University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ____2____

Date: ___2024-10-15___

Group Number: ____70____

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
April Cao	72028764	g4q6w	aprilcao2002@gmail.com
Sherry He	94345741	r8k8g	sherryhe1107@gmail.com
Xinya Lu	88957790	e1e1a	xinyalu13@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

---

2. A brief (~2-3 sentences) summary of your project. Many of your TAs are managing multiple projects so this will help them remember details about your project.

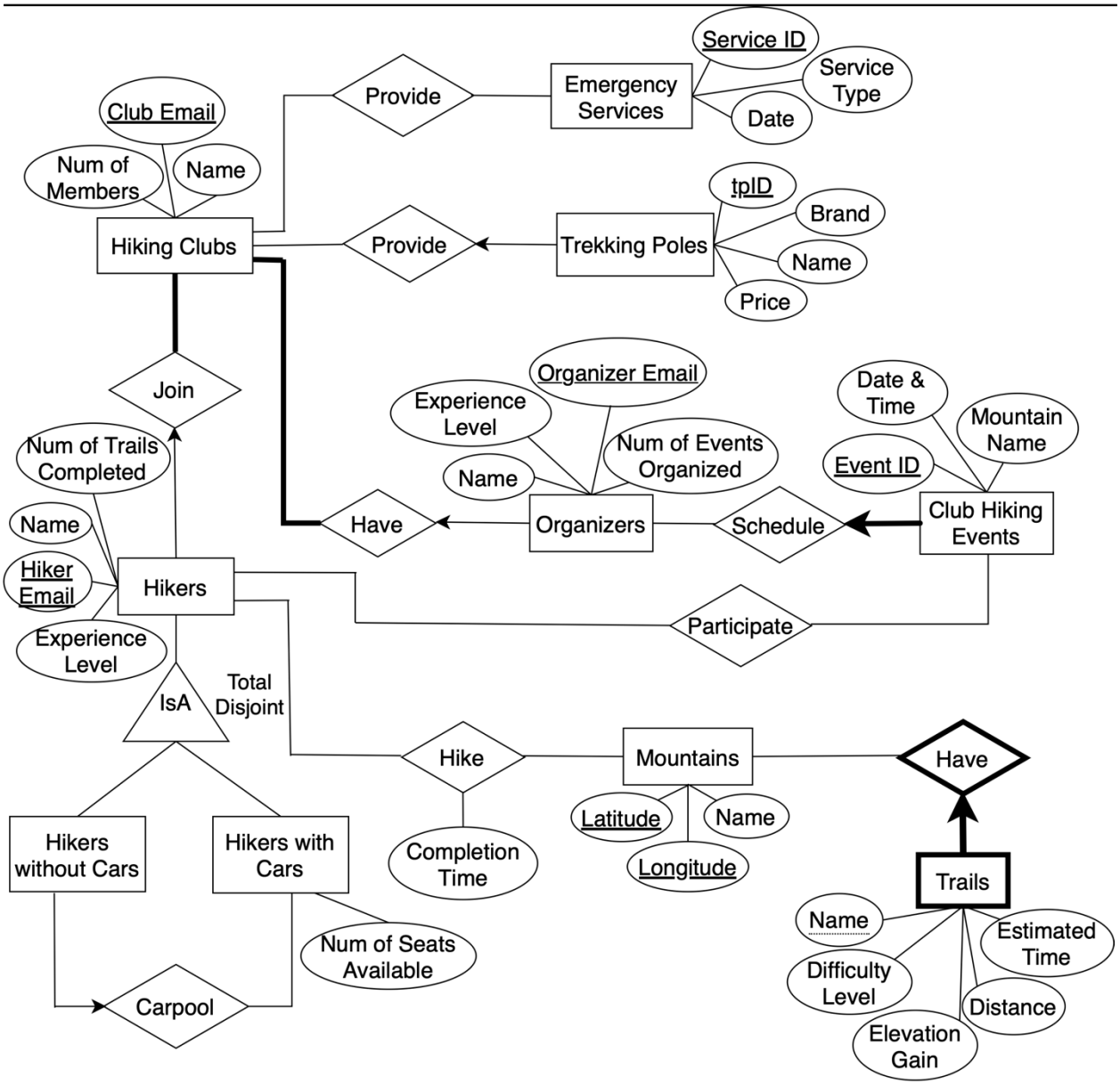
This project offers users a list of information about mountain trails and hiking clubs, along with their activities and services. Users can join the hiking clubs that schedule hiking events and provide trekking poles and emergency services. Additionally, users can join carpools.

