CPSC 304 Project Cover Page

Milestone #: 2
Date: Thu, Feb 29
Group Number: 7

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Linda Han	81054231	z7m9j	hanlinda0903@gmail.com
Yudan Chen	74330309	x6a2g	cydan199@student.ubc.ca
Mike Lu	72573983	e0s9m	mike020830@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. Summary

Our project entails a **ski resort operation system**, it focuses on several components of a ski resort such as the people (skiers and rescue team), facilities & programs (lift, ski slope, ski lessons), and environmental elements (weather condition and wildlife).

3. ERD

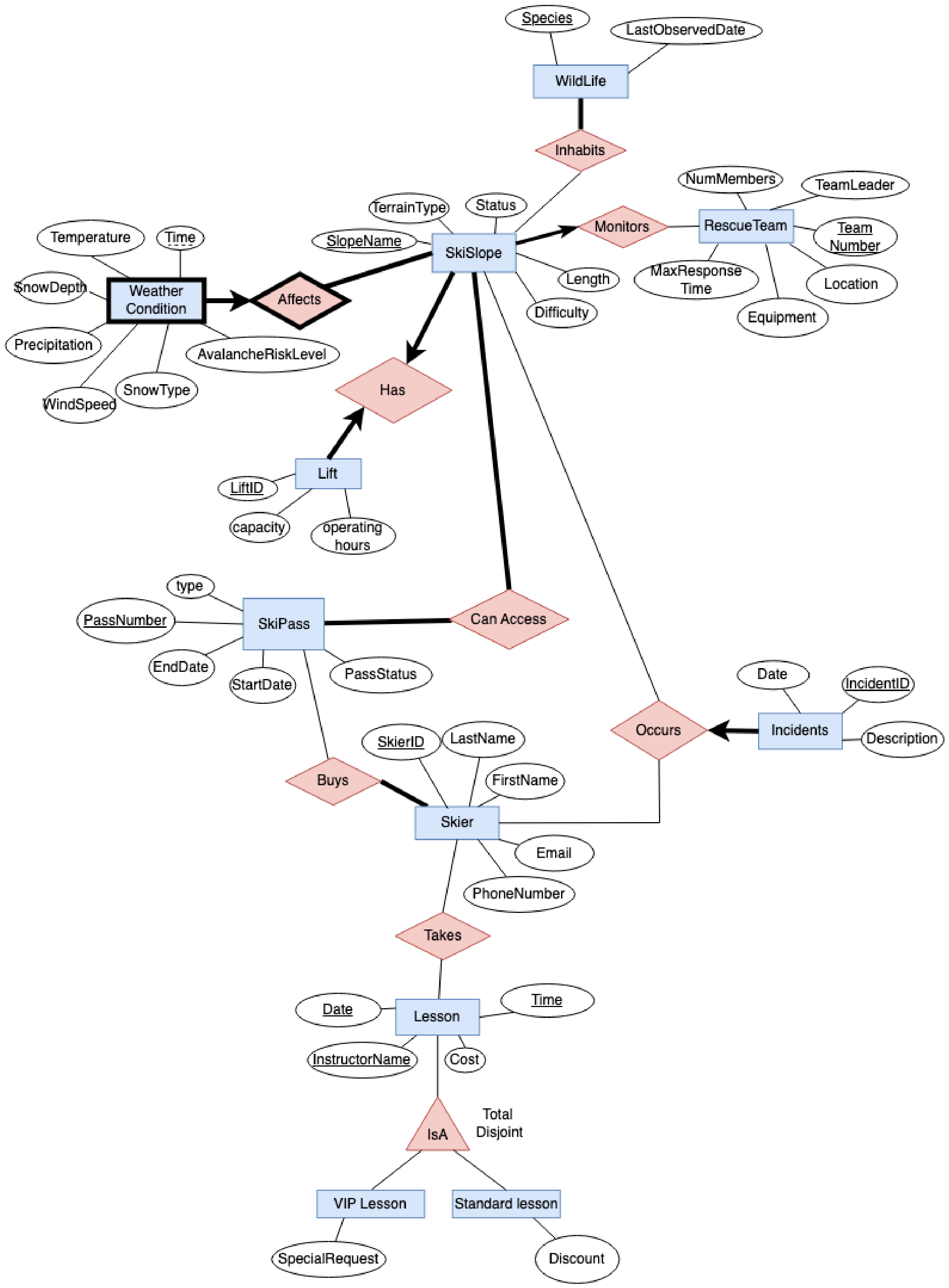
We decided to add the following changes to our ERD from milestone 1:

1) Converted **WeatherCondition** to be a weak entity as its key <u>Time</u>* alone cannot uniquely identify the weather condition (there could be other slopes with a weather condition recorded at the same time). Instead, the combination of <u>Time</u> and <u>SlopeName</u> of the slope is what uniquely identifies the entity.

2) Add total participation to **SkiSlope** in the **Affects** relationship as every slope must have a weather condition report.

3) Remove entity **MedicalHistory**, as it doesn't have a lot of attributes so keeping it doesn't add meaningful complexity to our project.

4) The Name attribute in **SkiSlope** changed to SlopeName to avoid ambiguity


Species
LastObservedDate
WildLife
Inhabits
NumMembers
TeamLeader
RescueTeam
Team Number
Location
Equipment
MaxResponse Time
Monitors
TerrainType
Status
SlopeName
SkiSlope
Length
Difficulty
Temperature
Time
SnowDepth
Weather Condition
Affects
Precipitation
AvalancheRiskLevel
WindSpeed
SnowType
Has
Lift
LiftID
capacity
operating hours
type
PassNumber
SkiPass
Can Access
EndDate
StartDate
PassStatus
Date
IncidentID
Incidents
Description
Occurs
SkierID
LastName
FirstName
Buys
Skier
Email
PhoneNumber
Takes
Date
Lesson
Time
InstructorName
Cost
IsA
Total
Disjoint
VIP Lesson
Standard lesson
SpecialRequest
Discount

4. Schema Before Decomposition

<u>Underline</u>: Primary Key, **Bold**: Foreign Key

Relation	Primary Key (and Candidate Key)	Foreign Key	Other constraints

WeatherCondition(
  <u>Time</u>: Timestamp,
  <u>SlopeName</u>: varchar,
  Temperature: Int,
  SnowDepth: Int,
  Precipitation: Int,
  WindSpeed: Int,
  SnowType: varchar,
  AvalancheRiskLevel: Int,
)
	- (Time (partial key), SlopeName)	**SlopeName** references Ski Slope	To enforce total participation: add ON DELETE CASCADE for foreign key SlopeName

SkiSlope(
  <u>SlopeName</u>: varchar,
  TerrainType: varchar,
  Status: varchar,
  Length: int,
  Difficulty: varchar,
  TeamNumber: int,
  LiftID: varchar
)
	- SlopeName is the primary key
	- LiftID is a Candidate Key
		TeamNumber references RescueTeam,
		
		LiftID references Lift
			TeamNumber is not null,
			LiftID is not null and unique

RescueTeam(
  <u>TeamNumber</u>: int,
  Location: varchar,
  MaxResponseTime: int,
  Equipment: varchar,
  TeamLeader: varchar,
  NumMembers: int
)
	TeamNumber	-	-

Lift(
  <u>LiftID</u>: varchar,
  Capacity: Int,
  OperatingHours: int
)
	LiftID	-	-

