University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: February 28th, 2023

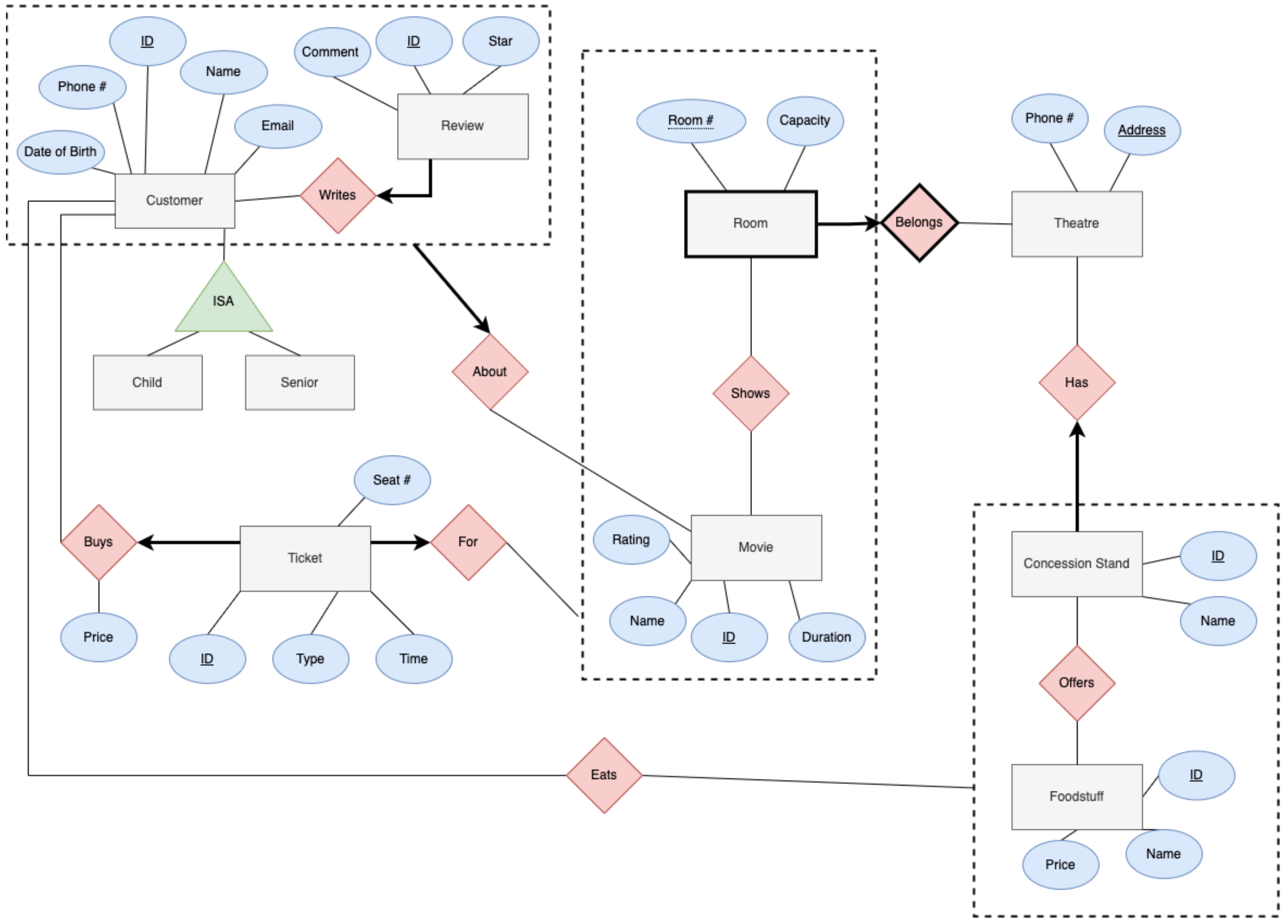
Group Number: 15

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Sean Quan	43496900	r9v5y	seanpquan@gmail.com
Tanmay Goyal	50368463	n7a7f	sancriuse75@gmail.com
Yuhei Arimoto	36561967	k3n3c	yuhei61627@icloud.com

1. Brief Summary

Our project is movie theater software. The customers will be able to buy tickets for movies at different theaters, and they will also be able to write reviews for movies and track what they buy at concession stands. The theater manager can manage which rooms show which movies at which times and see how well their business is running.