3. The ER diagram you are basing your item #3 (below) on. This ER diagram may be the same as your milestone 1 submission or it might be different. If you have made changes from the version submitted in milestone 1, attach a note indicating what changes have been made and why.

If you have decided not to implement the suggestions given by your project mentor, please be sure to leave a note stating why. This is not to say that you must do everything that your project mentor says. In many instances, there are trade-offs between design choices and your decision may be influenced by different factors. Your TAs will often leave suggestions that are meant to help massage your project into a form that will fit with the requirements in future project milestones. If you choose not to take their advice, it would be helpful for them to know why to better assist the group moving forward.

Some changes have been made to the ER diagram:

- Incorporated feedback from M1: We created a primary key, 'Event ID,' for the entity 'Club Hiking Event,' because it is difficult to find a natural key. Multiple events can occur simultaneously at the same location.
- Slightly changed the naming of some attributes in the ER diagram to ensure consistency throughout the ER diagram, schema, and FDs. For example, '# members' has become 'Num of Members.'
- The attribute 'Num of Seats Available' was originally an attribute of 'Carpool,' but it has now become an attribute of 'Hikers with Cars' because we think it is more relevant to that entity.
- Added a few more attributes to several entities to intentionally create non-PK FDs.
    - Trekking Poles: added the attribute 'Name'
    - Organizers: added the attribute 'Experience Level'
    - Club Hiking Events: combined the attributes 'Date' and 'Time" into a single attribute, 'Date & Time'
    - Hikers: added the attribute 'Experience Level'
    - Trails: added the attribute 'Elevation Gain'



4. The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. The process should be reasonably straightforward. For each table:

a. List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...)). Make sure to include the domains for each attribute.
b. Specify the primary key (PK), candidate key (CK), foreign keys (FK), and other constraints (e.g., not null, unique, etc.) that the table must maintain.

University of British Columbia, Vancouver
Department of Computer Science

Primary keys (PK) are underlined, foreign keys (FK) are **bold**.
Candidate keys (CK) are the same as primary keys (PK) in this ER diagram. We don't have (non-PK) CKs.

- HikingClubs(ClubEmail: VARCHAR, Name: VARCHAR, NumofMembers: INTEGER)
- EmergencyServices(ServiceID: INTEGER, ServiceType: VARCHAR, Date: Date)
- Provide(**ClubEmail**: VARCHAR, **ServiceID**: INTEGER)
- Provide_TrekkingPoles(tpID: INTEGER, Brand: VARCHAR, Name: VARCHAR, Price: FLOAT, **ClubEmail**: VARCHAR)
- Have_Organizers(OrganizerEmail: VARCHAR, Name: VARCHAR, ExperienceLevel: CHAR(12), NumofEventsOrganized: INTEGER, **ClubEmail**: VARCHAR)
    - Total Participation constraint on the one side (Hiking Clubs) cannot be enforced now.
    - Domain of ExperienceLevel is CHAR(12) because it can be junior[6], intermediate[12], or senior[6]. And we define:
        - junior: NumofEventsOrganized is [0, 50)
        - intermediate: NumofEventsOrganized is [50, 100)
        - senior: NumofEventsOrganized is [100, infinity)
- Schedule_ClubHikingEvents(EventID: INTEGER, DateTime: DATETIME, MountainName: VARCHAR, **OrganizerEmail**: VARCHAR NOT NULL)
- Join_Hikers(HikerEmail: VARCHAR, Name: VARCHAR, ExperienceLevel: CHAR(12), NumofTrailsCompleted: INTEGER, **ClubEmail**: VARCHAR)
    - Total Participation constraint on the one side (Hiking Clubs) cannot be enforced now.
    - Domain of ExperienceLevel is CHAR(12) because it can be junior[6], intermediate[12], or senior[6]. And we define:
        - junior: NumofTrailsCompleted is [0, 15)
        - intermediate: NumofTrailsCompleted is [15, 30)
        - senior: NumofTrailsCompleted is [30, infinity)
- Participate(**HikerEmail**: VARCHAR, **EventID**: INTEGER)
- Mountains(Latitude: DECIMAL(8,6), Longitude: DECIMAL(9,6), Name: VARCHAR)
    - Credits (for domains of latitude and longitude): https://stackoverflow.com/questions/1196415/what-datatype-to-use-when-storing-latitude-and-longitude-data-in-sql-databases
- Hike(**HikerEmail**: VARCHAR, **Latitude**: DECIMAL(8,6), **Longitude**: DECIMAL(9,6), CompletionTime: TIME)
- Have_Trails(**Latitude**: DECIMAL(8,6), **Longitude**: DECIMAL(9,6), Name: VARCHAR, DifficultyLevel: CHAR(8), ElevationGain: INTEGER, Distance: FLOAT, EstimatedTime: TIME)
    - Domain of DifficultyLevel is CHAR(8) because it can be easy[4], moderate[8], or hard[4]. And we define:
        - easy: ElevationGain is [0, 1000)

