**CREATING THE MySQL Database tables’ Schema to satisfy the GUI requirements IN MY PERSONAL DATABASE arodriguez103\_db:**

1. **Create User table:** CREATE TABLE User (

username VARCHAR(20) NOT NULL,

password VARCHAR(20) NOT NULL,

name VARCHAR(20) NOT NULL,

PRIMARY KEY(username))Engine=InnoDB;

1. **Create Admin table:** CREATE TABLE Admin (

username VARCHAR(20) NOT NULL,

FOREIGN KEY (username) REFERENCES User(username) ON UPDATE CASCADE ON DELETE CASCADE, PRIMARY KEY (username))Engine=InnoDB;

1. **Create Organization table:** CREATE TABLE Organization (

associationName VARCHAR(30) NOT NULL,

locationCity VARCHAR(20) NOT NULL,

locationState CHAR(2) NOT NULL,

organizationName VARCHAR(40) NOT NULL,

PRIMARY KEY (associationName))Engine=InnoDB;

1. **Create Advocate table:** CREATE TABLE Advocate (

username CHAR(20) NOT NULL,

email VARCHAR(35) NOT NULL,

advocate\_association CHAR(30) NOT NULL,

PRIMARY KEY (username),

FOREIGN KEY (username) REFERENCES User(username) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (advocate\_association) REFERENCES Organization(associationName) ON UPDATE CASCADE ON DELETE CASCADE)Engine=InnoDB;

1. **Create Student table:** CREATE TABLE Student (

username VARCHAR(20) NOT NULL,

classification INT NOT NULL,

email VARCHAR(35) NOT NULL,

student\_association VARCHAR(30) NOT NULL,

ethnicity VARCHAR(20) NOT NULL,

gender CHAR(1) NOT NULL,

FOREIGN KEY (username) REFERENCES User(username) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (student\_association) REFERENCES Organization(associationName) ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY (username))Engine=InnoDB;

//use as only association when referencing organization, it is not associationName

1. **Create Offer table**: CREATE TABLE Offer (

offerNum INT NOT NULL,

username\_S VARCHAR(20) NOT NULL,

offerAssociation VARCHAR(30) NOT NULL,

offerType CHAR(11),

offerStatus CHAR(9) DEFAULT ‘Pending’,

locationCity VARCHAR(20),

locationState CHAR(2),

interviewDate DATE NOT NULL,

interviewType VARCHAR(15) NOT NULL,

FOREIGN KEY (username\_S) REFERENCES Student(username) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (offerAssociation) REFERENCES Organization(associationName) ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY (offerNum, username\_S))Engine=InnoDB;

1. **Create Event table**: CREATE TABLE Event (

eventID INT NOT NULL,

nameEvent VARCHAR(20) NOT NULL,

startDate DATE NOT NULL,

endDate DATE NOT NULL,

locationCity VARCHAR(20) NOT NULL,

locationState CHAR(2) NOT NULL,

eventAssociation VARCHAR(30) NOT NULL,

userCreator VARCHAR(20) NOT NULL,

FOREIGN KEY (userCreator) REFERENCES User(username) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (eventAssociation) REFERENCES Organization(associationName) ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY (eventID))Engine=InnoDB;

1. **Create Session table**: CREATE TABLE Session (

sessionID INT NOT NULL,

eventID INT NOT NULL,

sessionAssociation VARCHAR(30) NOT NULL,

sessionName VARCHAR(60) NOT NULL,

locationSes VARCHAR(30) NOT NULL,

startTime TIME NOT NULL,

endTime TIME NOT NULL,

userCreator VARCHAR(20) NOT NULL,

FOREIGN KEY (userCreator) REFERENCES User(username) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (sessionAssociation) REFERENCES Organization(associationName) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (eventID) REFERENCES Event(eventID) ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY (sessionID, eventID))Engine=InnoDB;