2. The ER Diagram



Changes from Milestone 1 ER Diagram:

- Changed aggregation of Room, Movie, Theatre to include only Room and Movie based on the TA’s comment “Cannot use aggregation to treat multiple relationship sets as a single entity” on Milestone 1 ER Diagram
- Added aggregation to Offers relation between Concession Stand and Foodstuff, so that Customer buys foodstuff from concession stand. This was a suggestion given by TA on Milestone 1 ER Diagram.
- Removed the attribute “Guardian Name” for the child in the ISA because it is unnecessary and there is no use
- Moved the Time attribute from Shows to Ticket to simplify the diagram and because the double aggregation was unnecessary.

3. Schema

Note*:*
- Primary keys are underlined. Foreign keys are bolded
- For attribute naming, If there is an identical attribute for multiple entities, add the name of the entity before the attribute. Eg: customer has attribute ID becomes customerID.
- # is replaced with Num or omitted. Eg. room# is roomNum, phone# is phone

Customer(<u>ID</u>: varchar[20], name: varchar[30], email: varchar[40], phone: varchar[15], dateOfBirth: date)

- ck: ID, email, (name + phone + dateOfBirth)
- not null: email
- unique: email, (name + phone + dateOfBirth)

Child(<u>**ID**</u>: varchar[20])

- ck: ID

Senior(<u>**ID**</u>: varchar[20])

- ck: ID

Review(<u>ID</u>: varchar[20], star: float, comment: varchar[1000], **customerID**: varchar[20], **movieID**: varchar[20])

- ck: ID
- not null: movieID, customerID

Theatre(<u>address:</u> varchar[50], phone: varchar[20])

- ck: address, phone
- unique: phone

Room(<u>roomNum</u>: integer, <u>**address**</u>: varchar[50], capacity: integer)

- ck: (roomNum + address)

Movie(<u>ID</u>: varchar[20], duration: integer, rating: varchar[10], name: varchar[50])

- ck: ID
- not null: duration, name

Shows(<u>**address**</u>: varchar[50], <u>**roomNum**</u>: integer, <u>**movieID**</u>: varchar[20])

*Movie's ID is changed to movieID to make clear what ID it is referring to

- ck: address + roomNum + movieID

Ticket(<u>ID</u>: varchar[20], type: varchar[20], seatNum: varchar[4], time: timestamp, **customerID**: varchar[20], price: float, **address**: varchar[50], **roomNum**: integer, **movieID**: varchar[20])

- ck: ID, (time + roomNum + address + seatNum)
- unique: time + roomNum + address + seatNum
- not null: customerID, address, roomNum, movieID, time, price

ConcessionStand(<u>ID:</u> varchar[20], name: varchar[20], **address:** varchar[50])

- ck: ID
- not null: **address**

Foodstuff (<u>ID:</u> varchar[20], name: varchar[20], price: float)

- ck: ID
- not null: name, price

Offers (<u>**foodStuffID:**</u> varchar[20], <u>**concesssionStandID:**</u> varchar[20])

- ck: (foodStuffID + cooncessionStandID)

Eats (<u>**customerID:**</u> varchar[20], <u>**concessionStandID:**</u> varchar[20] , <u>**foodStuffID:**</u> varchar[20])

- ck: customerID + concessionStandID + foodStuffID

4. Functional Dependencies

Customer(ID: varchar[20], name: varchar[30], email: varchar[40], phone: varchar[15], dateOfBirth: date)

- ID -> name, email, phone, dateOfBirth
- email -> ID, name, phone, dateOfBirth
- (name + phone + dateOfBirth) -> ID, email

Child(**ID**: varchar[20])

- Only the trivial one

Senior(**ID**: varchar[20])

- Only the trivial one

Review(ID: varchar[20], star: float, comment: varchar[1000], **customerID**: varchar[20], **movieID**: varchar[20])

- ID -> star, comment, customerID, movieID

Theatre(address: varchar[50], phone: varchar[20])

- address -> phone
- phone -> address

Room(roomNum: integer, **address**: varchar[50], capacity: integer)