SkiPass(PassNumber: Int, StartDate: Date, EndDate: Date, PassStatus: varchar, Type: varchar)	PassNumber	-	Need assertion to enforce total participation in Can Access relationship
Skier(SkierID: Int, LastName: varchar, FirstName: varchar, Email: varchar, PhoneNumber: varchar)	SkierID (Primary Key), Email (Candidate Key), PhoneNumber (Candidate Key)	-	Need assertion to enforce total participation in Buys relationship Unique on Email Unique on PhoneNumber
IncidentsOccurs(IncidentID: Int, Date: Date, Description: varchar, **SlopeName:** varchar, **SkierID**: Int)	IncidentID	**SlopeName** references Ski Slope, **SkierID** references Skier	SlopeName and SkierID both are not null
Wildlife(Species: varchar, LastObservedDate: Date)	Species	-	Need assertion to enforce total participation in Inhabits relationship
Lesson(Date: Date, Time: Time, InstructorName: varchar, Cost: Int)	(Date, Time, InstructorName)	-	Need assertions to enforce total and disjoint constraints
VIPLesson(**Date**: Date, **Time**, Time, **InstructorName**:varchar, SpecialRequest: varchar)	(Date, Time, InstructorName)	(Date, Time, InstructorName) References Lesson	Use on delete cascade for foreign key Need assertions to enforce total and disjoint constraints

StandardLesson(**<u>Date</u>**: Date, **<u>Time</u>**, Time, **<u>InstructorName</u>**:varchar Discount: Int)	(Date, Time, InstructorName)	(Date, Time, InstructorName) References Lesson	- Use on delete cascade for foreign key - Need assertions to enforce total and disjoint constraints
CanAccess(**<u>PassNumber</u>**: Int, **<u>SlopeName</u>**: varchar)	(PassNumber, SlopeName)	**PassNumber** references SkiPass, **SlopeName** references SkiSlope	Can't capture total participation for now, need assertions
Buys(**<u>SkierID</u>**: Int, **<u>PassNumber</u>**: Int)	(SkierID, PassNumber)	**SkierID** references Skier, **PassNumber** references SkiPass	Can't capture total participation for now, need assertions
Takes(**<u>SkierID</u>**: Int, **<u>InstructorName</u>**: varchar, **<u>Date</u>**: Date, **<u>Time</u>**: Time)	(SkierID, InstructorName,Date, Time)	**SkierID** references Skier, **(InstructorName, Date, Time**) references Lesson	-
Inhabits(**<u>SlopeName</u>**: varchar, **<u>Species</u>**: varchar)	(SlopeName, Species)	SlopeName, Species	Can't capture total participation for now, need assertions

5. Functional Dependencies

Note: Highlighted FDs show the dependency given by the primary key or the candidate

Weather Condition:
Precipitation → SnowType, SnowDepth
WindSpeed → Temperature
Temperature → SnowType, Precipitation
Temperature, Precipitation, SnowDepth, SnowType, WindSpeed → AvalancheRiskLevel
SlopeName, Time → Time, SlopeName, Temperature, SnowDepth, Precipitation, WindSpeed, SnowType, AvanlacheRiskLevel

Ski Pass:
Type, StartDate → EndDate
StartDate, EndDate → PassStatus
PassNumber → PassNumber, Type, EndDate, StartDate, PassStatus

Ski Slope:
TerrainType, Length → Difficulty
SlopeName → SlopeName, TerrainType, Status, Length, Difficulty, LiftID, TeamNumber

Candidate key(Non-PK) FD:
LiftID → SlopeName, TerrainType, Status, Length, Difficulty, LiftID, TeamNumber

Rescue Team:
Location → Maximum Response Time, Equipment
TeamNumber → TeamNumber, NumberMembers, TeamLeader, Location, Equipment, MaxResponseTime

Skier:
SkierID → SkierID, LastName, FirstName, Email, PhoneNumber

Candidate key(Non-PK) FD:
Email → SkierID, LastName, FirstName, Email, PhoneNumber
PhoneNumber → SkierID, LastName, FirstName, Email, PhoneNumber

IncidentOccurs:
IncidentID → IncidentID, Date, Description, SlopeName, SkierID

WildLife:
Species → Species, LastObservedDate

Lift:
LiftID → LiftID, Capacity, OperatingHours

Lesson:
Date, Time, InstructorName → Date, Time, InstructorName, Cost

VIP Lesson:
Date, Time, InstructorName → Date, Time, InstructorName, SpecialRequest

*Note that we didn't include Cost here since it is a common attribute which is only present in the parent Lesson table.

