University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: 2023/02/28

Group Number: 46

Name	**Student Number**	**CS Alias (Userid)**	**Preferred E-mail Address**
Justin Li			contactJustinLi@gmail.com
Zach Chernenko			zach@chernenko.com
Anthony Hayek			anthony382@hotmail.com

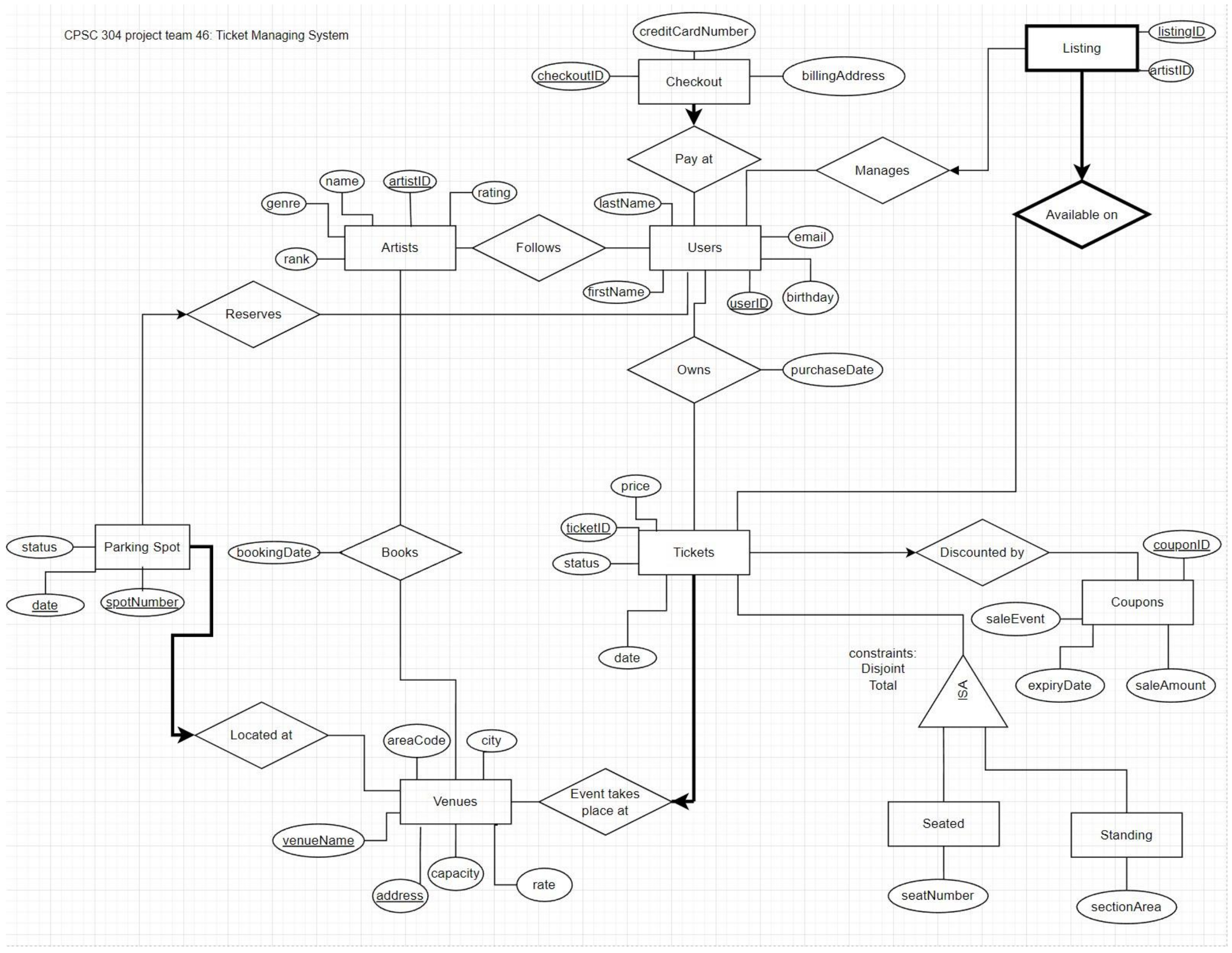
By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address,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

Step 2 - Description

Our project is a Ticket Managing System, similar to Ticketmaster. Users can buy and sell tickets, reserve parking, and follow their favourite artists. Additionally, tickets can be discounted with coupons and artists can book venues.