1. **Create SessionCheckedInStudents table:** CREATE TABLE SessionCheckedInStudents (

sessionID INT NOT NULL,

eventID INT NOT NULL,

usernameStu VARCHAR(20) NOT NULL,

FOREIGN KEY (sessionID, eventID) REFERENCES Session(sessionID, eventID) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (usernameStu) REFERENCES Student(username) ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY (sessionID, eventID, usernameStu))Engine=InnoDB;

1. **Create EventCheckedInStudentS table:** CREATE TABLE EventCheckedInStudents (

eventID INT NOT NULL,

usernameStu CHAR(20) NOT NULL,

FOREIGN KEY (eventID) REFERENCES Event(eventID) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (usernameStu) REFERENCES Student(username) ON UPDATE CASCADE ON DELETE CASCADE,

PRIMARY KEY (eventID, usernameStu))Engine=InnoDB;

1. **Create AdminCreateAdvocate table:** CREATE TABLE AdminCreateAdvocate(

adminUsername CHAR(20) NOT NULL,

advocateUsername CHAR(20) NOT NULL,

PRIMARY KEY (adminUsername, advocateUsername),

FOREIGN KEY (adminUsername) REFERENCES Admin(username) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (advocateUsername) REFERENCES User(username) ON UPDATE CASCADE ON DELETE CASCADE)Engine=InnoDB;

**Database Records in MySQL :**

1. **User Table:**

* INSERT INTO User (username, password, name) VALUES (‘admin’, ‘admin’, ‘admin’);
* INSERT INTO User (username, password, name) VALUES (‘user1’, ‘user1, ‘user1’);
* INSERT INTO User VALUES (‘student1’, ‘student1’, ‘Abel’);
* INSERT INTO User VALUES (‘student2’, ‘student2’, ‘Yamel’);
* INSERT INTO User VALUES (‘student3’, ‘student3’, ‘Chip’);
* INSERT INTO User VALUES (‘advocate1’, ‘advocate1’, ‘Ada’);
* INSERT INTO User VALUES (‘advocate2’, ‘advocate2’, ‘Aaron’);
* INSERT INTO User VALUES (‘advocate3’, ‘advocate3’, ‘Liz’);

1. **Admin Table:**

* INSERT INTO Admin (username) VALUES (‘admin’);

1. **Organization Table:**

* INSERT INTO Organization (associationName, locationCity, locationState, organizationName) VALUES (‘Google Cloud Department’, ‘Silicon Valley’, ‘CA’, ‘Google’);
* INSERT INTO Organization VALUES (‘Army Research at UTEP’, ‘El Paso’, ‘TX’, ‘The University of Texas at El Paso’);
* INSERT INTO Organization VALUES (‘Microsoft AI Cortana Team’, ‘Silicon Valley’, ‘CA’, ‘Microsoft’);
* INSERT INTO Organization VALUES (‘Undergrad at UTEP’, ‘El Paso’, ‘TX’, ‘The University of Texas at El Paso’);
* INSERT INTO Organization VALUES (‘Grad at UTEP’, ‘El Paso’, ‘TX’, ‘The University of Texas at El Paso’);
* INSERT INTO Organization VALUES (‘CAHSI’, ‘El Paso’, ‘TX’, ‘Computing Alliance of Hispanic-Serving Institutions’);

1. **Advocate Table:**

* INSERT INTO Advocate (username, email, advocate\_association) VALUES (‘advocate1’, ‘advocate1@gmail.com’, ‘Google Cloud Department’);
* INSERT INTO Advocate VALUES (‘advocate2’, ‘advocate2@gmail.com’, ‘Army Research at UTEP’);
* INSERT INTO Advocate VALUES (‘advocate3’, ‘advocate3@gmail.com’, ‘Microsoft AI Cortana Team’);

