CPSC 304 Project Cover Page

Milestone #: 2

Date: 2/26/2023

Group Number: 51

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Kate Saychaleun	40259574	v1m0l	khammyschl@gmail.com
Aditya Poluri	68466739	c4y2b	adipoluri@gmail.com
Jonathan Xu	28156065	m1u1r	jonnyux@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Milestone 2

2. Project Description

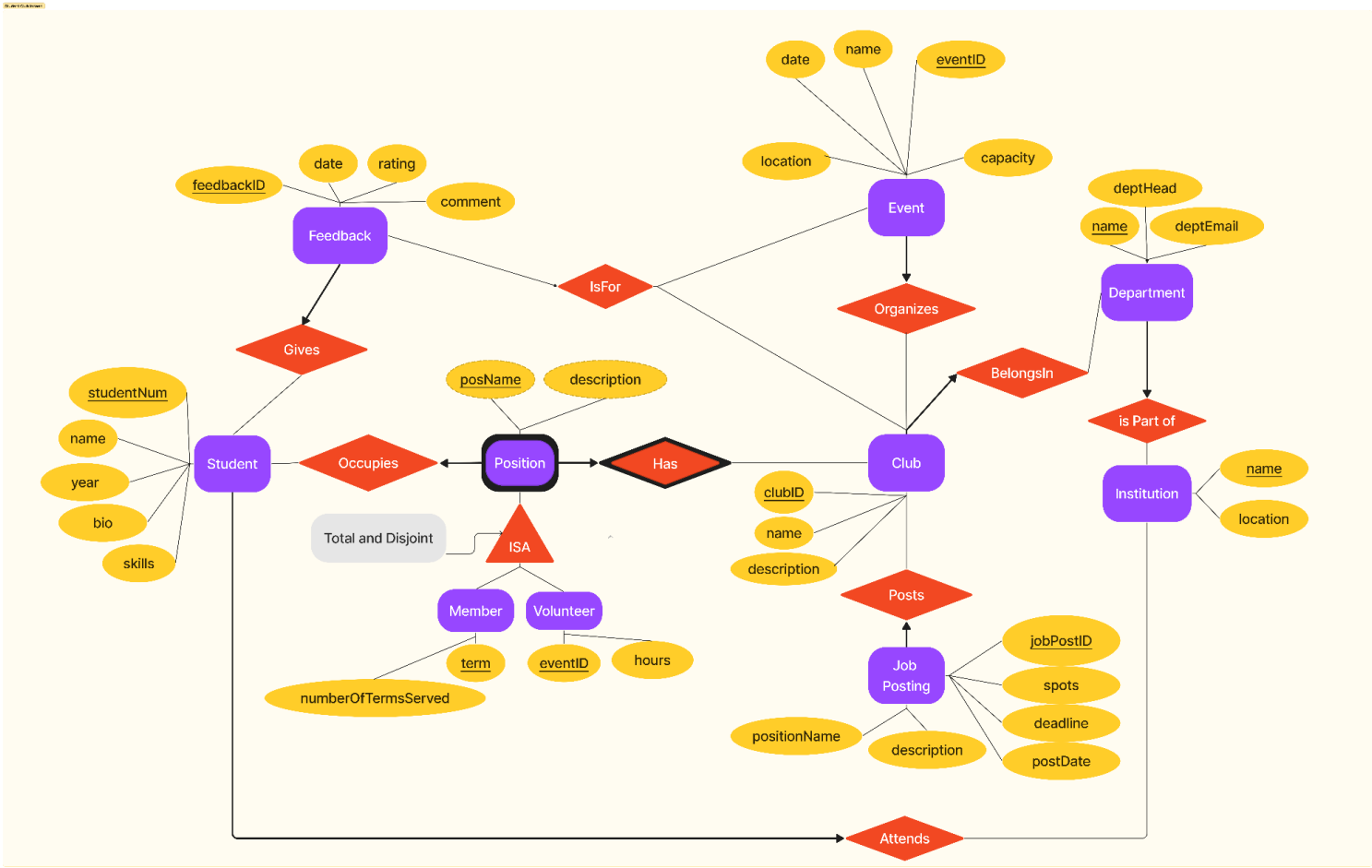
What is the domain of the application?
The domain of the project is a job/volunteer board that centers around advertising campus organizations and their opportunites while allowing students to apply for student club/organization positions that are posted by other students. Event logistics and volunteering opportunities are also included in the domain. We aim to centralize the search for campus opportunities.

