- Insert values into Exam table
INSERT INTO Exam (Stud_User_No, Exam_No, Time, Day, Room_No)
VALUES (1010, 1, '09:00:00', 1, 005),
       (1011, 1, '10:30:00', 1, 009),
       (1011, 4, '10:30:00', 1, 007);

-- Now all Values have been entered into Tables
```

After entering the Values, we get the tables:

- **Staff Member (Table):**

| F_Name | L_Name | User_No |
|---|---|---|
| Hasan | Yahya | 1001 |
| Wasee | Rehman | 1002 |

|  |  |  |
|---|---|---|
| Zayed | Abdullah | 1003 |

- **Project (Table):**

| Proj_No | Project_Name | Level | Keywords | Description |
|---|---|---|---|---|
| 1 | Weather | 1 | Weather, App | Made using Javascript |
| 2 | Pacman | 2 | Pacman, SFML | Made using C++ and SFML Library |
| 3 | Sudoku | 3 | Sudoku, SFML | Made using C++ and SFML Library |

- **Student (Table):**

| F_Name | L_Name | User_No | Project_No | Super_No | Assessor1 | Assessor2 |
|---|---|---|---|---|---|---|
| Ali | Sahab | 1010 | 1 | 1001 | Assessor1A | Assessor2A |
| Michel | Johnson | 1011 | 2 | 1002 | Assessor1B | Assessor2B |
| Mani | Hasan | 1012 | 3 | 1003 | Assessor1C | Assessor2C |

- **Exam (Table):**

| Stud_User_No | Exam_No | Time | Day | Room_No |
|---|---|---|---|---|
| 1010 | 1 | 09:00:00 | 1 | 5 |
| 1011 | 1 | 10:30:00 | 1 | 9 |
| 1011 | 4 | 10:30:00 | 1 | 7 |

2. **Delete the records of all students who are working on project no 10.**

```sql
-- 2) Delete the records of all students who are working on project no 10.
DELETE FROM Student
WHERE Project_No = 10;
```

Since, the tables have no **Project_No: 10**. No, change will happen in the tables.

3. **Change the level of projects from 4 to 5 which have ‘Android’ in their keywords.**

```
-- 3) Change the level of projects from 4 to 5 which have 'Android' in their
keywords.
UPDATE Project
SET Level = 5
WHERE Keywords LIKE '%Android%' AND Level = 4;
```

Since, there are no **"Android"** keywords in the given tables. This query won't cause any change in the tables.

### 4. Write query for the following problems:

- **Find exam time and date of student user no: 1010 and Room_no = 005:**

```
-- 1. Find exam time and date of student user no: 1010 and Room_no = 005
SELECT Time, Day
FROM Exam
WHERE Stud_User_No = 1010 AND Room_No = 005;
```

This will return the following table:

| Time | Day |
|---|---|
| 09:00:00 | 1 |

- **Show all project names in descending order:**

```
-- 2. Show all project names in descending order.
SELECT Project_Name
FROM Project
ORDER BY Project_Name DESC;
```

This returns the following table:

| Project_Name |
|---|
| Weather |
| Sudoku |
| Pacman |

- **Show F_Name of staff member with L_Name starting with A and ending with A:**

```sql
-- 3. Show fname of staff member with L_Name starting with A and ending with
A.
SELECT F_Name
FROM Staff_Member
WHERE L_Name LIKE 'A%A';
```

This returns the following table (It has a column but no tuples/rows):

| F_Name |
|---|

- **Show name of student who has not been assigned any supervisor as yet:**

```sql
-- 4. Show name of student who has not been assigned any supervisor as yet.
SELECT F_Name, L_Name
FROM Student
WHERE Super_No IS NULL;
```

This returns a table with two columns (but no tuples):

| F_Name | L_Name |
|---|---|

- **Show user no who are sitting between room 005 and 009 during exam:**

```sql
-- 5. Show user no who are sitting between room 005 and 009 during exam.
SELECT Stud_User_No
FROM Exam
WHERE Room_No BETWEEN 005 AND 009;
```

This returns the table:

| Stud_User_No |
|---|
| 1010 |
| 1011 |
| 1011 |

- **Show students name with its supervisor name and project number:**

```sql
-- 6. Show students name with its supervisor name and project number.
SELECT S.F_Name, S.L_Name, SM.F_Name, SM.L_Name, S.Project_No
FROM Student S
JOIN Staff_Member SM
ON S.Super_No = SM.User_No;
```

This returns the table:

| F_Name | L_Name | F_Name | L_Name | Project_No |
|---|---|---|---|---|
| Ali | Sahab | Hasan | Yahya | 1 |
| Michael | Johnson | Wasee | Rehman | 2 |
| Mani | Hasan | Wasee | Rehman | 2 |

**Note:** I submitted the original SQL Files in Google Classroom as well for both Question 01 & 02. You can check the `Exam_Project_System.sql` and `Airline_Reservation_System.sql` for both of them. Moreover, ***Microsoft SQL Server*** was used for the Database (which is reflected in the SQL Syntax). In Question 01, I created extra tables for Attributes (1 to N) and used composition keys because when I visited Course Instructor, **Hina Iqbal** in her office, she allowed me to do so.