1. **Student Table:**

* INSERT INTO Student (username, classification, email, student\_association, ethnicity, gender) VALUES (‘student1’, 4, ‘arodriguez103@miners.utep.edu’, ‘Undergrad at UTEP’, ‘Hispanic’, ‘M’);
* INSERT INTO Student VALUES (‘student2’, 1, ‘yamel@miners.utep.edu’, ‘Undergrad at UTEP’, ‘Non-Hispanic’, ‘F’);
* INSERT INTO Student VALUES (‘student3’, 5, ‘chip@miners.utep.edu’, ‘Grad at UTEP’, ‘Hispanic’, ‘M’);

1. **Offer Table:**

* INSERT INTO Offer (username\_S, offerNum, offerAssociation, offerType, offerStatus, locationCity, locationState, interviewDate, interviewType) VALUES (‘student3’, 1, ‘Google Cloud Department’, ‘Internship’, ‘No Offer’, ‘Silicon Valley’, ‘CA’, ‘2020-02-15’, ‘On-Site’);
* INSERT INTO Offer (username\_S, offerNum, offerAssociation, interviewDate, interviewType) VALUES (‘student3’, 2, ‘Microsoft AI Cortana Team’, ‘2020-04-09’, ‘In-Person’);
* INSERT INTO Offer (username\_S, offerNum, offerAssociation, offerType, offerStatus, locationCity, locationState, interviewDate, interviewType) VALUES (‘student1’, 1, ‘Army Research at UTEP’, ‘Research’, ‘Accepted’, ‘El Paso’, ‘TX’, ‘2020-03-21’, ‘Phone’);

1. **Event Table:**

* INSERT INTO Event (eventID, nameEvent, startDate, endDate, locationCity, locationState, eventAssociation, userCreator) VALUES (12345, ‘HENNAC’, ‘2020-05-25’, ‘2020-05-28’, ‘Austin’, ‘TX’, ‘CAHSI’, ‘admin’);
* INSERT INTO Event VALUES (09876, ‘Military CS Careers’, ‘2020-08-25’, ‘2020-08-28’, ‘Los Angeles’, ‘CA’, ‘Army Research at UTEP’, ‘advocate2’);
* INSERT INTO Event VALUES (2468, ‘Military CS Research Grad Schools’, ‘2020-01-08’, ‘2020-01-10’, ‘Denver’, ‘CO’, ‘Army Research at UTEP’, ‘advocate2’);

1. **Session Table:**

* INSERT INTO Session (sessionID, eventID, sessionAssociation, ‘sessionName’, locationSession, startTime, endTime, userCreator) VALUES (100, 2468, ‘Army Research at UTEP’, ‘Current Military Research Topics and Positions at UTEP’ , ‘Room 100’, ’13:00:00’, ’13:45:00’, ‘advocate 2’);
* INSERT INTO Session VALUES (2525, 09876, ‘Army Research at UTEP’, ‘Current Military Research Topics and Positions at UTEP’, ‘Stage 6’, ’09:15:00’, ’10:30:00’, ‘advocate 2’);
* INSERT INTO Session VALUES (0, 2468, ‘CAHSI’, ‘Introductions On All Attending Grad Schools and Check In Students To Event & Session’, ‘Main Room’, ’08:00:00’, ’09:00:00’, ‘admin’);

1. **SessionCheckedInStudents Table:**

* INSERT INTO SessionCheckedInStudents (sessionID, eventID, usernameStu) VALUES (0, 2468, ‘student1’);
* INSERT INTO SessionCheckedInStudents VALUES (0, 2468, ‘student2’);
* INSERT INTO SessionCheckedInStudents VALUES (2525, 09876, ‘student3’);

1. **EventCheckedInStudents Table:**

* INSERT INTO EventCheckedInStudents (eventID, usernameStu) VALUES (2468, ‘student1’);
* INSERT INTO EventCheckedInStudents VALUES (2468, ‘student2’);
* INSERT INTO EventCheckedInStudents VALUES (09876, ‘student3’);

