**University of British Columbia, Vancouver**
Department of Computer Science

# CPSC 304 Project Cover Page

Milestone #: 2

Date: October 16

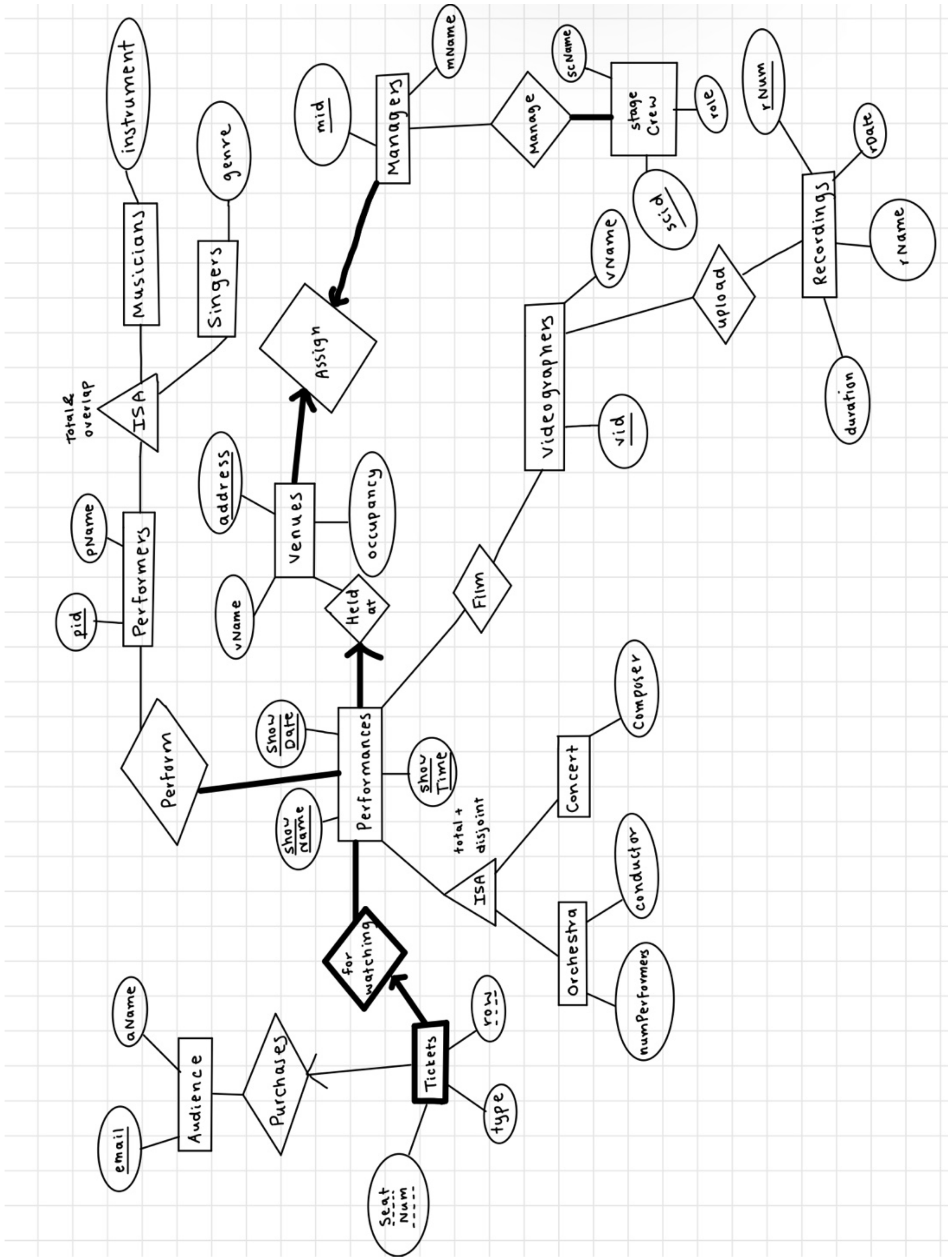
Group Number: 100

| Name | Student Number | CS Alias (Userid) | Preferred E-mail Address |
|---|---|---|---|
| Hannah Jeong | 14094205 | e9x2b | hannahj5678@gmail.com |
| Aaron Cui | 94952777 | w4k5a | aaroncui17@gmail.com |
| Tony Wang | 31550379 | c9r7a | wangtony2003@gmail.com |