Standard Lesson:
Date, Time, InstructorName → Date, Time, InstrutorName, Discount

*Note that we didn't include Cost here since it is a common attribute which is only present in the parent Lesson table.

CanAccess:
PassNumber, SlopeName → PassNumber, SlopeName

Buys:
SkierID, PassNumber → SkierID, PassNumber

Takes:
SkierID, Date, Time, InstructorName, Time → SkierID, Date, Time, InstructorName, Time

Inhabits:
Species, SlopeName → Species, SlopeName

6. Schema After Decomposition (using BCNF)

Weather Condition Decomposition

FD1: Precipitation → SnowType, SnowDepth
FD2: WindSpeed → Temperature
FD3: Temperature → SnowType, Precipitation
FD4: Temperature, Precipitation, SnowDepth, SnowType, WindSpeed → AvalancheRiskLevel
FD5: SlopeName, Time → Time, SlopeName, Temperature, SnowDepth, Precipitation, WindSpeed, SnowType, AvanlacheRiskLevel

Since the LHS of FD1 (Precipitation) is not a super key, we decompose based on FD1:

> **R1**(Precipitation, SnowType, SnowDepth)
> R2(Time, SlopeName, Temperature, Precipitation, WindSpeed, AvalancheRiskLevel)

R1 is now in BCNF, but an implicit FD from FD3 (Temperature → Precipitation) is a violating FD for R2 since Temperature is not a super key in R2, so we decompose based on that:

> **R3**(Temperature, Precipitation)
> R4(Time, SlopeName, Temperature, WindSpeed, AvalancheRiskLevel)

R3 is now in BCNF, but we have FD2 which is a violating FD for R4 since WindSpeed is not a super key in R4. We decompose R4 based on the FD2:

> **R5**(WindSpeed, Temperature)
> R6(Time, SlopeName, WindSpeed, AvalancheRiskLevel)

R5 is now in BCNF, but we still have an implicit FD (WindSpeed → AvalancheRiskLevel), and WindSpeed is not a super key in R6, therefore R6 is not in BCNF, and we decompose R6 based on the implicit FD:

R7(WindSpeed, AvalancheRiskLevel)
R8(Time, SlopeName, WindSpeed)

R7 and R8 are both in BCNF, since there are no more violating FDs.

We have completed the decomposition for WeatherCondition, and we have bolded the relevant Relations after decomposition. But since we see that R5 and R7 all have the same Primary Key, and we see that if we combine them, we won't cause violation of BCNF. So we combine R5 and R7 to form a new **R9**(WindSpeed, Temperature, AvalancheRiskLevel).

The final relations (renamed) after decomposition is:

PrecipitationInformation = **R1**(Precipitation, SnowType, SnowDepth)
TemperatureInformation = **R3**(Temperature, **Precipitation**)
WindSpeedInformation = **R9**(WindSpeed, **Temperature**, AvalancheRiskLevel)
WeatherCondition = **R8**(Time, **SlopeName**, **WindSpeed**)

Now we describe them in detail below:

WeatherCondition(
Time: Timestamp,
SlopeName: varchar,
WindSpeed: Int
)
Primary Key:
- (Time, SlopeName)
- Time is a partial key

Foreign Key:
- **SlopeName** references SkiSlope (use on delete cascade)
- **WindSpeed** references WindSpeedInformation (use on delete no action)

WindSpeedInformation(
WindSpeed: Int,
Temperature: Int,
AvalancheRiskLevel: Int
)
Primary Key:
- WindSpeed

Foreign Key:
- **Temperature** References TemperatureInformation (use on delete no action)

TemperatureInformation(
Temperature: Int,
Precipitation: Int

)
Primary Key:
- <u>Temperature</u>

Foreign Key:
- **Precipitation** References the relation PrecipitationInformation (Use on delete no action)

PrecipitationInformation(
    <u>Precipitation</u>: Int,
    SnowType: varchar,
    SnowDepth: Int
)
Primary Key:
- <u>Precipitation</u>