Step 3 - ER Diagram



Changes made to ER diagram from milestone 1 Parking Spot

- Changed lotNumber under Parking Spot to spotNumber ● Added date attributes under Parking Spot. Reserves
- Removed id
- Change the relationship to one (user) to many (parkingSpot) so now user can reserve same spot on different day

Books

- Changed date to bookingDate under Books

Artist

- Removed ISA under artist because it is unnecessary
- Added rank attribute

Listing

- Changed artist to artistID Tickets
- Removed barcodeNum because it is unnecessary
- Added date

Coupons

- Change endDate to expiryDate
- Added couponID and make it as primary key instead of saleAmount
- Added saleEvent

Venues

- Added areaCode and City

Step 4 - Schema (Underline = PK, Bold = FK, CKs are Unique)

- Users(userID: INTEGER, firstName: CHAR(30), lastName: CHAR(30), email: CHAR(50), birthday: DATE) (email needs to be unique)
- CheckoutAndPayAt(checkoutID: INTEGER, creditCardNumber: INTEGER, billingAddress: CHAR(30), **userID**: INTEGER) (userID cannot be null)
- Artists(artistID: INTEGER, name: CHAR(30), rating: INTEGER, genre: CHAR(30), rank: CHAR(1))
- Follows(**artistID**: INTEGER, **userID**: INTEGER)
- ParkingSpotLocatedAtAndReserved(spotNumber: INTEGER, date: DATE, status: char(20), **venueName:** char(30)**, address:** char(30)**, userID:** char(30)) (name and address cannot be null)
- Venues(venueName: char(30), address: char(40), capacity: INTEGER, rate: INTEGER, city: char(30), areaCode: char(30))
- Books(**artistID**: INTEGER, **venueName**: char(30), **address**: char(30), bookingDate: DATE)
- Owns(**userID**: INTEGER, **ticketID**: INTEGER, purchaseDate: DATE)
- Seated(ticketID: INTEGER, price: INTEGER, status: char(30), date: DATE, seatNumber: INTEGER, **couponID**: INTEGER, **venueName**: char(30), **address**: char(30)) (venueName and address cannot be null)
- Standing(ticketID: INTEGER, price: INTEGER, status: char(30), date: DATE, sectionArea: char(30), **couponID**: INTEGER, **venueName**: char(30), **address**: char(30)) (venueName and address cannot be null)
- Coupons(couponID: INTEGER, expiryDate: DATE, saleAmount: INTEGER, saleEvent: char(30))
- ManagesAndListingAndAvailbleOn(**ticketID**: INTEGER, listingID: INTEGER, artistID: INTEGER , **userID**: INTEGER)

Step 5 - Functional Dependencies

Users:

- userID -> firstName, lastName, email, birthday
- email -> firstName, lastName, userID, birthday **CheckoutAndPayAt:**
- checkoutID - > creditCardNumber, userID, billingAddress
- creditCardNumber -> billingAddress **Artists:**
- artistID - > name, rating, genre
- name -> genre
- rating -> rank

ParkingSpotLocatedAtAndReserved:

- spotNumber, Date -> status, venueName, address, userID, city

Venues

- venueName, address -> capacity, rate, city, areaCode
- address -> city
- city -> areaCode
- venueName -> capacity
- capacity -> rate

