MIS: 111903110
MIS: 111903114

Online Course Management

Schemas along with Normalization:

```
use College;

CREATE TABLE IF NOT EXISTS student_login(
   MIS int Primary Key,
   password varchar(255)
);

CREATE TABLE IF NOT EXISTS department(
   deptID varchar(8) primary key,
   deptName varchar(50) not null
);

CREATE TABLE IF NOT EXISTS student_account(
   MIS int Primary Key,
   firstname varchar(20) not null,
   lastname varchar(20) not null,
   email varchar(40) not null,
   address varchar(400) not null,
   gender varchar(6) not null,
   yearEnrolled varchar(4) not null,
   DOB date not null,
   deptID varchar(8) not null,
   profilepic varchar(100) default 'default.png' null,
   foreign key (MIS) references student_login(MIS) on delete cascade,
   foreign key (deptID) references department(deptID));

CREATE TABLE IF NOT EXISTS instructor_login(
   instID int Primary Key,
   password varchar(255)
);
```

```
CREATE TABLE IF NOT EXISTS instructor_account(
  instID int Primary Key,
  firstname varchar(20) not null,
  lastname varchar(20) not null,
  email varchar(40) not null,
  address varchar(400) not null,
  gender varchar(6) not null,
  yearEnrolled varchar(4) not null,
  DOB date not null,
  deptID varchar(8) not null,
  profilepic varchar(100) default 'default.png' null,
  foreign key (instID) references instructor_login(instID) on delete cascade,
  foreign key (deptID) references department(deptID));

CREATE TABLE IF NOT EXISTS course (
        courseId varchar(20) not null,
        courseName varchar(40) not null,
        deptID varchar(8),
        term varchar(10) not null,
        credits int,
        textbook varchar(50),
        refTextbook varchar(50),
        courselink varchar(50),
        maxCap int not null,
        seatsLeft int not null,
        primary key (courseId, term),
        foreign key (deptID) references department(deptID));

CREATE TABLE IF NOT EXISTS taken_courses(
        MIS int,
        courseId varchar(20) not null,
        foreign key(courseId) references course(courseId) on delete cascade,
        foreign key(MIS) references student_account(MIS) on delete cascade);

CREATE TABLE IF NOT EXISTS classroom(
        classID int primary key,
        loc varchar(12) not null,
        capacity int,
        deptID varchar(8),
        foreign key(deptID) references department(deptID));
```

```
CREATE TABLE IF NOT EXISTS prereq(
        prereqId varchar(20) not null primary key,
        courseId varchar(20) not null,
        foreign key(prereqId) references course(courseId) on delete cascade,
        foreign key(courseId) references course(courseId) on delete cascade);

CREATE TABLE IF NOT EXISTS taken_in(
        courseId varchar(20) not null primary key,
        classID int,
        foreign key(classID) references classroom(classID) on delete cascade);

CREATE TABLE IF NOT EXISTS handled_by(
  instID int not null,
  courseId varchar(20) primary key,
  foreign key (instID) references instructor_login(instID) on delete cascade,
  foreign key (courseId) references course(courseId) on delete cascade);
```

FUNCTIONAL DEPENDENCIES:

- ***Logins***:
    - ***Student Login:***
        - Here, we have each student having attributes (MIS, password)
        - Since each (MIS, password) can be uniquely identified by MIS, we can add a primary key constraint to the MIS attribute.
        - MIS → password
    - ***Instructor Login***
        - Here, we have each student having attributes (instID, password)
        - Since each (instID, password) can be uniquely identified by instID, we can add a primary key constraint to the instID attribute.
        - instID → password
- ***Department***:
    - Here we get each department have (deptID, deptName), but we can distinguish between two different values using deptID so the deptID attribute can be made to hold the primary key constraint.
    - deptID → deptName
- ***Accounts***:
    - **Student Account:**
        - Here can have the dependencies as listed down below
        - MIS → firstName
        - MIS → lastName
        - MIS → email
        - MIS → yearEnrolled
        - MIS → dob
        - MIS → deptID
        - Hence we have MIS → firstName, lastName, email, yearEnrolled, dob, deptID
        - So we can have that MIS relates to all the other attributes of the same table, so we can have MIS as the unique identifier and hence primary key.

- **Instructor Account:**
    - Here we have the dependencies as listed down below
    - instID → firstName
    - instID → lastName
    - instID → email
    - instID → yearEnrolled
    - instID → dob
    - instID → deptID
    - Hence we have instID→ firstName, lastName, email, yearEnrolled, dob, deptID
    - So we can have that instID relates to all the other attributes of the same table, so we can have instID as the unique identifier and hence primary key.

- **Course**:
    - Here, we have courseId as a unique identifier / primary key. All other attributes are dependent on coursId.
    - We have following functional dependencies:
        - courseId → courseName
        - courseId → *deptID*
        - courseId → term
        - courseId → credits
        - courseId → textbook
        - courseId → refTextbook
        - courseId → courselink
        - courseId → maxCap
        - courseId → seatsLeft

- **Classroom** :
    - Here, we have classID as a unique identifier / primary key. All other attributes are dependent on coursId.
    - We have following functional dependencies:
        - classID → loc
        - classID → capacity
        - classID → deptID

- **Taken_in**:
    - Here, we have courseId as a unique identifier / primary key. All other attributes are dependent on coursId.
    - We have following functional dependencies:
        - courseId → classID

- **Prereq**:
    - Here, prereqId is primary which uniquely determines tuples in the prereq table.
    - We have following functional dependencies here:
        - prereId → courseId

- **Handled_by**:
    - Here, we have courseId as a unique identifier / primary key. All other attributes are dependent on coursId.
    - We have following functional dependencies:
        - courseId → instID

LINK TO GITHUB:

LINK TO CODE