SkiPass Decomposition

FD1: Type, StartDate → EndDate
FD2: StartDate, EndDate → PassStatus
FD3: PassNumber → PassNumber, Type, EndDate, StartDate, PassStatus

FD1 violates BCNF because the LHS(Type, StartDate) is not a super key, so we decompose it to:

R1(<u>Type</u>, <u>StartDate</u>, EndDate)
R2(<u>PassNumber,</u> Type, StartDate, PassStatus)

R2 violates BCNF because the implicit FD (Type, StartDate → PassStatus) applies to this relation and its LHS is not a super key:

R3(<u>Type</u>, <u>StartDate</u>, PassStatus)
R4(Type, StartDate, <u>PassNumber</u>)

After decomposition we get the following:

R1(<u>Type</u>, <u>StartDate</u>, EndDate),
R3(<u>Type</u>, <u>StartDate</u>, PassStatus),
R4(Type, StartDate, <u>PassNumber</u>)

But we see that R1 and R3 have the same primary key, and combining them won't violate BCNF. So we combine them and get **R5**(<u>Type</u>, <u>StartDate</u>, EndDate, PassStatus).

The final result (renamed) is:

PassType = **R5**(<u>Type</u>, <u>StartDate</u>, EndDate, PassStatus), SkiPass = **R4**(**Type**, **StartDate**, PassNumber)

Now we describe the decomposed relations in detail:

SkiPass(
    PassNumber: Int,
    StartDate: Date,
    Type: varchar
)

Primary Key:
- PassNumber

Foreign Key:
- (StartDate, Type) References PassType (use On delete no action)

PassType(
    Type: varchar,
    StartDate: Date,
    EndDate: Date,
    PassStatus: varchar
)

Primary Key:
- (Type, StartDate)

SkiSlope Decomposition

FD1: TerrainType, Length → Difficulty
FD2: SlopeName → SlopeName, TerrainType, Status, Length, Difficulty, LiftID, TeamNumber
FD3: LiftID → SlopeName, TerrainType, Status, Length, Difficulty, LiftID, TeamNumber

FD1 violates the BCNF because the LHS is not a superkey, so we decompose it to:

R1(TerrainType, Length, Difficulty)
R2(SlopeName, TerrainType, Status, Length, TeamNumber, LiftID)

SkiSlope(
    SlopeName: varchar,
    TerrainType: varchar,
    Status: varchar,
    Length: Int,
    TeamNumber: Int NOT NULL
    LiftId: varchar NOT NULL UNIQUE

)

Primary Key:
- SlopeName

Candidate Key(Non-PK):
- LiftID is a candidate key here since it is unique

Foreign Key:
- **TeamNumber** references RescueTeam (use ON UPDATE CASCADE)
- **LiftID** references Lift (use ON UPDATE CASCADE)
- (**TerrainType**, **Length**) references Terrain (use ON DELETE NO ACTION)

Note: Can't capture total participation for now(every ski slope must have at least 1 weather condition).

Terrain(
    <u>TerrainType</u>: varchar,
    <u>Length</u>: Int,
    Difficulty: varchar
)

Primary Key:
- (<u>TerrainType</u>, <u>Length</u>)

<u>**RescueTeam Decomposition**</u>

FD1: Location → Maximum Response Time, Equipment
FD2: TeamNumber → TeamNumber, NumberMembers, TeamLeader, Location, Equipment, MaxResponseTime

FD1 violates the BCNF because the LHS is not a superkey, so we decompose it to:

RescueLocation = **R1**(<u>Location</u>, MaximumResponseTime, Equipment)
RescueTeam = **R2**(<u>TeamNumber</u>, Location, TeamLeader, NumMembers)

RescueTeam(
    <u>TeamNumber</u>: Int,
    Location: varchar,
    TeamLeader: varchar,
    NumMembers: Int
)

Primary Key:
- TeamNumber

Foreign Key:

- **Location** references RescueLocation (use ON DELETE NO ACTION)