Books

- artistID, venueName, address -> date

Owns

- userID, ticketID - > purchaseDate

Seated

- ticketID - > price, status, date, seatNumber, couponID, venueName, address

Standing

- ticketID - > price, status, date, sectionArea, couponID, venueName, address

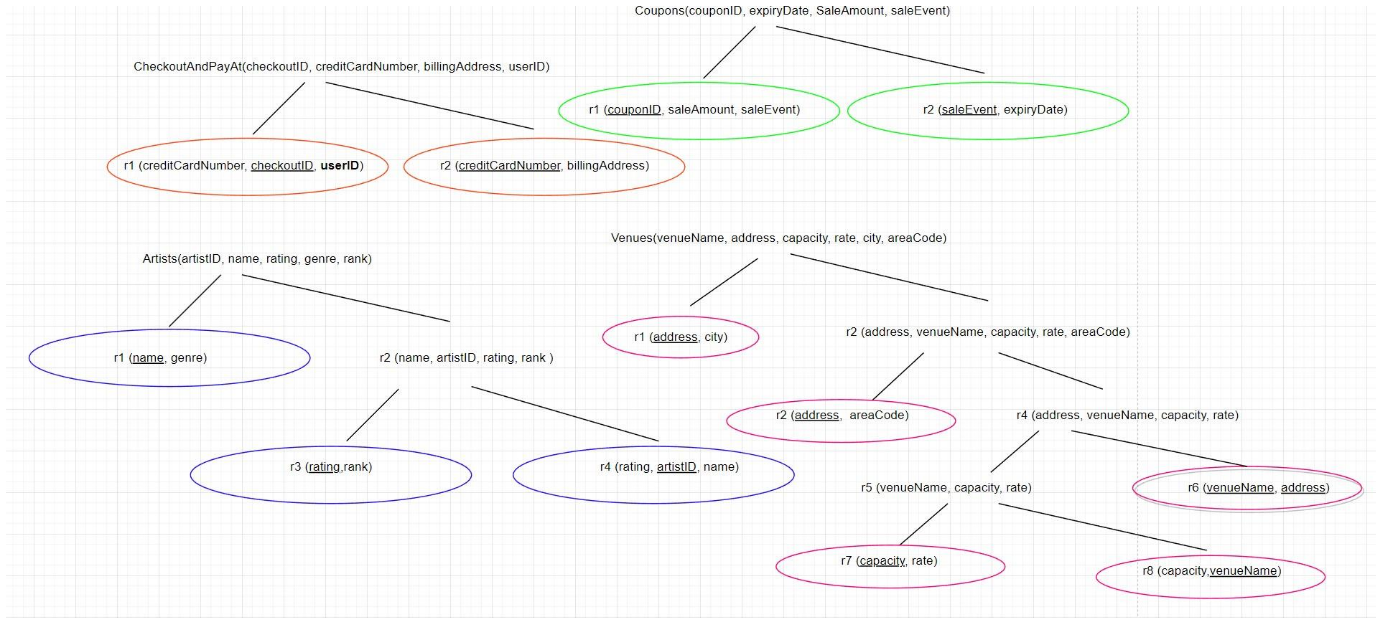
Coupons

- couponID - > expiryDate, saleAmount, saleEvent
- saleEvent -> expiryDate

ManagesAndListingAndAvailbleOn

- ticketID - > listingID, artistID, userID

Step 6 - Normalization


Coupons(couponID, expiryDate, SaleAmount, saleEvent)
r1 (couponID, saleAmount, saleEvent)
r2 (saleEvent, expiryDate)
CheckoutAndPayAt(checkoutID, creditCardNumber, billingAddress, userID)
r1 (creditCardNumber, checkoutID, userID)
r2 (creditCardNumber, billingAddress)
Artists(artistID, name, rating, genre, rank)
r1 (name, genre)
r2 (name, artistID, rating, rank)
r3 (rating,rank)
r4 (rating, artistID, name)
Venues(venueName, address, capacity, rate, city, areaCode)
r1 (address, city)
r2 (address, venueName, capacity, rate, areaCode)
r2 (address, areaCode)
r4 (address, venueName, capacity, rate)
r5 (venueName, capacity, rate)
r6 (venueName, address)
r7 (capacity, rate)
r8 (capacity,venueName)

- Users(<u>userID</u>: INTEGER, firstName: CHAR(30), lastName: CHAR(30), email: CHAR(50), birthday: DATE) (email needs to be unique)