What aspects of the domain are modeled by the database?
The database is a conceptual model designed to capture the interaction between student entities and clubs, positions, and events at UBC. The database will be modeled based on the job postings authorized by AMS clubs.

ER diagram changes:

We weren't given that much feedback to our ER diagram, other than adding some more attributes to our ISA so we added some additional fields to our ISA. We also clarified what our ISA is, writing that its Disjoint and Total. We also renamed a few attributes.

3. ER Diagram On Next Page



4. The schema derived from your ER diagram (above).

- Department(deptName: char(30), **institutionName: char(50),** deptEmail: char(20), deptHead: char(20)); institutionName not null
- Institution(name: char(30), location: char(50));
- Event(eventID: int, **clubID: int**, capacity: int, name: char(30), location: char(20), date: date); clubID not null
- Club(clubID: int, **deptName: char(30)**, name: char(30), description: char(250)); deptName is not null
- JobPosting(jobPostID: int, **clubID: int**, description: char(250), positionName: char(50), spots: int; deadline: date, postDate: date); clubID not null
- EventFeedback(feedbackID: int, **studentNum: int**, **eventID: int**, date: date, rating: int, comment: char(250)); studentNum and eventID is not null
- ClubFeedback(feedbackID: int, **studentNum: int**, **clubID: int**, date: date, rating: int, comment: char(250)); studentNum and clubID is not null
- VolunteerPosition(posName: char(50), eventID: int, **clubID: int, studentNum: int,** description: char(250), hours: int); clubID and studentNum not null
- MemberPosition(posName: char(50), term: char(7), **clubID: int, studentNum: int,** description: char(250), numberOfTermsServed: int); clubID and studentNum not null
- Student(studentNum: int, **institutionName: char(50)**, name: char(30), year: int, bio: char(250), skills: char(30)); institutionName not null

5. Functional Dependencies (FDs)

Department(deptName: char(30), **institutionName: char(50),** deptEmail: char(20), deptHead: char(20));

- deptName, institutionName → deptName, institutionName, deptEmail, deptHead, location
- deptEmail → deptHead
- deptHead → deptEmail

Institution(name: char(30), location: char(50));

- name → location

Event(eventID: int, **clubID: int**, capacity: int, eventName: char(30), location: char(20), date: date);

- eventID → eventID, clubID, capacity, name, location, date
- Location, date → Location, date, clubID, eventID, eventName, capacity

Club(clubID: int, **deptName: char(30)**, name: char(30), description: char(250));

- clubID → clubID, deptName, clubName, description

JobPosting(jobPostID: int, **clubID: int**, description: char(250), positionName: char(50), spots: int; deadline: date, postDate: date);

- jobPostID → jobPostID, clubID, description, positionName, spots, deadline, postDate

EventFeedback(feedbackID: int, **studentNum: int**, **eventID: int**, date: date, rating: int, comment: char(250));

- feedbackID → feedbackID, studentNum, eventID, date, rating, comment
- eventID, studentNum -> feedbackID

ClubFeedback(feedbackID: int, **studentNum: int**, **clubID: int**, date: date, rating: int, comment: char(250));

- feedbackID → feedbackID, studentNum, clubID, date, rating, comment
- clubID, studentNum -> feedbackID

VolunteerPosition(posName: char(50), eventID: int, **clubID: int, studentNum: int,** description: char(250), hours: int);

- posName, eventID, clubID, studenNum → posName, eventID, clubID, studentNum, description, hours

MemberPosition(posName: char(50), term: char(7), **clubID: int, studentNum: int,** description: char(250), numberOfTermsServed: int);

- posName, term, clubID, studenNum → posName, term, clubID, studentNum, description, numberOfTermsServed

Student(studentNum: int, **institutionName: char(50)**, name: char(30), year: int, bio: char(250), skills: char(30))

- studentNum, institionName → studentNum, institionName, name, year, bio, skills

6. Normalization **(HIGHLIGHTED TABLES ARE THE NORMALIZED SCHEMAS)**

Department(deptName: char(30), **institutionName: char(50),** deptEmail: char(20), deptHead: char(20));

- deptName, institutionName → deptName, institutionName, deptEmail, deptHead, location
- deptEmail → deptHead