        - moderate: ElevationGain is [1000, 1500)
        - hard: ElevationGain is [1500, infinity)
    - Units of Elevation Gain are in feet (ft) and Units of Distance are in miles (mi).
- HikersWithCars(**HikerEmail**: VARCHAR, NumofSeatsAvailable: INTEGER)
- Carpool_HikersWithoutCars(**HikerEmailWithoutCar**: VARCHAR, **HikerEmailWithCar**: VARCHAR)

5. Functional Dependencies (FDs)
    a. Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key).

    PKs and CKs are considered functional dependencies and should be included in the list of FDs. You do not need to include trivial FDs such as A → A.

    Note: In your list of FDs, there must be some kind of valid FD other those identified by a PK or CK. If you observe that no relations have FDs other than the PK and CK(s), then you will have to intentionally add some (meaningful) attributes to show valid FDs. We want you to get a good normalization exercise. Your design must go through a normalization process. You do not need to have a non-PK/CK FD for each relation but be reasonable. If your TA feels that some non-PK/CK FDs have been omitted, your grade will be adjusted accordingly.

- HikingClubs(ClubEmail, Name, NumofMembers)
    - ClubEmail → Name, NumofMembers

- EmergencyServices(ServiceID, ServiceType, Date)
    - ServiceID → ServiceType, Date

- Provide(**ClubEmail**, **ServiceID**)

- Provide_TrekkingPoles(tpID, Brand, Name, Price, **ClubEmail**)
    - tpID → Brand, Name, Price, ClubEmail
    - Brand, Name → Price

- Have_Organizers(OrganizerEmail, Name, ExperienceLevel, NumofEventsOrganized, **ClubEmail**)
    - OrganizerEmail → Name, ExperienceLevel, NumofEventsOrganized, ClubEmail
    - NumofEventsOrganized → ExperienceLevel

- Schedule_ClubHikingEvents(EventID, DateTime, MountainName, **OrganizerEmail**)
    - EventID → DateTime, MountainName, OrganizerEmail

- Join_Hikers(HikerEmail, Name, ExperienceLevel, NumofTrailsCompleted, **ClubEmail**)
  - HikerEmail → Name, ExperienceLevel, NumofTrailsCompleted, ClubEmail
  - NumofTrailsCompleted → ExperienceLevel

- Participate(**HikerEmail**, **EventID**)

- Mountains(Latitude, Longitude, Name)
  - Latitude, Longitude → Name

- Hike(**HikerEmail**, **Latitude**, **Longitude**, CompletionTime)
  - HikerEmail, Latitude, Longitude → CompletionTime

- Have_Trails(**Latitude**, **Longitude**, Name, DifficultyLevel, ElevationGain, Distance, EstimatedTime)
  - Latitude, Longitude, Name → DifficultyLevel, ElevationGain, Distance, EstimatedTime
  - ElevationGain → DifficultyLevel

- HikersWithCars(**HikerEmail**, NumofSeatsAvailable)
  - HikerEmail → NumofSeatsAvailable

- Carpool_HikersWithoutCars(**HikerEmailWithoutCar**, **HikerEmailWithCar**)
  - HikerEmailWithoutCar → HikerEmailWithCar

6. Normalization

a. Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization.

   You should show the steps taken for the decomposition in a manner similar to that done in class. Should there be errors, and no work is shown, no partial credit can be awarded without steps shown.