- CheckoutAndPayAt(checkoutID: INTEGER, creditCardNumber: INTEGER, **userID**: INTEGER) (userID cannot be null)
- CheckoutAndPayAt2(creditCardnumber: INTEGER, billingAddress: CHAR(30))
- Artists1(name: CHAR(30), genre: CHAR(30))
- Artists2(rating: INTEGER, rank CHAR(1))
- Artists3(rating: INTEGER, artistID: INTEGER, name: CHAR(30))
- Follows(**artistID**: INTEGER, **userID**: INTEGER)
- ParkingSpotLocatedAtAndReserved(spotNumber: INTEGER, date: DATE, status: char(20), **venueName:** char(30)**, address:** char(30)**, userID:** char(30)) (name and address cannot be null)
- Venues1(address: char(40), city: char(30))
- Venues2(address: char(40), areaCode: char(30))
- Venues3(venueName: char(30), address: char(40))
- Venues4(capacity: INTEGER, rate: INTEGER)
- Venues5(venueName: char(30),capacity: INTEGER)
- Books(**artistID**: INTEGER, **venueName**: char(30), **address**: char(30), bookingDate: DATE)
- Owns(**userID**: INTEGER, **ticketID**: INTEGER, purchaseDate: DATE)
- Seated(ticketID: INTEGER, price: INTEGER, status: char(30), date: DATE, seatNumber: INTEGER, **couponID**: INTEGER, **venueName**: char(30), **address**: char(30)) (venueName and address cannot be null)
- Standing(ticketID: INTEGER, price: INTEGER, status: char(30), date: DATE, sectionArea: char(30), **couponID**: INTEGER, **venueName**: char(30), **address**: char(30)) (venueName and address cannot be null)
- Coupons1(couponID: INTEGER, saleAmount: INTEGER, saleEvent: CHAR(30))
- Coupons2(saleEvent: CHAR(30), expiryDate: DATE)
- ManagesAndListingAndAvailbleOn(**ticketID**: INTEGER, listingID: INTEGER, artistID: INTEGER , **userID**: INTEGER)