- deptHead → deptEmail

  First Decomposition:

  deptEmail+ = {deptHead, deptEmail} **NOT** in BCNF,

  ⇒ split into Dept(deptHead, deptEmail) and Dept(deptName, instName, deptHead)

  Result: both **in BCNF**

  New Tables:

- DepartmentHead(deptHead: char(20)**,** deptEmail: char(20));
- Department(deptName: char(30), **institutionName:** char(50), **deptHead**: char(20));

Institution(name: char(30), location: char(50));

- name → location

  Name+ = {name, location} **IN BCNF**

Event(eventID: int, **clubID: int**, capacity: int, eventName: char(30), location: char(20), date: date);

- eventID → eventID, clubID, capacity, name, location, date
- Location → capacity
- Location, date → eventID

  eventID+ = {everything since PK}

  Location,date+ = {eventID, + everything since gets PK}

  Location+ = {capacity} **NOT** in BCNF, decompose

  Decomposition:

  Split into Event(location, capacity) and Event(eventID, clubID, eventName, location, date)

  eventID+ = {everything since PK}

  Location,date+ = {eventID, + everything since gets PK}

  Location+ = {capacity} **IN BCNF**

  Resulting Tables:

- Event(eventID: int, **clubID: int**, eventName: char(30), **location**: char(20), date: date);
- LocationCapacity(location: char(20), capacity: int);

Club(clubID: int, **deptName: char(30)**, name: char(30), description: char(250));

- clubID → clubID, deptName, clubName, description

  clubID+ = {clubID, deptName, clubName, description} **IN BCNF**

JobPosting(jobPostID: int, **clubID: int**, description: char(250), positionName: char(50), spots: int; deadline: date, postDate: date);

- jobPostID → jobPostID, clubID, description, positionName, spots, deadline, postDate

  jobPostID+ = {jobPostID, clubID, description, positionName, spots, deadline, postDate} **IN BCNF**

EventFeedback(feedbackID: int, **studentNum: int**, **eventID: int**, date: date, rating: int, comment: char(250));

- feedbackID → feedbackID, studentNum, eventID, date, rating, comment
- eventID, studentNum -> feedbackID

  eventID, studentNum+ = {feedbackID + everything since PK is obtained}

  IN BCNF

ClubFeedback(feedbackID: int, **studentNum: int**, **clubID: int**, date: date, rating: int, comment: char(250));

- feedbackID → feedbackID, studentNum, clubID, date, rating, comment
- clubID, studentNum -> feedbackID

  eventID, studentNum+ = {feedbackID + everything since PK is obtained}

  IN BCNF

VolunteerPosition(posName: char(50), eventID: int, **clubID: int, studentNum: int,** description: char(250), hours: int);

- posName, eventID, clubID, studenNum → posName, eventID, clubID, studentNum, description, hours

  posName, eventID, clubID, studenNum+ = {posName, eventID, clubID, studentNum, description, hours}**IN BCNF**

MemberPosition(posName: char(50), term: char(7), **clubID: int, studentNum: int,** description: char(250), numberOfTermsServed: int);

- posName, term, clubID, studenNum → posName, term, clubID, studentNum, description, numberOfTermsServed

  posName, term, clubID, studenNum+ = { posName, term, clubID, studentNum, description, numberOfTermsServed} **IN BCNF**

Student(studentNum: int, **institutionName: char(50)**, name: char(30), year: int, bio: char(250), skills: char(30))

- studentNum, institionName → studentNum, institionName, name, year, bio, skills

studentNum, institionName+ = {studentNum, institionName, name, year, bio, skills} **IN BCNF**

Resulting Normalized Tables:

DepartmentHead(deptHead: char(20)**,** deptEmail: char(20));
Department(deptName: char(30), **institutionName:** char(50), **deptHead**: char(20));
Institution(name: char(30), location: char(50));
Event(eventID: int, **clubID: int**, eventName: char(30), **location**: char(20), date: date);
LocationCapacity(location: char(20), capacity: int);
Club(clubID: int, **deptName: char(30)**, name: char(30), description: char(250));
JobPosting(jobPostID: int, **clubID: int**, description: char(250), positionName: char(50), spots: int; deadline: date, postDate: date);
EventFeedback(feedbackID: int, **studentNum: int**, **eventID: int**, date: date, rating: int, comment: char(250));
ClubFeedback(feedbackID: int, **studentNum: int**, **clubID: int**, date: date, rating: int, comment: char(250));
VolunteerPosition(posName: char(50), eventID: int, **clubID: int, studentNum: int,** description: char(250), hours: int);
MemberPosition(posName: char(50), term: char(7), **clubID: int, studentNum: int,** description: char(250), numberOfTermsServed: int);
Student(studentNum: int, **institutionName: char(50)**, name: char(30), year: int, bio: char(250), skills: char(30))