   The format should be the same as Step 3, with tables listed similar to Table1(attr1:domain1, attr2:domain2, …). ALL Tables must be listed, not only the ones post normalization.

Primary keys (PK) are underlined, foreign keys (FK) are **bold**.
Candidate keys (CK) are the same as primary keys (PK) in this ER diagram. We don't have (non-PK) CKs.

University of British Columbia, Vancouver
Department of Computer Science

The following relations are already in BCNF and 3NF because, for each of their non-trivial FDs, the attributes on the LHS form a superkey for that relation.

- HikingClubs(ClubEmail, Name, NumofMembers)
- EmergencyServices(ServiceID, ServiceType, Date)
- Provide(**ClubEmail**, **ServiceID**)
- Schedule_ClubHikingEvents(EventID, DateTime, MountainName, **OrganizerEmail**)
- Participate(**HikerEmail**, **EventID**)
- Mountains(Latitude, Longitude, Name)
- Hike(**HikerEmail**, **Latitude**, **Longitude**, CompletionTime)
- HikersWithCars(**HikerEmail**, NumofSeatsAvailable)
- Carpool_HikersWithoutCars(**HikerEmailWithoutCar**, **HikerEmailWithCar**)

We will decompose the rest:

- Provide_TrekkingPoles(tpID, Brand, Name, Price, **ClubEmail**)
    - tpID → Brand, Name, Price, ClubEmail
    - Brand, Name → Price

  The first FD does not violate BCNF because tpID is a superkey.
  The second FD violates BCNF since Brand, Name$^+$ = {Brand, Name, Price} and hence not a superkey.

  Decompose on Brand, Name → Price, and we get
    - Provide_TrekkingPoles1(tpID, **Brand**, **Name**, **ClubEmail**)
    - Provide_TrekkingPoles2(Brand, Name, Price)

- Have_Organizers(OrganizerEmail, Name, ExperienceLevel, NumofEventsOrganized, **ClubEmail**)
    - OrganizerEmail → Name, ExperienceLevel, NumofEventsOrganized, ClubEmail
    - NumofEventsOrganized → ExperienceLevel

  The first FD does not violate BCNF because OrganizerEmail is a superkey.
  The second FD violates BCNF since NumofEventsOrganized$^+$ = {NumofEventsOrganized, ExperienceLevel} and hence not a superkey.

  Decompose on NumofEventsOrganized → ExperienceLevel, and we get
    - Have_Organizers1(OrganizerEmail, Name, **NumofEventsOrganized**, **ClubEmail**)
    - Have_Organizers2(NumofEventsOrganized, ExperienceLevel)

- Join_Hikers(HikerEmail, Name, ExperienceLevel, NumofTrailsCompleted, **ClubEmail**)
    - HikerEmail → Name, ExperienceLevel, NumofTrailsCompleted, ClubEmail
    - NumofTrailsCompleted → ExperienceLevel

  The first FD does not violate BCNF because HikerEmail is a superkey.
  The second FD violates BCNF since NumofTrailsCompleted $^+$ = {NumofTrailsCompleted, ExperienceLevel} and hence not a superkey.

Decompose on NumofTrailsCompleted → ExperienceLevel, and we get
- Join_Hikers1(HikerEmail, Name, **NumofTrailsCompleted**, **ClubEmail**)
- Join_Hikers2(NumofTrailsCompleted, ExperienceLevel)

- Have_Trails(**Latitude**, **Longitude**, Name, DifficultyLevel, ElevationGain, Distance, EstimatedTime)
  - Latitude, Longitude, Name → DifficultyLevel, ElevationGain, Distance, EstimatedTime
  - ElevationGain → DifficultyLevel

  The first FD does not violate BCNF because Latitude, Longitude, and Name form a superkey.
  The second FD violates BCNF since ElevationGain $^{+}$ = {ElevationGain, DifficultyLevel} and hence not a superkey.

  Decompose on ElevationGain → DifficultyLevel, and we get
  - Have_Trails1(**Latitude**, **Longitude**, Name, **ElevationGain**, Distance, EstimatedTime)
  - Have_Trails2(ElevationGain, DifficultyLevel)

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.