- roomNum, address -> capacity

Movie(ID: varchar[20], duration: integer, rating: varchar[10], name: varchar[50])

- ID -> duration, rating, name

Shows(**address**: varchar[50], **roomNum**: integer, **movieID**: varchar[20])

- only the trivial one

Ticket(<u>ID</u>: varchar[20], type: varchar[20], seatNum: varchar[4], time: timestamp, **customerID**: varchar[20], price: float, **address**: varchar[50], **roomNum**: integer, **movieID**: varchar[20])

- ID -> type, seatNUm, time, customerID, price, address, roomNum, movieID
- time, roomNum, address, seatNum -> ID, type, customerID, price, movieID
- time, roomNum, address -> movieID (non-PK or CK)

ConcessionStand(<u>ID:</u> varchar[20], name: varchar[20], **address:** varchar[50])

- ID -> name, address

Foodstuff (<u>ID:</u> varchar[20], name: varchar[20], price: float)

- ID -> name, price

Offers (<u>**foodStuffID:**</u> varchar[20], <u>**concesssionStandID:**</u> varchar[20])

- only the trivial one

Eats (<u>**customerID:**</u> varchar[20], <u>**concessionStandID:**</u> varchar[20] , <u>**foodStuffID:**</u> varchar[20])

- Only the trivial one

5. Normalization

Not in BCNF or 3NF:

Ticket(ID: varchar[20], type: varchar[20], seatNum: varchar[4], time: timestamp, **customerID**: varchar[20], price: float, **address**: varchar[50], **roomNum**: integer, **movieID**: varchar[20])

- ID -> type, seatNum, time, customerID, price, address, roomNum, movieID
    - ID is PK, thus super key; does not violate BCNF
- time, roomNum, address, seatNum -> ID, type, customerID, price, movieID
    - time + roomNum + address + seatNum is CK, thus super key; does not violate BCNF
- time, roomNum, address -> movieID (non-PK or CK)
    - violates BCNF or 3NF
    - time + roomNum + address is not super key & movieID is not part of minimal key

R1(time, **roomNum**, **address**, **movieID**)

R2(ID, type, seatNum, time, **customerID**, price, **address**, **roomNum**)

- ID+ = {type, seatNum, time, customerID, price, address, roomNum}; thus, ID is PK

Let R1 be renamed to MovieSchedule and R2 be renamed to Ticket.

Post Normalization to BCNF:

MovieSchedule(time: timestamp, **roomNum**: integer, **address**: varchar[50], **movieID**: varchar[20])

- ck: time + roomNum + address
- not null: movieID

Ticket(ID: varchar[20], type: varchar[20], seatNum: varchar[4], **time**: timestamp, **customerID**: varchar[20], price: float, **address**: varchar[50], **roomNum**: integer)

- ck: ID, (time + roomNum + address + seatNum)
- unique: (time + roomNum + address + seatNum)
- not null: time, customerID, address, roomNum, price

No decomposition (Tables already in 3NF or BCNF):

Customer(ID: varchar[20], name: varchar[30], email: varchar[40], phone: varchar[15], dateOfBirth: date)

- ck: ID, email, (name + phone + dateOfBirth)
- not null: email
- unique: email, (name + phone + dateOfBirth)

Child(**ID**: varchar[20])

- ck: ID

Senior(**ID**: varchar[20])

- ck: ID

Review(ID: varchar[20], star: float, comment: varchar[1000], **customerID**: varchar[20], **movieID**: varchar[20])

- ck: ID
- not null: movieID, customerID

Theatre(address: varchar[50], phone: varchar[20])

- ck: address, phone
- unique: phone

Room(roomNum: integer, **address**: varchar[50], capacity: integer)

- ck: (roomNum + address)

Movie(ID: varchar[20], duration: integer, rating: varchar[10], name: varchar[50])

- ck: ID
- not null: duration, name

Shows(**<u>address</u>**: varchar[50], **<u>roomNum</u>**: integer, **<u>movieID</u>**: varchar[20])

*Movie’s ID is changed to movieID to make clear what ID it is referring to

- ck: address + roomNum + movieID

ConcessionStand(<u>ID:</u> varchar[20], name: varchar[20], **address:** varchar[50])