RescueLocation(
    Location: varchar,
    MaxResponseTime: Int,
    Equipment: varchar
)

Primary Key:
- Location

7. SQL DDL statements (Foreign key referential constraint default to: on delete no action, on update no action)

```
CREATE TABLE WeatherCondition (
    Time          TIMESTAMP,
    SlopeName     VARCHAR(255),
    WindSpeed     INT,

    PRIMARY KEY (Time, SlopeName),
    FOREIGN KEY (SlopeName) REFERENCES SkiSlope(SlopeName)
            ON DELETE CASCADE,
    FOREIGN KEY (WindSpeed) REFERENCES WindSpeedInformation(WindSpeed)
);


CREATE TABLE WindSpeedInformation (
    WindSpeed           INT PRIMARY KEY,
    Temperature         INT,
    AvalancheRiskLevel  INT,

    FOREIGN KEY (Temperature) REFERENCES TemperatureInformation(Temperature)
);


CREATE TABLE TemperatureInformation (
    Temperature         INT PRIMARY KEY,
    Precipitation       INT,

    FOREIGN KEY (Precipitation) REFERENCES
                  PrecipitationInformation(Precipitation)
);
```

```
CREATE TABLE PrecipitationInformation (
    Precipitation       INT PRIMARY KEY,
    SnowType            VARCHAR(255),
    SnowDepth           INT
);

CREATE TABLE SkiSlope (
    SlopeName           VARCHAR(255) PRIMARY KEY,
    TerrainType         VARCHAR(255),
    Status              VARCHAR(255),
    Length              INT,
    TeamNumber          INT NOT NULL,
    LiftID              VARCHAR(255) NOT NULL UNIQUE,

    FOREIGN KEY (TerrainType, Length) REFERENCES Terrain(TerrainType,
Length),
    FOREIGN KEY (TeamNumber) REFERENCES RescueTeam(TeamNumber)
            ON UPDATE CASCADE,
    FOREIGN KEY (LiftID) REFERENCES Lift(LiftID)
            ON UPDATE CASCADE
);

CREATE TABLE SkiPass(
      PassNumber        INT PRIMARY KEY,
      StartDate         DATE,
      Type              VARCHAR(255),

      FOREIGN KEY (Type, StartDate) REFERENCES PassType(Type, StartDate)
);


CREATE TABLE PassType (
    Type          VARCHAR(255),
    StartDate     DATE,
    EndDate       DATE,
    PassStatus    VARCHAR(255),

    PRIMARY KEY(Type, StartDate)
);
```

```
CREATE TABLE Lift (
      LiftID            VARCHAR(255) PRIMARY KEY,
      Capacity          INT,
      OperatingHours    INT
);


CREATE TABLE Terrain (
      TerrainType       VARCHAR(255),
      Length            INT,
      Difficulty        VARCHAR(255),

      PRIMARY KEY (TerrainType, Length)
);


CREATE TABLE RescueLocation (
      Location          VARCHAR(255) PRIMARY KEY,
      MaxResponseTime   INT,
      Equipment         VARCHAR(255)
);

CREATE TABLE RescueTeam (
      TeamNumber        INT PRIMARY KEY,
      TeamLeader        VARCHAR(255) DEFAULT NULL,
      Location          VARCHAR(255) DEFAULT NULL,
      NumMembers        INT,

      FOREIGN KEY (Location) REFERENCES RescueLocation(Location)
);
```