Unless you know that you will always have exactly x characters for a given character, it is better to use the VARCHAR data type as opposed to a CHAR(Y). For example, UBC courses always use four characters to represent which department offers a course. In that case, you will want to use CHAR(4) for the department attribute in your SQL DDL statement. If you are trying to represent the name of a UBC course, you will want to use VARCHAR as the number of characters in a course name can vary greatly.

```sql
CREATE TABLE HikingClubs(
    ClubEmail VARCHAR PRIMARY KEY,
    Name VARCHAR,
    NumofMembers INTEGER
);

CREATE TABLE EmergencyServices(
    ServiceID INTEGER PRIMARY KEY,
    ServiceType VARCHAR,
    Date Date
);

CREATE TABLE Provide(
    ClubEmail VARCHAR,
    ServiceID INTEGER,
    PRIMARY KEY (ClubEmail, ServiceID),
    FOREIGN KEY (ClubEmail) REFERENCES HikingClubs
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (ServiceID) REFERENCES EmergencyServices
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Provide_TrekkingPoles1(
    tpID INTEGER PRIMARY KEY,
    Brand VARCHAR,
    Name VARCHAR,
    ClubEmail VARCHAR,
    FOREIGN KEY (ClubEmail) REFERENCES HikingClubs
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Provide_TrekkingPoles2(
    Brand VARCHAR,
    Name VARCHAR,
    Price FLOAT,
    PRIMARY KEY (Brand, Name)
);

CREATE TABLE Have_Organizers1(
    OrganizerEmail VARCHAR PRIMARY KEY,
    Name VARCHAR,
    NumofEventsOrganized INTEGER,
    ClubEmail VARCHAR,
    FOREIGN KEY (ClubEmail) REFERENCES HikingClubs
        ON DELETE SET NULL ON UPDATE CASCADE
);

CREATE TABLE Have_Organizers2(
    NumofEventsOrganized INTEGER PRIMARY KEY,
    ExperienceLevel  CHAR(12)
);

CREATE TABLE Schedule_ClubHikingEvents(
    EventID INTEGER PRIMARY KEY,
    DateTime DATETIME,
    MountainName VARCHAR,
    OrganizerEmail VARCHAR NOT NULL,
    FOREIGN KEY (OrganizerEmail) REFERENCES Have_Organizers1
        ON DELETE NO ACTION ON UPDATE CASCADE
);

CREATE TABLE Join_Hikers1(
    HikerEmail VARCHAR PRIMARY KEY,
    Name VARCHAR,
    NumofTrailsCompleted INTEGER,
    ClubEmail VARCHAR,
    FOREIGN KEY (ClubEmail) REFERENCES HikingClubs
        ON DELETE SET NULL ON UPDATE CASCADE
);

CREATE TABLE Join_Hikers2(
    NumofTrailsCompleted INTEGER PRIMARY KEY,
    ExperienceLevel  CHAR(12)
);

CREATE TABLE Participate(
    HikerEmail VARCHAR,
    EventID INTEGER,
    PRIMARY KEY (HikerEmail, EventID),
    FOREIGN KEY (HikerEmail) REFERENCES Join_Hikers1
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (EventID) REFERENCES Schedule_ClubHikingEvents
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Mountains(
    Latitude DECIMAL(8,6),
    Longitude DECIMAL(9,6),
    Name VARCHAR,
    PRIMARY KEY (Latitude, Longitude)
);

CREATE TABLE Have_Trails1(
    Latitude DECIMAL(8,6),
    Longitude DECIMAL(9,6),
    Name VARCHAR,
    ElevationGain INTEGER,
    Distance FLOAT,
    EstimatedTime TIME,
    PRIMARY KEY (Latitude, Longitude, Name),
    FOREIGN KEY (Latitude, Longitude) REFERENCES Mountains
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Have_Trails2(
    ElevationGain INTEGER PRIMARY KEY,
    DifficultyLevel CHAR(8)
);

CREATE TABLE Hike(
    HikerEmail VARCHAR,
    Latitude DECIMAL(8,6),
    Longitude DECIMAL(9,6),
    CompletionTime TIME,
    PRIMARY KEY (HikerEmail, Latitude, Longitude),
    FOREIGN KEY (HikerEmail) REFERENCES Join_Hikers1
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (Latitude, Longitude) REFERENCES Mountains
        ON DELETE SET NULL ON UPDATE CASCADE
);

CREATE TABLE HikersWithCars(
    HikerEmail VARCHAR PRIMARY KEY,
    NumofSeatsAvailable INTEGER,
    FOREIGN KEY (HikerEmail) REFERENCES Join_Hikers1
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Carpool_HikersWithoutCars(
    HikerEmailWithoutCar VARCHAR PRIMARY KEY,
    HikerEmailWithCar VARCHAR,
    FOREIGN KEY (HikerEmailWithoutCar) REFERENCES Join_Hikers1
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (HikerEmailWithCar) REFERENCES Join_Hikers1
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

---

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.