- ck: ID
- not null: **address**

Foodstuff (<u>ID:</u> varchar[20], name: varchar[20], price: float)

- ck: ID
- not null: name, price

Offers (**<u>foodStuffID:</u>** varchar[20], **<u>concesssionStandID:</u>** varchar[20])

- ck: (foodStuffID + cooncessionStandID)

Eats (**<u>customerID:</u>** varchar[20], **<u>concessionStandID:</u>** varchar[20] , **<u>foodStuffID:</u>** varchar[20])

- ck: customerID + concessionStandID + foodStuffID

6. SQL DDL Statements

```
CREATE TABLE Customer (
        ID              VARCHAR(20) PRIMARY KEY,
        name            VARCHAR(30),
        email           VARCHAR(40)  UNIQUE NOT NULL,
        phone           VARCHAR(15),
        dateOfBirth     DATE,
        UNIQUE  (name, phone, dateOfBirth)
)

CREATE TABLE Child (
        ID      VARCHAR(20) PRIMARY KEY,
        FOREIGN KEY (ID) REFERENCES Customer(ID)
                ON DELETE CASCADE
)

CREATE TABLE Senior (
        ID      VARCHAR(20) PRIMARY KEY,
        FOREIGN KEY (ID) REFERENCES Customer(ID)
                ON DELETE CASCADE
)

CREATE TABLE Review (
        ID              VARCHAR(20) PRIMARY KEY,
        star            FLOAT,
        comment         VARCHAR(500),
        customerID      VARCHAR(20)  NOT NULL,
        movieID         VARCHAR(20)  NOT NULL,
        FOREIGN KEY (customerID) REFERENCES Customer(ID),
        FOREIGN KEY (movieID) REFERENCES Movie(ID)
)

CREATE TABLE Theatre (
        address         VARCHAR(50) PRIMARY KEY,
        phone           VARCHAR(20) UNIQUE
)
```

```
CREATE TABLE Room (
          roomNum       INTEGER,
          address       VARCHAR(50),
          capacity      INTEGER,
          PRIMARY KEY (roomNum, address),
          FOREIGN KEY (address) REFERENCES Theatre(address)
                    ON DELETE CASCADE
)

CREATE TABLE Movie (
          ID            VARCHAR(20) PRIMARY KEY,
          duration      INTEGER NOT NULL,
          rating        VARCHAR(10),
          name          VARCHAR(50) NOT NULL
)

CREATE TABLE Shows (
          address       VARCHAR(50),
          roomNum       INTEGER,
          movieID       VARCHAR(20),
          PRIMARY KEY (address, roomNum, movieID),
          FOREIGN KEY (roomNum, address) REFERENCES Room(roomNum, address),
          FOREIGN KEY (movieID) REFERENCES Movie(ID)
)

CREATE TABLE MovieSchedule (
          time          TIMESTAMP,
          roomNum       INTEGER,
          address       VARCHAR(50),
          movieID       VARCHAR(20) NOT NULL,
          PRIMARY KEY (time, roomNum, address),
          FOREIGN KEY (address, roomNum, movieID) REFERENCES Shows(address, roomNum,
movieID)
)
```

```
CREATE TABLE Ticket (
	ID			VARCHAR(20)  PRIMARY KEY,
	type			VARCHAR(20),
	seatNum		VARCHAR(4),
	time			TIMESTAMP NOT NULL,
	customerID	VARCHAR(20) NOT NULL,
	price			FLOAT NOT NULL,
	address		VARCHAR(50) NOT NULL,
	roomNum		INTEGER NOT NULL,
	UNIQUE (time, roomNum, address, seatNum),
	FOREIGN KEY (customerID) REFERENCES Customer(ID),
	FOREIGN KEY (time, roomNum, address) REFERENCES MovieSchedule(time, roomNum,
address)
)

CREATE TABLE ConcessionStand (
	ID			VARCHAR(20) PRIMARY KEY,
	name			VARCHAR(20),
	address		VARCHAR(50) NOT NULL,
	FOREIGN KEY (address) REFERENCES Theatre(address)
)

CREATE TABLE Foodstuff (
	ID			VARCHAR(20) PRIMARY KEY,
	name			VARCHAR(20) NOT NULL,
	price			FLOAT NOT NULL
)

CREATE TABLE Offers (
	foodStuffID			VARCHAR(20),
	concessionStandID	VARCHAR(20),
	PRIMARY KEY (foodStuffID, concessionStandID),
	FOREIGN KEY (foodStuffID) REFERENCES Foodstuff(ID),
	FOREIGN KEY (concessionStandID) REFERENCES ConcessionStand(ID)
)

CREATE TABLE Eats (
	customerID			VARCHAR(20),
	concessionStandID	VARCHAR(20),
	foodStuffID			VARCHAR(20),
	PRIMARY KEY (customerID, concessionStandID, foodStuffID),
	FOREIGN KEY (customerID) REFERENCES Customer(ID),
	FOREIGN KEY (foodStuffID, concessionStandID) REFERENCES Offers(foodStuffID,
concessionStandID)
```