7. The SQL DDL statements required to create all the tables from item #6. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

DepartmentHead(deptHead: char(20), deptEmail: char(20));

```
CREATE TABLE DEPARTMENT_HEAD(
        deptHead char(20),
        deptEmail char(20),
        PRIMARY KEY (deptHead)
);
```

Department(deptName: char(30), **institutionName:** char(50), **deptHead**: char(20));

```
CREATE TABLE DEPARTMENT(
        deptName char(30),
        institutionName char(50) NOT NULL,
        deptHead char(20),
        PRIMARY KEY (deptName),
        FOREIGN KEY (institutionName) REFERENCES INSTITUTION(name) ON DELETE
        SET NULL, ON UPDATE CASCADE,
        FOREIGN KEY (deptHead) REFERENCES DEPARTMENT_HEAD(deptHead)
);
```

Institution(name: char(30), location: char(50));

```
CREATE TABLE INSTITUTION(
        name char(30),
        location char(50) NOT NULL,
        PRIMARY KEY (name)
);
```

Event(eventID: int, **clubID: int**, eventName: char(30), **location**: char(20), date: date);

```
CREATE TABLE EVENT(
        eventID int,
        clubID int,
        eventName char(30) NOT NULL,
```

```
        location char(20),
        date date,
        PRIMARY KEY (eventID),
        FOREIGN KEY (clubID) REFERENCES CLUB(clubID),
        FOREIGN KEY (location) REFERENCES LOCATION_CAPACITY(location)
);
```

LocationCapacity(location: char(20), capacity: int);

```
CREATE TABLE LOCATION_CAPACITY(
        location char(20),
        capacity int NOT NULL,
        PRIMARY KEY (location)
);
```

Club(clubID: int, **deptName: char(30)**, name: char(30), description: char(250));

```
CREATE TABLE CLUB(
        clubID int,
        deptName char(30),
        name char(30) NOT NULL,
        description char(250),
        PRIMARY KEY (clubID),
        FOREIGN KEY (deptName) REFERENCES DEPARTMENT(deptName)
);
```

JobPosting(jobPostID: int, **clubID: int**, description: char(250), positionName: char(50), spots: int; deadline: date, postDate: date);

```
CREATE TABLE JOB_POSTING(
        jobPostID int,
        clubID int,
        description char(250) NOT NULL,
        positionName char(50) NOT NULL,
```

```
    spots int NOT NULL,
    deadline date NOT NULL,
    postDate date NOT NULL,
    PRIMARY KEY (jobPostID),
    FOREIGN KEY (clubID) REFERENCES CLUB(clubID) ON DELETE SET NULL, ON
    UPDATE CASCADE
);
```

EventFeedback(<u>feedbackID</u>: int, **studentNum: int**, **eventID: int**, date: date, rating: int, comment: char(250));

```
CREATE TABLE EVENT_FEEDBACK(
    feedbackID int,
    studentNum int,
    eventID int,
    date date NOT NULL,
    rating int,
    comment char(250),
    PRIMARY KEY (feedbackID),
    FOREIGN KEY (studentNum) REFERENCES STUDENT(studentID) ON DELETE
    SET NULL, ON UPDATE CASCADE
    FOREIGN KEY (eventID) REFERENCES EVENT(eventID) ON DELETE SET NULL,
    ON UPDATE CASCADE
);
```

ClubFeedback(<u>feedbackID</u>: int, **studentNum: int**, **clubID: int**, date: date, rating: int, comment: char(250));

```
CREATE TABLE CLUB_FEEDBACK(
    feedbackID int,
    studentNum int,
    clubID int,
    date date NOT NULL,
    rating int,
```

comment char(250),

PRIMARY KEY (feedbackID),

FOREIGN KEY (studentNum) REFERENCES STUDENT(studentID) ON DELETE SET NULL, ON UPDATE CASCADE,