```
INSERT INTO HikingClubs
    VALUES ('AlpineClub123@gmail.com', 'Alpine', 10),
    ('BigHikingClub01@gmail.com', 'Big Hiking Club', 370),
    ('RidgeInfo@gmail.com', 'Ridge Hiking Club', 63),
    ('OutdoorClub@gmail.com', 'Outdoor Club', 150),
    ('Info@outdoor.ca', 'Outdoor Hiking Club', 63),
    ('AlpineHikers@gmail.com', 'Alpine', 25);

INSERT INTO EmergencyServices
    VALUES (1, 'Search and Rescue', '2017-01-10'),
    (2, 'Emergency Medical Services', '2019-03-08'),
    (13, 'Helicopter Rescue Services', '2020-11-07'),
    (16, 'Search and Rescue', '2020-11-07'),
    (20, 'Search and Rescue', '2020-11-07'),
    (100, 'Emergency Medical Services', '2024-10-15');

INSERT INTO Provide
    VALUES ('AlpineClub123@gmail.com', 2),
    ('BigHikingClub01@gmail.com', 1),
    ('RidgeInfo@gmail.com', 13),
    ('OutdoorClub@gmail.com', 100),
    ('AlpineHikers@gmail.com', 16);

INSERT INTO Provide_TrekkingPoles1
    VALUES (1, 'Black Diamond', 'Trail Trek Poles - Women', 'AlpineClub123@gmail.com'),
    (3, 'Black Diamond', 'Explorer 3 Trekking Poles', 'AlpineClub123@gmail.com'),
    (4, 'Salomon', 'Hacker', 'BigHikingClub01@gmail.com'),
    (10, 'Cline', 'Collapsible Trekking Pole', 'RidgeInfo@gmail.com'),
    (21, 'TheFitLife', 'Nordic Walking Trekking Poles', 'AlpineHikers@gmail.com');

INSERT INTO Provide_TrekkingPoles2
    VALUES ('Black Diamond', 'Trail Trek Poles - Women', 108.94),
    ('Black Diamond', 'Explorer 3 Trekking Poles', 99.95),
    ('Salomon', 'Hacker', 59.95),
    ('Cline', 'Collapsible Trekking Pole', 24.99),
    ('TheFitLife', 'Nordic Walking Trekking Poles', 39.98);

INSERT INTO Have_Organizers1
    VALUES ('angie456@gmail.com', 'Angie', 10, 'AlpineClub123@gmail.com'),
    ('2angie@gmail.com', 'Angie', 9, 'AlpineClub123@gmail.com'),
    ('gtp789@gmail.com', 'Lena', 123, 'BigHikingClub01@gmail.com'),
    ('andrew1@gmail.com', 'Andrew', 50, 'RidgeInfo@gmail.com'),
    ('luc556@gmail.com', 'Lucas', 23, 'OutdoorClub@gmail.com'),
    ('oliver@gmail.com', 'Oliver', 7, 'Info@outdoor.ca'),
    ('mmia8@gmail.com', 'Mia', 66, 'AlpineHikers@gmail.com');

INSERT INTO Have_Organizers2
    VALUES (10, 'junior'),
    (9, 'junior'),
    (123, 'senior'),
    (50, 'intermediate'),
    (23, 'junior'),
    (7, 'junior'),
    (66, 'intermediate');

INSERT INTO Schedule_ClubHikingEvents
    VALUES (1, '2017-01-10 13:00:00', 'Stanley Park', '2angie@gmail.com'),
    (5, '2017-01-10 14:25:00', 'Stanley Park', 'gtp789@gmail.com'),
    (100, '2020-11-07 08:30:00', 'Stawamus Chief Provincial Park', 'angie456@gmail.com'),
    (116, '2022-08-16 07:00:00', 'Stawamus Chief Provincial Park', 'mmia8@gmail.com'),
    (202, '2024-09-25 10:45:00', 'Burnaby Mountain', 'oliver@gmail.com');
```