)

7. INSERT Statements

Customer(<u>ID</u>: varchar[20], name: varchar[30], email: varchar[40], phone: varchar[15], dateOfBirth: date)

1. INSERT INTO Customer(ID, name, email, phone, dateOfBirth)
   VALUES ('1', 'Yuhei Arimoto', 'yuhei61627@icloud.com', '6043652065', 20021220)
2. INSERT INTO Customer(ID, name, email, phone, dateOfBirth)
   VALUES (‘2’, ‘Sean Quan’, ‘seanpquan@gmail.com’, ‘6044961125’, 20021231)
3. INSERT INTO Customer(ID, name, email, phone, dateOfBirth)
   VALUES (‘3’, ‘Tanmay Goyal’, ‘sancriuse75@gmail.com’, ‘2083891111’, 20030120)
4. INSERT INTO Customer(ID, name, email, phone, dateOfBirth)
   VALUES (‘4’, ‘Jack Aaaa’, ‘jack1234@icloud.com’, ‘6041655235’, 20001005)
5. INSERT INTO Customer(ID, name, email, phone, dateOfBirth)
   VALUES (‘5’, ‘Sam Bcde’, ‘samsam@gmail.com’, ‘1234561234’, 19900610)

Child(**<u>ID</u>**: varchar[20])

1. INSERT INTO Child(ID) VALUES ('1')
2. INSERT INTO Child(ID) VALUES ('2')
3. INSERT INTO Child(ID) VALUES ('10') *assume Customer with ID 10 exists
4. INSERT INTO Child(ID) VALUES ('15') *assume Customer with ID 15 exists
5. INSERT INTO Child(ID) VALUES ('105') *assume Customer with ID 105 exists

Senior(**<u>ID</u>**: varchar[20])

1. INSERT INTO Child(ID) VALUES ('3')
2. INSERT INTO Child(ID) VALUES ('5')
3. INSERT INTO Child(ID) VALUES ('9') *assume Customer with ID 9 exists
4. INSERT INTO Child(ID) VALUES ('82') *assume Customer with ID 82 exists
5. INSERT INTO Child(ID) VALUES ('100') *assume Customer with ID 100 exists

Review(<u>ID</u>: varchar[20], star: float, comment: varchar[1000], **customerID**: varchar[20], **movieID**: varchar[20])

1. INSERT INTO Review(ID, star, comment, customerID, movieID)
   VALUES ('1', 4.0, 'Great movie', '1', '10')
2. INSERT INTO Review(ID, star, comment, customerID, movieID)
   VALUES ('2', 4.5, 'Awesome plot', '2', '10')
3. INSERT INTO Review(ID, star, comment, customerID, movieID)
   VALUES ('3', 1.0, 'Just boring', '5', '2345')
4. INSERT INTO Review(ID, star, comment, customerID, movieID)
   VALUES ('4', 3.2, 'Super Mid movie, not bad', '3', '157982')
5. INSERT INTO Review(ID, star, comment, customerID, movieID)
   VALUES ('10', 4.1, 'Pretty good. Great actors', '4', '113')

Theatre(<u>address:</u> varchar[50], phone: varchar[20])