8. INSERT statements

```
INSERT INTO RescueLocation (Location, MaxResponseTime, Equipment) VALUES
('Basecamp Ranger Station', 01, 'First Aid Kit'),
('Summit Watchtower', 15, 'Radio Communication Set'),
('Northern Trailhead', 20, 'GPS Devices'),
('Southern Access Point', 25, 'Emergency Flares'),
('East Ridge Outpost', 30, 'Avalanche Probes');

INSERT INTO RescueTeam (TeamNumber, Location, TeamLeader, NumMembers) VALUES
(1, 'Basecamp Ranger Station', 'John Smith', 10),
(2, 'Summit Watchtower', 'Jane Doe', 5),
(3, 'Northern Trailhead', 'Mike Johnson', 20),
(4, 'Southern Access Point', 'Emily Davis', 10),
(5, 'East Ridge Outpost', 'David Wilson', 6);

INSERT INTO Lift (LiftID, Capacity, OperatingHours) VALUES
('LiftA', 8, 10),
('LiftB', 4, 12),
('LiftC', 6, 8),
('LiftD', 10, 9),
('LiftE', 5, 11);

INSERT INTO Terrain (TerrainType, Length, Difficulty) VALUES
('Alpine', 500, 'Intermediate'),
('Freestyle', 400, 'Advanced'),
('Groomed', 600, 'Beginner'),
('Off-Piste', 700, 'Expert'),
('Backcountry', 800, 'Expert');

INSERT INTO SkiSlope (SlopeName, TerrainType, Status, Length, TeamNumber, LiftId)
VALUES
('Slope 1', 'Alpine', 'Open', 500, 1, 'LiftA'),
('Slope 2', 'Freestyle', 'Closed', 400, 2, 'LiftB'),
('Slope 3', 'Groomed', 'Open', 600, 3, 'LiftC'),
('Slope 4', 'Off-Piste', 'Open', 700, 4, 'LiftD'),
('Slope 5', 'Backcountry', 'Closed', 800, 5, 'LiftE');

INSERT INTO PassType (Type, StartDate, EndDate, PassStatus) VALUES
('Seasonal', '2023-11-01', '2024-04-30', 'Active'),
```

```sql
('Weekly', '2024-01-01', '2024-01-07', 'Active'),
('Daily', '2024-02-01', '2024-02-01', 'Expired'),
('Weekend', '2024-02-05', '2024-02-06', 'Active'),
('Holiday', '2023-12-23', '2023-12-26', 'Expired');

INSERT INTO SkiPass (PassNumber, StartDate, Type) VALUES
(101, '2023-11-01', 'Seasonal'),
(102, '2024-01-01', 'Weekly'),
(103, '2024-02-01', 'Daily'),
(104, '2024-02-05', 'Weekend'),
(105, '2023-12-23', 'Holiday');

INSERT INTO PrecipitationInformation (Precipitation, SnowType, SnowDepth) VALUES
(0, 'No precipitation', 0),
(10, 'Light snow', 10),
(20, 'Moderate snow', 15),
(30, 'Heavy snow', 30),
(40, 'Blizzard', 50);

INSERT INTO TemperatureInformation (Temperature, Precipitation) VALUES
(-5, 0),   -- Below freezing, no precipitation
(-2, 10),  -- Below freezing, light snow
(-10, 20), -- Well below freezing, moderate snow
(-7, 30),  -- Below freezing, heavy snow
(-15, 40); -- Well below freezing, blizzard

INSERT INTO WindSpeedInformation (WindSpeed, Temperature, AvalancheRiskLevel)
VALUES
(5, -5, 1),   -- Calm winds, low avalanche risk
(10, -2, 2),  -- Light winds, moderate avalanche risk
(20, -10, 3), -- Moderate winds, high avalanche risk
(15, -7, 2),  -- Light winds, moderate avalanche risk
(25, -15, 4); -- Strong winds, very high avalanche risk

INSERT INTO WeatherCondition (Time, SlopeName, WindSpeed) VALUES
('2024-02-28 09:00:00', 'Slope 1', 5),
('2024-02-28 09:00:00', 'Slope 2', 10),
('2024-02-28 11:00:00', 'Slope 3', 20),