FOREIGN KEY (clubID) REFERENCES CLUB(clubID) ON DELETE SET NULL, ON UPDATE CASCADE,

);

VolunteerPosition(posName: char(50), eventID: int, **clubID: int, studentNum: int,** description: char(250), hours: int);

CREATE TABLE VOLUNTEER_POSITION(

posName char(50),

eventID int,

clubID int,

studentNum int,

description char(250) NOT NULL,

hours int NOT NULL,

PRIMARY KEY(posName, eventID, clubID),

FOREIGN KEY (clubID) REFERENCES CLUB(clubID) ON DELETE SET NULL, ON UPDATE CASCADE

FOREIGN KEY (studentNum) REFERENCES STUDENT(studentNum) ON DELETE SET NULL, ON UPDATE CASCADE

);

MemberPosition(posName: char(50), term: char(7), **clubID: int, studentNum: int,** description: char(250), numberOfTermsServed: int);

CREATE TABLE MEMBER_POSITION(

posName char(50),

term char(7),

clubID int,

studentNum int,

description char(250),

```
        numberOfTermsServed int NOT NULL,
        PRIMARY KEY (posName, term)
        FOREIGN KEY (clubID) REFERENCES CLUB(clubID),
        FOREIGN KEY (studentNum) REFERENCES STUDENT(studentNum)
);
```

Student(studentNum: int, **institutionName: char(50)**, name: char(30), year: int, bio: char(250), skills: char(30))

```
CREATE TABLE STUDENT(
        studentNum int,
        institutionName char(50),
        name char(30) NOT NULL,
        year int NOT NULL,
        bio char(250),
        skills char(30),
        PRIMARY KEY (studentNum),
        FOREIGN KEY (institutionName) REFERENCES INSTITUTION(institutionName)
);
```

8. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later on.

DepartmentHead(deptHead: char(20)**,** deptEmail: char(20));

```
INSERT INTO DEPARTMENT_HEAD(deptHead, deptEmail)
VALUES
        ('Santa Calu', 'Santa@ubc.com'),
        ('Ono Ying', 'Ono@ubc.com'),
        ('Bob Bun', Bob@ubc.com'),
        ('Leo', 'Leo@ubc.com'),
        ('Donald Xi', 'Donald@ubc.com');
```

Department(deptName: char(30), **institutionName:** char(50), **deptHead**: char(20));

INSERT INTO DEPARTMENT(deptName, institutionName, deptHead)
VALUES
('Sciences', 'UBC', 'Santa Calu'),
('Arts', 'UBC', 'Ono Ying'),
('Literature', 'UBC', 'Leo'),
('History', 'UBC', 'Bob Bun'),
('Business', 'UBC', 'Donald Xi');

Institution(name: char(30), location: char(50));

INSERT INTO INSTITUTION(name, location)
VALUES
('UBCV', 'Vancouver'),
('UBCO', 'Okanagan'),
('UofT', 'Toronto'),
('University of Waterloo', 'Waterloo'),
('SFU', 'Burnaby');

Event(eventID: int, **clubID: int**, eventName: char(30), **location**: char(20), date: date);

INSERT INTO EVENT(eventID, clubID, eventName, location, date)
VALUES
(1, 1, 'IntroNight', 'ICICS', '1/1/2022'),
(2, 2, 'Improv Night', 'Nest', '2/1/2023'),
(3, 1, 'Quantum Computing Showcase', 'Hugh Dempster Pavillion', '12/5/2022'),
(4, 454, 'Job Fair', 'Allard Hall', '5/5/2022'),
(5, 3, 'Slam Poetry Competition', 'Buchanan', '8/12/2023');

LocationCapacity(location: char(20), capacity: int);

INSERT INTO LOCATION_CAPACITY(location, capacity)

VALUES

('ICICS', 60),

('Allard Hall', 250),

('Hugh Dempster Pavillion', 35),

('Buchanan', 150),

('Nest', 400);

Club(clubID: int, **deptName: char(30)**, name: char(30), description: char(250));

INSERT INTO CLUB(clubID, deptName, name, description)

VALUES

(1, 'Sciences', 'Quantum Computing Club'),

(2, 'Arts'', 'Improv Club'),

(3, 'Literature', 'Slam Poetry Club'),

(31, 'Sciences', 'Soccer Robotics Team'),

(454, 'Business', Startup Club');