1. INSERT INTO Theatre(address, phone)
   VALUES ('1234 West Mall, Vancouver, BC', '3251112682')
2. INSERT INTO Theatre(address, phone)
   VALUES ('5959 Student Union Bouldvard, Vancouver, BC', '6041112222')
3. INSERT INTO Theatre(address, phone)
   VALUES ('2929 Main Mall, Victoria, BC', '6049871234')
4. INSERT INTO Theatre(address, phone)
   VALUES ('104 58 Ave SE, Calgary, AB', '4032555501')
5. INSERT INTO Theatre(address, phone)
   VALUES ('452 SW Marine Dr, Vancouver, BC', '6046300414')

Room(roomNum: integer, **address**: varchar[50], capacity: integer)

1. INSERT INTO Room(roomNum, address, capacity)
   VALUES (1, '1234 West Mall, Vancouver, BC', 100)
2. INSERT INTO Room(roomNum, address, capacity)
   VALUES (2, '1234 West Mall, Vancouver, BC', 150)
3. INSERT INTO Room(roomNum, address, capacity)
   VALUES (1, '452 SW Marine Dr, Vancouver, BC', 85)
4. INSERT INTO Room(roomNum, address, capacity)
   VALUES (4, '452 SW Marine Dr, Vancouver, BC', 50)
5. INSERT INTO Room(roomNum, address, capacity)
   VALUES (5, '452 SW Marine Dr, Vancouver, BC', 200)

Movie(ID: varchar[20], duration: integer, rating: varchar[10], name: varchar[50])

1. INSERT INTO Movie(ID, duration, rating, name)
   VALUES ('10', 148 , 'PG-13' , 'Spider-Man: No Way Home')
2. INSERT INTO Movie(ID, duration, rating, name)
   VALUES ('53', 121 , 'PG-13' , 'Spider-Man')
3. INSERT INTO Movie(ID, duration, rating, name)
   VALUES ('2345', 95 , '18A' , 'Cocaine Bear')
4. INSERT INTO Movie(ID, duration, rating, name)
   VALUES ('157982', 126 , 'G' , 'The Karate Kid')
5. INSERT INTO Movie(ID, duration, rating, name)
   VALUES ('113', 125 , 'PG-13' , 'Ant-Man and the Wasp: Quantumania')

Shows(**address**: varchar[50], **roomNum**: integer, **movieID**: varchar[20])

1. INSERT INTO Shows(address, roomNum, movieID)
   VALUES ('1234 West Mall, Vancouver, BC', 1, '10')
2. INSERT INTO Shows(address, roomNum, movieID)
   VALUES ('1234 West Mall, Vancouver, BC', 1, '2345')

3. INSERT INTO Shows(address, roomNum, movieID)
   VALUES ('452 SW Marine Dr, Vancouver, BC', 4, '10')
4. INSERT INTO Shows(address, roomNum, movieID)
   VALUES ('452 SW Marine Dr, Vancouver, BC', 4, '157982')
5. INSERT INTO Shows(address, roomNum, movieID)
   VALUES ('452 SW Marine Dr, Vancouver, BC', 4, '113')

MovieSchedule(<u>time</u>: timestamp, <u>**roomNum**</u>: integer, <u>**address**</u>: varchar[50], **movieID**: varchar[20])

1. INSERT INTO MovieSchedule(time, roomNum, address, movieID)
   VALUES ('2023-4-1 09:00', 4, '452 SW Marine Dr, Vancouver, BC', '10')
2. INSERT INTO MovieSchedule(time, roomNum, address, movieID)
   VALUES ('2023-4-1 15:00', 4, '452 SW Marine Dr, Vancouver, BC', '10')
3. INSERT INTO MovieSchedule(time, roomNum, address, movieID)
   VALUES ('2023-4-10 13:00', 4, '452 SW Marine Dr, Vancouver, BC', '10')
4. INSERT INTO MovieSchedule(time, roomNum, address, movieID)
   VALUES ('2023-4-1 10:00', 4, '452 SW Marine Dr, Vancouver, BC', '157982')
5. INSERT INTO MovieSchedule(time, roomNum, address, movieID)
   VALUES ('2023-4-1 09:00', 1, '1234 West Mall, Vancouver, BC', '10')

Ticket(<u>ID</u>: varchar[20], type: varchar[20], seatNum: varchar[4], **time**: timestamp, **customerID**: varchar[20], price: float, **address**: varchar[50], **roomNum**: integer)