1. **AdminCreateAdvocate Table:**

* INSERT INTO AdminCreateAdvocate (adminUsername, advocateUsername) VALUES (‘admin’, ‘advocate1’);
* INSERT INTO AdminCreateAdvocate VALUES (‘admin’, ‘advocate2’);
* INSERT INTO AdminCreateAdvocate VALUES (‘admin’, ‘advocate3’);

Assumptions and modifications :

* Assumptions on keeping User as the parent class to the three sub classes that represent the sub type of User is because User encapsulates the required attributes needed for each sub class while allowing each subclass to be able to add, delete, or modify their specialized attributes in the near future. Example is that I added the Email attribute into Advocate’s table schema since Admin needs a way to contact the Advocate(s) that their Advocate User account is created or any other matter that needs to be addressed.
* Admin was kept and not encapsulated within Advocate because of the high possibility of creating more Admin User Accounts and while adding the functionality/representation of being able to query all Advocate accounts to the Admin that approved and created their account. This can be a form security to be able to trace the account creation activity and allow for possible future specialized attributes to be added to Admin table schema that isn’t shared by Advocate.
* Removed ‘# of checked in student’ attribute from Event’s table schema and Session table will also not add ‘#of checked in student’ attribute into its table schema. Just use the count function on EventCheckedInStudents or SessionCheckedInStudents table to get the number of students checked in the event and/or session without having to add an unnecessary column/attribute into the table’s schema.
* Offer can ONLY be created or modified by Student. Student will report their interviews and the Association’s Name that is giving the interview. Student will later be able to update and complete Offer’s remaining columns that were left empty by reporting the info sent to the Student in regard to the Offer. associationName attribute in Offer is a derived attribute from Organization which represents who is interviewing the Student and who will be notifying the student if an Offer will be extended or not.
* No attribute/column in Event or Session for capacity limits. We aren’t concerned about the capacity limits or capping the number of students that can attend a session or event for now. If desired, it can easily be added through a mysql alter table and add column command.
* First name and last name are together in the Name attribute of a table for a person where applicable throughout the tables. Names of any User, the name attribute can contain their either just the first name or both first and last name.
* Added Email attribute into the Advocate’s table schema. I assumed we would need advocate’s email in order the advocate to receive a confirmation email from the admin when the admin approves and creates the advocate user account
* Added sessionName into the table Session. I assumed session needed a name so that CASHI Would give clarity on what the session did for students like Info, interviewing, or any other activity instead of just going by ID’s.
* All Location attributes except the Location attribute in Session were made into two composite attributes or two separate attributes where one will contain the city name and the other will contain a two CHAR representation of the state name. This allows the location’s state name to be represented and stored at a more efficient and precise method rather than bundling city and state name into one string value attribute within a table.
* Session’s Location is the location of the Session within the hosted building or area of the Event and will not be represented as holding the values of city and state names.
* Removed Accept/Decline attribute from Offer and added more possible value options that the Offer Status attribute can store/contain. Offer Status’s value can be either Pending, No Offer, Accept, or Decline. Pending will be the default value, unless otherwise inputted into the Offer table, until an update on no offer was extended or an offer was extended and the student decided Accept or Decline. If no offer was extended, then the value updates to No Offer. Otherwise if an offer was extended, then the student’s decision will update the value to Accept or Decline.
* Interview type only has 4 options which are Phone, Online, On-Site, and In-Person. On-site means the interview was done right at the Session and/or Event the student checked into. In-Person means the interview happened in person on a later date and was done outside from the Event the student checked into.
* Added userCreator attribute to both Session and Event table schema since an Event or Session is unable to exist or be created in the DB unless it was created by an Admin or Advocate which will also be most likely the one hosting the Event or Session for either their Association or Organization. Having userCreator as an attribute in both table schema also serves as what is required for an Event or Session to take place and who is responsible.