JobPosting(jobPostID: int, **clubID: int**, description: char(250), positionName: char(50), spots: int; deadline: date, postDate: date);

INSERT INTO JOB_POSTING(jobPostingID, clubID, description, positionName, spots, deadline, postDate)

VALUES

(1, 2, 'manage our funds!', 'VP Finance', 1, 1/3/2023, 1/1/2023),

(2, 2, ''manage our events!', 'VP Events', 1, 1/5/2023, 1/2/2023),

(3, 1, 'manage our logistics!', 'VP Logistics', 1, 11/3/2023, 1/1/2023),

(4, 3, 'manage our sponsors!', 'VP Outreach', 1, 1/3/2023, 1/1/2023),

(5,4 54, 'manage our funds!', 'VP Finance', 1, 1/3/2023, 1/1/2023);

EventFeedback(feedbackID: int, **studentNum: int**, **eventID: int**, date: date, rating: int, comment: char(250));

INSERT INTO EVENT_FEEDBACK(feedbackID, studentNum, eventID, date, rating, comment)

VALUES

(1, 111, 22, 3/2/2023, 10, ‘I love this event!’),

(2, 112, 22, 3/2/2023, 1, ‘I hate this event!’),

(3, 113, 22, 3/2/2023, 9, ‘I actually sorta like this event!’),

(4, 114, 22, 3/2/2023, 10, ‘Wait… I love this event!’),

(5, 115, 22, 3/2/2023, 5, ‘This event has weird people’);

ClubFeedback(feedbackID: int, **studentNum: int**, **clubID: int**, date: date, rating: int, comment: char(250));

INSERT INTO CLUB_FEEDBACK(feedbackID, studentNum, clubID, date, rating, comment)

VALUES

(1, 213, 22, 3/2/2023, 10, ‘I love this club!’),

(2, 222, 22, 3/2/2023, 1, ‘I hate this club!’),

(3, 213, 22, 3/2/2023, 9, ‘I actually sorta like this club!’),

(4, 222, 22, 3/2/2023, 10, ‘Wait… I love this club!’),

(5, 55, 22, 3/2/2023, 5, ‘This club has weird people’);

VolunteerPosition(posName: char(50), eventID: int, **clubID: int, studentNum: int,** description: char(250), hours: int);

INSERT INTO VOLUNTEER_POSITION(posName, eventID, clubID, studentNum, description, hours)

VALUES

(‘Desk Volunteer’, 1, 22, 213, ‘work to man the desk’, 5),

(‘Floor Volunteer’, 1, 22, 213, ‘work to man the venue’, 5),

(‘Floor Volunteer’, 1, 22, 213, ‘work to man the venue’, 5),

(‘General Help Volunteer’, 1, 21, 212, ‘help work at food pantry’, 8),

(‘General Help Volunteer’, 1, 21, 212, ‘help work at food pantry’, 5);

MemberPosition(posName: char(50), term: char(7), **clubID: int, studentNum: int,** description: char(250), numberOfTermsServed: int);

INSERT INTO MEMBER_POSITION(posName, term, clubID, studentNum, description, numberOfTermsServed)

VALUES

('VP Finance', 2023W1, 22, 213, 'finance manager', 2),

('VP Events', 2023W1, 21, 213, 'events manager', 1),

('VP Logistics', 2023W1, 22, 212, 'logistics manager', 2),

('VP Marketing', 2023W1, 22, 215, 'marketing manager', 3),

('VP Outreach', 2023W1, 22, 263, 'outreach manager', 1);

Student(studentNum: int, **institutionName: char(50)**, name: char(30), year: int, bio: char(250), skills: char(30))

INSERT INTO STUDENT(studentNum, institutionName, name, year, bio, skills)

VALUES

('213', 'UBC', 'Bobby Han', 2, 'hi my name is bobby', 'java'),

('212', 'UBC', 'Bobo Jones', 2, 'hi my name is bobo', 'sql'),

('214', 'UBC', 'Boy Tong', 2, 'hi my name is boy', 'javascript'),

('215', 'UBC', 'Bobby Bai;, 2, 'hi my name is bobby boy', 'c++'),

('263', 'UBC', 'Bob Wei', 2, 'hi my name is bob', 'java');