('2024-02-28 11:00:00', 'Slope 4', 15),
('2024-02-28 13:00:00', 'Slope 5', 25);

INSERT INTO Skier (SkierID, LastName, FirstName, Email, PhoneNumber) VALUES
(1, 'Li', 'Joe', 'joeli@gmail.com', ‘7786668888’),
(2, 'Wang', 'Jason', 'jason@gmail.com', ‘7781231234’),
(3, 'Zhang', 'Ken', 'ken@gmail.com', ‘7789990909’),
(4, 'Lai', 'Jonathan', 'jk@gmail.com', ‘13300918291’),
(5, 'Zheng', 'Bob', 'bz@gmail.com', ‘18899087866’);


INSERT INTO IncidentsOccurs (IncidentID, Date, Description, SlopeName, SkierID) VALUES
(1, '2023-11-01', 'broke leg', 'Slope 1', 1),
(2, '2023-11-21', 'broke arm', 'Slope 2', 3),
(3, '2023-11-03', 'broke leg', 'Slope 1', 4),
(4, '2023-11-05', 'heart attack', 'Slope 3', 2),
(5, '2023-11-01', 'broke arm', 'Slope 2', 3);

INSERT INTO Wildlife (Species, LastObservedDate) VALUES
('Elk', '2024-02-25'),
('Mountain Goat', '2024-02-20'),
('Snowshoe Hare', '2024-02-18'),
('Lynx', '2024-02-15'),
('Brown Bear', '2024-02-10');

INSERT INTO Lesson (Date, Time, InstructorName, Cost) VALUES
('2024-03-01', '09:00:00', 'Emily Johnson', 100),
('2024-03-01', '10:00:00', 'Michael Smith', 120),
('2024-03-02', '11:00:00', 'Sophia Brown', 110),
('2024-03-02', '12:00:00', 'Daniel Garcia', 90),
('2024-03-03', '13:00:00', 'Olivia Martinez', 95),
('2024-03-04', '09:00:00', 'Lucas Allen', 70),
('2024-03-04', '10:30:00', 'Eva Turner', 60),
('2024-03-05', '11:00:00', 'Grace Lee', 50),
('2024-03-05', '13:30:00', 'Samuel Walker', 80),
('2024-03-06', '14:00:00', 'Emma Thomas', 90);

INSERT INTO VIPLesson (Date, Time, InstructorName, SpecialRequest) VALUES
('2024-03-01', '09:00:00', 'Emily Johnson', 'Private slope session'),
('2024-03-01', '10:00:00', 'Michael Smith', 'Video analysis of technique'),
('2024-03-02', '11:00:00', 'Sophia Brown', 'Focus on carving skills'),
('2024-03-02', '12:00:00', 'Daniel Garcia', 'Early access to lifts'),
('2024-03-03', '13:00:00', 'Olivia Martinez', 'Extended lesson time');

INSERT INTO StandardLesson (Date, Time, InstructorName, Discount) VALUES
('2024-03-04', '09:00:00', 'Lucas Allen', 10),
('2024-03-04', '10:30:00', 'Eva Turner', 20),
('2024-03-05', '11:00:00', 'Grace Lee', 20),
('2024-03-05', '13:30:00', 'Samuel Walker', 15),
('2024-03-06', '14:00:00', 'Emma Thomas', 5);
```

```
INSERT INTO CanAccess (PassNumber, SlopeName) VALUES
(101, 'Slope 1'),
(102, 'Slope 2'),
(103, 'Slope 3'),
(104, 'Slope 4'),
(105, 'Slope 5');

INSERT INTO Buys (SkierID, PassNumber) VALUES
(1, 101),
(2, 102),
(3, 103),
(4, 104),
(5, 105);

INSERT INTO Takes (SkierID, InstructorName, Date, Time) VALUES
(1, 'Emily Johnson', '2024-03-01', '09:00:00'),
(2, 'Michael Smith', '2024-03-01', '10:00:00'),
(3, 'Sophia Brown', '2024-03-02', '11:00:00'),
(4, 'Daniel Garcia', '2024-03-02', '12:00:00'),
(5, 'Olivia Martinez', '2024-03-03', '13:00:00');

INSERT INTO Inhabits (SlopeName, Species) VALUES
('Slope 1', 'Elk'),
('Slope 2', 'Mountain Goat'),
('Slope 3', 'Snowshoe Hare'),
('Slope 4', 'Lynx'),
('Slope 5', 'Brown Bear');
```