Step 7 - SQL DDL Statements

```
CREATE TABLE Users(userID INTEGER,
                    firstName CHAR(30),
                    lastName CHAR(30), email
                    CHAR(50), birthday DATE,
                    PRIMARY KEY (userID)
                    UNIQUE (email))
CREATE TABLE CheckoutAndPayAt1(checkoutID INTEGER,
                    creditCardNumber INTEGER, userID
                    INTEGER NOT NULL,
                      PRIMARY KEY (checkoutID),
                    FOREIGN KEY (userID) REFERENCES Users
                   ON DELETE CASCADE ON UPDATE CASCADE
CREATE TABLE CheckoutAndPayAt2(creditCardNumber INTEGER,
       billingAddress CHAR(30), PRIMARY KEY (creditCardNumber))
CREATE TABLE Artists1(name CHAR(30),
                    genre CHAR(30),
                    PRIMARY KEY (name))
CREATE TABLE Artists2(rating INTEGER,
                    rank CHAR(1),
                    PRIMARY KEY (rating))
CREATE TABLE Artists3(rating INTEGER,
                    artistID INTEGER, name
                    CHAR(30),
                      PRIMARY KEY (artistID))
CREATE TABLE Follows(artistID INTEGER,
                    userID INTEGER,
                      PRIMARY KEY (artistID, userID))
CREATE TABLE ParkingSpotLocatedAtAndReserved(spotNumber INTEGER,
                    date DATE, status
                    CHAR(20),
                    venueName
                    CHAR(30) NOT
                    NULL, address
                    CHAR(30) NOT
```

```
                              NULL, userID
                              CHAR(30),
                                 PRIMARY KEY (spotNumber, date),
                              FOREIGN KEY (venueName, address) REFERENCES Venues
  ON DELETE CASCADE ON UPDATE CASCADE, FOREIGN KEY (userID) REFERENCES Users
                            ON DELETE CASCADE ON UPDATE CASCADE)
CREATE TABLE Venues1(address CHAR(40),
                              City char(30),
                              PRIMARY KEY (address))
CREATE TABLE Venues2(address CHAR(40),
                              areaCode CHAR(30),
                              PRIMARY KEY (address))
CREATE TABLE Venues3(venueName char(30),
                              address CHAR(40),
                                 PRIMARY KEY (venueName, address))
CREATE TABLE Venues4(capacity INTEGER,
                              rate INTEGER,
                              PRIMARY KEY (capacity))
CREATE TABLE Venues5(capacity INTEGER,
                              venueName CHAR(30),
                              PRIMARY KEY (venueName))
CREATE TABLE Books(artistID INTEGER,
                              venueName CHAR(30),
                              address CHAR(30), date
                              DATE,
                              PRIMARY KEY (artistID, venueName, address))
CREATE TABLE Owns(userID INTEGER,
                              ticketID INTEGER, purchaseDATE
                              DATE,
                                 PRIMARY KEY (userID, ticketID))
CREATE TABLE Seated(ticketID INTEGER,
                              price INTEGER, status CHAR(30), date
                              DATE, seatNumber INTEGER,
                              couponID INTEGER, venueName
                              CHAR(30) NOT NULL, address
                              CHAR(30) NOT NULL,
                              PRIMARY KEY (ticketID)
                              FOREIGN KEY (venueName, address) REFERENCES VENUES
                            ON DELETE CASCADE ON UPDATE CASCADE
                              FOREIGN KEY (couponID) REFERENCES Coupons
                            ON DELETE CASCADE ON UPDATE CASCADE)
CREATE TABLE Standing(ticketID INTEGER,
                              price INTEGER, status CHAR(30), date
                              DATE, sectionArea CHAR(30),
                              couponID INTEGER, venueName
                              CHAR(30) NOT NULL, address
                              CHAR(30) NOT NULL,
                              PRIMARY KEY (ticketID)
                              FOREIGN KEY (venueName, address) REFERENCES VENUES
                            ON DELETE CASCADE ON UPDATE CASCADE
                              FOREIGN KEY (couponID) REFERENCES Coupons
                            ON DELETE CASCADE ON UPDATE CASCADE)
CREATE TABLE Coupons1(couponID INTEGER,
                              saleAmount INTEGER, saleEvent
                              CHAR(30),
                              PRIMARY KEY (couponID))
CREATE TABLE Coupons2(saleEvent, CHAR(30),
                              expiryDate DATE,
                                PRIMARY KEY (saleEvent))
CREATE TABLE ManagesAndListingAndAvailbleOn(ticketID INTEGER,
                              listingID INTEGER, artistID
                              INTEGER, userID INTEGER,
                              PRIMARY KEY (ticketID, listingID),
                              FOREIGN KEY (userID) REFERENCES Users
                            ON DELETE CASCADE ON UPDATE CASCADE)
                              FOREIGN KEY (ticketID) REFERENCES Tickets
                            ON DELETE CASCADE ON UPDATE CASCADE)
```

Step 8 - INSERT Statements

Users:

```
INSERT INTO Users(userID, firstName, lastName, email, birthday ) VALUES
        (789, ‘John’, ‘Smith’, ‘johnsmith98@gmail.com’, ‘1988-07-14’),
        (288, ‘Paul’, ‘Johnson, ‘pjohnson@gmail.com’, ‘1999-04-18’),
        (728, ‘Mike’, ‘Rogan’, ‘miker93@gmail.com’, ‘1993-02-20’),
        (292, ‘Samantha’, ‘Miller’, ‘SamanthaMill@gmail.com’, ‘2002-08-10’),
        (237, ‘Alexis’, ‘Garcia’, ‘agarcia17@gmail.com’, ‘2000-03-16’)
```

CheckoutAndPayAt1:

```
INSERT INTO CheckoutAndPay1(checkoutID, creditCardNumber, userID ) VALUES
        (10008954, 4566345623452837,  789),
        (10003802, 5566245634952934, 288),
        (10029302, 28939462345830928, 728),
        (10294020, 3394302034229876, 292),
        (13839202, 2493930222234567, 237)
```