1. INSERT INTO Ticket(ID, type, seatNum, time, customerID, price, address, roomNum)
   VALUES ('1452', '3D', 'D15', '2023-4-1 09:00', '1', 19.0, '452 SW Marine Dr, Vancouver, BC', 4)
2. INSERT INTO Ticket(ID, type, seatNum, time, customerID, price, address, roomNum)
   VALUES ('1453', '3D Luxury', 'A05', '2023-4-1 09:00', '2', 20.0, '1234 West Mall, Vancouver, BC', 1)
3. INSERT INTO Ticket(ID, type, seatNum, time, customerID, price, address, roomNum)

VALUES ('1500', '4DX', 'C05', '2023-4-1 10:00', '4', 25.0, '452 SW Marine Dr, Vancouver, BC', 4)

4. INSERT INTO Ticket(ID, type, seatNum, time, customerID, price, address, roomNum)
   VALUES ('173', 'Normal', 'C05', '2023-4-10 13:00', '1', 17.0, '452 SW Marine Dr, Vancouver, BC', 4)
5. INSERT INTO Ticket(ID, type, seatNum, time, customerID, price, address, roomNum)
   VALUES ('782', 'Normal', 'D21', '2023-4-1 15:00', '5', 17.0, '452 SW Marine Dr, Vancouver, BC', 4)

ConcessionStand(ID: varchar[20], name: varchar[20], **address:** varchar[50])

1. INSERT INTO ConcessionStand(ID, name, address)
   VALUES ('12', 'McDonald', '452 SW Marine Dr, Vancouver, BC')
2. INSERT INTO ConcessionStand(ID, name, address)
   VALUES ('115', 'StarBucks', '452 SW Marine Dr, Vancouver, BC')
3. INSERT INTO ConcessionStand(ID, name, address)
   VALUES ('1', 'A&W', '1234 West Mall, Vancouver, BC')
4. INSERT INTO ConcessionStand(ID, name, address)
   VALUES ('1234', 'StarBucks', '1234 West Mall, Vancouver, BC')
5. INSERT INTO ConcessionStand(ID, name, address)
   VALUES ('602', 'Tim Hortons', '104 58 Ave SE, Calgary, AB')

Foodstuff (ID: varchar[20], name: varchar[20], price: float)

1. INSERT INTO Foodstuff(ID, name, price)
   VALUES ('100', 'BigMac Meal', 12.0)
2. INSERT INTO Foodstuff(ID, name, price)
   VALUES ('101', 'Fries', 3.5)
3. INSERT INTO Foodstuff(ID, name, price)
   VALUES ('2001', 'Iced Latte', 5.5)
4. INSERT INTO Foodstuff(ID, name, price)

VALUES ('57', 'Wedges', 4.5)

5. INSERT INTO Foodstuff(ID, name, price)

   VALUES ('3', 'Root Beer', 2.5)

Offers (**foodStuffID:** varchar[20], **concesssionStandID:** varchar[20])

1. INSERT INTO Offers(foodStuffID, concessionStandID)

   VALUES ('100', '12')
2. INSERT INTO Offers(foodStuffID, concessionStandID)

   VALUES ('2001', '1234')
3. INSERT INTO Offers(foodStuffID, concessionStandID)

   VALUES ('2001', '115')
4. INSERT INTO Offers(foodStuffID, concessionStandID)

   VALUES ('3', '1')
5. INSERT INTO Offers(foodStuffID, concessionStandID)

   VALUES ('57', '602')

Eats (**customerID:** varchar[20], **concessionStandID:** varchar[20] , **foodStuffID:** varchar[20])

1. INSERT INTO Eats(customerID, concessionStandID, foodStuffID)

   VALUES ('1', '12', '100')
2. INSERT INTO Eats(customerID, concessionStandID, foodStuffID)

   VALUES ('2', '115', '2001')
3. INSERT INTO Eats(customerID, concessionStandID, foodStuffID)

   VALUES ('2', '12', '100')
4. INSERT INTO Eats(customerID, concessionStandID, foodStuffID)

   VALUES ('3', '602', '57')
5. INSERT INTO Eats(customerID, concessionStandID, foodStuffID)

   VALUES ('4', '1234', '2001')