```
INSERT INTO Join_Hikers1
    VALUES ('8leo9@gmail.com', 'Leo', 3, 'AlpineClub123@gmail.com'),
    ('olivia20@gmail.com', 'Olivia', 25, 'BigHikingClub01@gmail.com'),
    ('ttpp11@gmail.com', 'Emily', 40, 'BigHikingClub01@gmail.com'),
    ('henry0@gmail.com', 'Henry', 29, 'RidgeInfo@gmail.com'),
    ('sophia55@gmail.com', 'Sophia', 5, 'OutdoorClub@gmail.com'),
    ('will678@gmail.com', 'William', 14, 'Info@outdoor.ca'),
    ('emma@gmail.com', 'Emma', 61, 'AlpineHikers@gmail.com'),
    ('noah56@gmail.com', 'Noah', 9, 'AlpineHikers@gmail.com'),
    ('amelia10@gmail.com', 'Amelia', 27, 'RidgeInfo@gmail.com'),
    ('kathy3@gmail.com', 'Kathy', 50, 'RidgeInfo@gmail.com');

INSERT INTO Join_Hikers2
    VALUES (3, 'junior'),
    (25, 'intermediate'),
    (40, 'senior'),
    (29, 'intermediate'),
    (5, 'junior'),
    (14, 'junior'),
    (61, 'senior'),
    (9, 'junior'),
    (27, 'intermediate'),
    (50, 'senior');

INSERT INTO Participate
    VALUES ('8leo9@gmail.com', 5),
    ('olivia20@gmail.com', 5),
    ('ttpp11@gmail.com', 116),
    ('ttpp11@gmail.com', 202),
    ('sophia55@gmail.com', 100);

INSERT INTO Mountains
    VALUES (49.299999, -123.139999, 'Stanley Park'),
    (49.331602, -123.263650, 'Lighthouse Park'),
    (49.686389, -123.136944, 'Stawamus Chief Provincial Park'),
    (49.954213, -123.013532, 'Panorama Ridge'),
    (49.279369, -122.908605, 'Burnaby Mountain');

INSERT INTO Have_Trails1
    VALUES (49.299999, -123.139999, 'Stanley Park Seawall', 249, 6.0, '01:58:00'),
    (49.299999, -123.139999, 'Stanley Park Inner Loop', 337, 4.9, '01:45:00'),
    (49.686389, -123.136944, 'Sea to Summit Trail', 3136, 7.2, '05:23:00'),
    (49.686389, -123.136944, 'Stawamus Chief First, Second, Thrid Peak Loop', 2142, 3.6, '04:50:00'),
    (49.279369, -122.908605, 'Burnaby Mountain Park Loop', 1486, 6.9, '03:32:00');

INSERT INTO Have_Trails2
    VALUES (249, 'easy'),
    (337, 'easy'),
    (3136, 'hard'),
    (2142, 'hard'),
    (1486, 'moderate');

INSERT INTO Hike
    VALUES ('8leo9@gmail.com', 49.299999, -123.139999, '02:00:00'),
    ('olivia20@gmail.com', 49.299999, -123.139999, '01:45:23'),
    ('ttpp11@gmail.com', 49.686389, -123.136944, '05:47:11'),
    ('henry0@gmail.com', 49.686389, -123.136944, '05:01:00'),
    ('sophia55@gmail.com', 49.279369, -122.908605, '03:55:03');
```

```
262 INSERT INTO HikersWithCars
263     VALUES ('8leo9@gmail.com', 2),
264     ('olivia20@gmail.com', 3),
265     ('ttpp11@gmail.com', 2),
266     ('henry0@gmail.com', 4),
267     ('sophia55@gmail.com', 2);
268
269
270 INSERT INTO Carpool_HikersWithoutCars
271     VALUES ('will678@gmail.com', '8leo9@gmail.com'),
272     ('emma@gmail.com', '8leo9@gmail.com'),
273     ('noah56@gmail.com', 'ttpp11@gmail.com'),
274     ('amelia10@gmail.com', 'henry0@gmail.com'),
275     ('kathy3@gmail.com', 'henry0@gmail.com');
```