CheckoutAndPay2:

```
INSERT INTO CheckoutAndPay2( creditCardNumber, billingAddress ) VALUES
        (2493930222234567, ‘1456 Summer st’),
        (3394302034229876,  ‘1292 Dover st’),
        (292920022929229228,  ‘3212 Hudson st’),
        (392922567959229765,  ‘4613 Timber st’),
        (456712862929222364,  ‘3489 Wright st’)
```

Artists1:

```
INSERT INTO Artists1(name, genre) VALUES
        (‘The Weeknd’, ‘R&B’),
        (‘Drake’, ‘Hip Hop’),
        (‘J. Cole’, ‘Hip Hop’),
        (‘Kendrick Lamar’, ‘Hip Hop’),
        (‘Luke Combs’, ‘Country’),
        (‘Skrillex’, ‘EDM’)
```

Artists2:

```
INSERT INTO Artists2(rating, rank) VALUES
```

```
            (10, ‘S’),
            (9, ‘A’),
            (8, ‘B’), (7,
            ‘C’),
            (6, ‘D’),
            (5, ‘E’),
            (4, ‘F’)
```

Artists3:

```
INSERT INTO Artists3(rating, artistID, name) VALUES
            (10, 189, ‘J. Cole’)
            (10, 101, ‘Kendrick Lamar’),
            (9, 167, ‘Drake’),
            (6, 192, ‘Skrillex’),
            (8, 398, ‘Luke Combs’),
```

Follows:

```
INSERT INTO Follows(artistID, userID) VALUES
            (192, 789),
            (398, 789),
            (167, 789),
            (189, 288),
            (192, 288),
            (101, 292),
            (167, 292);
```

ParkingSpotLocatedAtAndReserved:

```
INSERT INTO ParkingSpotLocatedAtAndReserved(spotNumber, date, status , venueName, address,
userID) VALUES
            (8, ‘2023-06-18’, ‘Reserved’, ‘Rogers Arena’),
            (8, ‘2023-07-19’, ‘Reserved’, ‘Rogers Arena’),
            (9, ‘2023-08-01’, ‘Reserved’, ‘Rogers Arena’),
            (11, ’2024-03-18’, ‘Reserved’, ‘BC Place’),
            (12, ‘2024-03-18’, ‘Reserved’, ‘BC Place’),
            (26, ‘2024-03-18’, ‘Reserved’, ‘BC Place’),
            (8, ‘2024-03-18’, ‘Reserved’, ‘BC Place’),
            (15, ‘2023-08-28’, ‘Reserved’, ‘UBC Thunder Arena’),
            (18, ‘2023-08-28’, ‘Reserved’, ‘UBC Thunder Arena’);
```

Venues:

```
INSERT INTO Venues1(address, city) VALUES
        (‘800 Griffiths Way’, ‘Vancouver’),
        (‘6133 University Blvd’, ‘Vancouver’),
        (‘630 Hamilton St’, ‘Vancouver’),
        (‘868 Granville St’, ‘Vancouver’),
        (‘6066 Thunderbird Blvd’, ‘Vancouver’)

INSERT INTO Venues2(address, areaCode) VALUES
        (‘800 Griffiths Way’, 604),
        (‘6133 University Blvd’, 604),
        (‘630 Hamilton St’, 604),
        (‘868 Granville St’, 604),
        (‘6066 Thunderbird Blvd’, 604)

INSERT INTO Venues3(venueName, address) VALUES (‘Rogers
        Arena’, ‘800 Griffiths Way’),
        (‘AMS Student Nest’, ‘6133 University Blvd’),
        (‘Queen Elizabeth Theatre’, ‘630 Hamilton St’),
        (‘The Commodore Ballroom’, ‘868 Granville St’),
        (‘UBC Thunder Arena’, ‘6066 Thunderbird Blvd’)
INSERT INTO Venues4(capacity, rate) VALUES
        (3000, 10000),
        (50, 500),
        (1000, 3000),
        (500, 5000),
        (100, 7000)

INSERT INTO Venues5(capacity, venueName) VALUES
        (3000, ‘Rogers Arena’),
        (50, ‘AMS Student Nest’),
        (1000,‘Queen Elizabeth Theatre’),
        (500, ‘The Commodore Ballroom’),
        (100, ‘UBC Thunder Arena’)
```

Books:

```
INSERT INTO Books(artistID, venueName, address, date) VALUES
            (192, ‘Rogers Arena’, ‘800 Griffiths Way’ , ‘2023-06-18’),
            (167, ‘Rogers Arena’, ‘800 Griffiths Way’ , ‘2023-07-19’),
            (167, ‘Rogers Arena’, ‘800 Griffiths Way’ , ‘2024-03-18’),
            (101, ‘UBC Thunderbird Arena’, ‘6066 Thunderbird Blvd’, ‘2023-08-28’),
            (398, ‘UBC Thunderbird Arena’, ‘6066 Thunderbird Blvd’, ‘2023-08-29’);
```

Owns:

```
INSERT INTO Owns(userID, ticketID, purchaseDATE) VALUES
            (789, 1, ‘2022-06-09’),
            (789, 2, ‘2022-06-09’),
            (288, 3, ‘2022-04-22’),
            (728, 6, ‘2022-09-19’),
            (292, 7, ‘2023-08-23’);
```

Coupons:

```
INSERT INTO Coupons1(couponID, saleAmount, saleEvent) VALUES
            (1, 80,’christmas’),
            (2, 70, ‘boxing day’),
            (3, 50, ‘winter’),
            (4, 20, ‘summer’),
            (5, 30, ‘new year’)

INSERT INTO Coupons2(saleEvent, expiryDate) VALUES
            (’christmas’, ‘2023-12-31’),
            (‘boxing day’, ‘2023-12-31’),
            (‘winter’, ‘2023-03-31’),
            (‘summer’, ‘2023-06-31’),
            (‘new year’, ‘2023-02-10’)
```

Seated:

```
INSERT INTO Seated(ticketID, price, status, date, seatNumber, couponID, venueName, address) VALUES
            (1, 300, ‘Sold’, ‘2023-06-18’, 1, 3, ‘Rogers Arena’, ‘800 Griffiths Way’),
            (2, 50, ‘Available’, ‘2023-07-19’, 20, 3, ‘AMS Student Nest’, ‘6133 University Blvd’),
```

(3, 150, ‘Available’, ‘2024-03-18’, 45,2, ‘Queen Elizabeth Theatre’, 630 Hamilton St’),

(4, 400, ‘Sold’, ‘2023-08-28’, 65, 4, ‘The Commodore Ballroom’, ‘868 Granville St’), (5, 30, ‘Sold’, ‘2023-08-29’, 52, 5, ‘UBC Thunder Arena’, ‘6066 Thunderbird Blvd’)

Standing:

INSERT INTO Standing(ticketID, price, status, date, sectionArea, couponID, venueName, address) VALUES

(6, 300, ‘Sold’, ‘2023-06-18’, ‘A’, 3, ‘Rogers Arena’, ‘800 Griffiths Way’),

(7, 50, ‘Available’, ‘2023-07-19’, ‘B’, 3, ‘AMS Student Nest’, ‘6133 University Blvd’),

(8, 150, ‘Available’, ‘2024-03-18’, ‘G’,2, ‘Queen Elizabeth Theatre’, 630 Hamilton St’),

(9, 400, ‘Sold’, ‘2023-08-28’, ’E’, 4, ‘The Commodore Ballroom’, ‘868 Granville St’),

(10, 30, ‘Sold’, ‘2023-08-29’, ‘C’, 5, ‘UBC Thunder Arena’, ‘6066 Thunderbird Blvd’)

ManagesAndListingAndAvailbleOn:

INSERT INTO ManagesAndListingAndAvailbleOn(ticketID, listingID, artistID, userID) VALUES

(1,1,189,789),

(3,2,167,288),

(4,3,192,728),

(2,4,398,728),

(5,5,101,237)