2. A brief (~2-3 sentences) summary of your project.
- Our project is an application that will help with event management. Our application specifically manages venues that hold concerts and orchestras. Our application also tracks tickets and audience members associated with them, as well as performance recordings and stage crew members

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.
- Some changes we made according to the Milestone 1 feedback:
    - Adding a primary key to the Recordings entity
    - Adding another entity: Stage Crew Members
    - Adding another relationship with Managers where "Managers-manage-stageCrewMembers"
    - Adding non-key attributes on children entities of ISA: Orchestra (NumOfPerformers, Conductor), Concert (Composer), Musicians (Instruments), Singers (Genres)
- Additional changes:
    - Changed the names of attributes so that they are distinct and succinct
    - Changed the name of ManagedBy to Assign
    - Changed the relationship between Audience-Purchases-Tickets from one-to-one to one-to-many
    - Changed the relationship between Venues-Assign-Managers from many-to-many to one-to-one
    - Removed participation constraints on Performances-Film, Perform-Performers and Uploads-Recording to minimize cases where we need assertions later on
    - Changed Performances ISA from partial + disjoint to total + disjoint

**University of British Columbia, Vancouver**
Department of Computer Science



## 4. The schema derived from your ER diagram.

Recordings (rNum: integer, duration: integer, rName: varchar, rDate: DATE)
- PK: rNum
- Not null: Duration, rName
- CK, Unique: rName

Videographers (vid: integer, vName: varchar)
- PK: vid
- Not null: vName

Upload (rNum: integer, vid: integer)
- PK: rNum, vid
- FK: rNum, vid

Performances (sName: varchar, sDate: DATE, sTime: integer, address: varchar, numPerformers: integer, conductor: varchar, composer: varchar)
- PK: sName, sDate, sTime
- FK: address
- Not null: address

Film (sName: varchar, sDate: DATE, sTime: integer, vid: integer)
- PK: sName, sDate, sTime, vid
- FK: sName, sDate, sTime, vid

Venues (address: varchar, vName: varchar, occupancy: integer, mid: integer)
- PK: address
- FK: mid
- Not null: vName, mid
- CK, Unique: mid

Managers (mid: integer, mName: varchar, address: varchar)
- PK: mid
- FK: address
- Not null: mName, address
- CK, Unique: address

StageCrew (scid: integer, scName: varchar, role: varchar)
- PK: scid
- Not null: scName

Manage (mid: integer, scid: integer)
- PK: mid, scid

- FK: mid, scid

Tickets (seatNum: integer, row: char[2], type: varchar, sName: varchar, sDate: DATE, sTime: integer, email: varchar)
- PK: seatNum, row, sName, sDate, sTime
- FK: sName, sDate, sTime, email

Audience (email: varchar, aName: varchar)
- PK: email
- Not null: aName

Performers (pid: integer, pName: varchar, instrument: varchar, genre: varchar)
- PK: pid
- Not null: pName

Perform (pid: integer, sName: varchar, sDate: DATE, sTime: integer)
- PK: pid, sName, sDate, sTime
- FK: pid, sName, sDate, sTime

**Participation constraint between StageCrew and Manage, Performances and Perform, Performances and Tickets: we will need assertions to cover this and will add them after we have been taught them*

## 5. Functional Dependencies (FDs)
Recordings:
- rNum -> duration, rName, rDate
- rName -> rNum, duration, rDate

Videographers:
- vid -> vName

Performances:
- sName, sDate, sTime -> address, numPerformers, conductor, composer

Venues:
- address -> vName, occupancy, mid
- mid -> address, vName, occupancy

Managers:
- mid -> mName, address
- address -> mid, mName

StageCrew:

- scid -> sName, role

Tickets:
- seatNum, row, sName, sDate, sTime -> email, type

Audience:
- email -> aName

Performers:
- pid -> pName, instruments, genres

** Upload, Film, Manage, Perform all have trivial FDs*

## 6. Normalization

[*Note*: underline for PK, **bolded** for FK]

Tables already in BCNF:
- Videographers (vid: integer, vName: varchar)
- Upload (**rNum**: integer, **vid**: integer)
- Performances (sName: varchar, sDate: DATE, sTime: integer, **address**: varchar, numPerformers: integer, conductor: varchar, composer: varchar)
- Film (**sName**: varchar, **sDate**: DATE, **sTime**: integer, **vid**: integer)
- StageCrew (scid: integer, scName: varchar, role: varchar)
- Manage (**mid**: integer, **scid**: integer)
- Tickets (seatNum: integer, row: char[2], type: varchar, **sName**: varchar, **sDate**: DATE, **sTime**: integer, **email**: varchar)
- Audience (email: varchar, aName: varchar)
- Performers (pid: integer, pName: varchar, instrument: varchar, genre: varchar)
- Perform (**pid**: integer, **sName**: varchar, **sDate**: DATE, **sTime**: integer)

Tables that need decomposition:
- Recordings
- Venues
- Managers

**Recordings**: R(rNum: integer, duration: integer, rName: varchar, rDate: DATE)

Minimal Cover:
- rNum -> duration (Already in BCNF)*
- rNum -> rName (Already in BCNF)*
- rNum -> rDate (Already in BCNF)*

- rName -> rNum
- ~~rName -> duration~~
- ~~rName -> rDate~~

// decompose with rName -> rNum
- R1 (<u>rName</u>: varchar, rNum: integer)
- R2 (<u>duration</u>: integer, <u>rName</u>: varchar, <u>rDate</u>: DATE)

// Final BCNF:
- [R1] RecordingsName (<u>rName</u>: varchar, rNum: integer)
- [R2] RecordingsAttributes(<u>duration</u>: integer, <u>rName</u>: varchar, <u>rDate</u>: DATE)
- [R] Recordings (<u>rNum</u>: integer, duration: integer, rName: varchar, rDate: DATE)
    - *Combined the three tables (*) that were in BCNF before the decomposition into one. Note that combining the tables still leaves the relationship in BCNF.*

**Venues**: V (<u>address</u>: varchar, vName: varchar, occupancy: integer, **mid**: integer)

Minimal Cover:
- address -> vName (already in BCNF)*
- address -> occupancy (already in BCNF)*
- address -> mid (already in BCNF)*
- mid -> address
- ~~mid -> vName~~
- ~~mid -> occupancy~~

// decompose with mid -> address
- V1 (<u>**mid**</u>: integer, address: varchar)
- V2 (<u>vName</u>: varchar, <u>occupancy</u>: integer, <u>**mid**</u>: integer)

// Final BCNF:
- [V1] VenuesMID (<u>**mid**</u>: integer, address: varchar)
- [V2] VenuesAttributes(<u>vName</u>: varchar, <u>occupancy</u>: integer, <u>**mid**</u>: integer)
- [V] Venues (<u>address</u>: varchar, vName: varchar, occupancy: integer, **mid**: integer)
    - *Combined the three tables (*) that were in BCNF before the decomposition into one. Note that combining the tables still leaves the relationship in BCNF.*

**Managers**: M (<u>mid</u>: integer, mName: varchar, **address**: varchar)

Minimal Cover:
- mid -> mName (already in BCNF)
- mid -> address (already in BCNF)
- address -> mid
- ~~address -> mName~~

// decompose with address -> mid

- M1 (**<u>address</u>**: varchar, mid: integer)
- M2 ( <u>mName</u>: varchar, **<u>address</u>**: varchar)

// Final BCNF:

- [M1] ManagersAddress (**<u>address</u>**: varchar, mid: integer)
- [M2] ManagersAttributes (<u>mName</u>: varchar, **<u>address</u>**: varchar)
- Managers (<u>mid</u>: integer, mName: varchar, **address**: varchar)
    - *Combined the three tables that were in BCNF before the decomposition into one. Note that combining the tables still leaves the relationship in BCNF.*

## 7. The SQL DDL statements required to create all the tables from item #6.

```
CREATE TABLE Recordings
        (rNum INTEGER PRIMARY KEY,
        duration INTEGER NOT NULL,
        rName VARCHAR(255) NOT NULL,
        rDate DATE,
        UNIQUE (rName));

CREATE TABLE RecordingsName
        (rName VARCHAR(255) PRIMARY KEY,
        rNum INTEGER NOT NULL,
        UNIQUE (rNum));

CREATE TABLE RecordingsDuration
        (duration INTEGER,
        rName VARCHAR(255),
        rDate DATE,
        PRIMARY KEY (duration, rName, rDate),
        UNIQUE (rName));

CREATE TABLE Videographers
        (vid INTEGER PRIMARY KEY,
        vName varchar(255) NOT NULL);

CREATE TABLE Upload
        (rNum INTEGER,
        vid INTEGER,
        PRIMARY KEY (rNum, vid),
        FOREIGN KEY (rNum) REFERENCES Recordings(rNum)
            ON DELETE CASCADE,
        FOREIGN KEY (vid) REFERENCES Videographers(vid)
            ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar functio*n

```
CREATE TABLE Performances
    (sName VARCHAR(255),
    sDate DATE,
    sTime INTEGER,
    address VARCHAR(255) NOT NULL,
    numPerformers INTEGER,
    conductor VARCHAR(255),
    composer VARCHAR(255),
    PRIMARY KEY (sName, sDate, sTime)
    FOREIGN KEY (address) REFERENCES Venues(address)
       ON DELETE NO ACTION);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE Film
    (sName VARCHAR(255),
    sDate DATE,
    sTime INTEGER,
    vid INTEGER,
    PRIMARY KEY (sName, sDate, sTime, vid),
    FOREIGN KEY (sName, sDate, sTime) REFERENCES Performances(sName, sDate, sTime)
       ON DELETE CASCADE,
    FOREIGN KEY (vid) REFERENCES Videographer(vid)
       ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE Venues
    (address VARCHAR(255) PRIMARY KEY,
    vName VARCHAR(255) NOT NULL,
    occupancy INTEGER,
    mid INTEGER NOT NULL,
    UNIQUE (mid),
    FOREIGN KEY (mid) REFERENCES Managers(mid)
        ON DELETE NO ACTION);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE VenuesMID
        (mid INTEGER PRIMARY KEY,
        address VARCHAR(255) NOT NULL,
        UNIQUE (address),
        FOREIGN KEY (mid) REFERENCES Managers(mid)
                ON DELETE NO ACTION);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE VenuesAttributes
        (vName VARCHAR(255) NOT NULL,
        occupancy INTEGER,
        mid INTEGER PRIMARY KEY,
        FOREIGN KEY (mid) REFERENCES Managers(mid)
                ON DELETE NO ACTION);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE Managers
        (mid INTEGER PRIMARY KEY,
        mName VARCHAR(255) NOT NULL,
        address VARCHAR(255) NOT NULL,
        UNIQUE (address),
        FOREIGN KEY (address) REFERENCES Venues(address)
                ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE ManagersAddress
        (address VARCHAR(255) PRIMARY KEY,
        mid INTEGER NOT NULL,
        UNIQUE (mid),
        FOREIGN KEY (address) REFERENCES Venues(address)
                ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE ManagersAttributes
        (mName VARCHAR(255) NOT NULL,
        address VARCHAR(255) PRIMARY KEY,
        UNIQUE (mName)
```

```
    FOREIGN KEY (address) REFERENCES Venues(address)
        ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE StageCrew
    (scid INTEGER PRIMARY KEY,
    scName VARCHAR(255) NOT NULL,
    role VARCHAR(255));

CREATE TABLE Manage
    (mid INTEGER,
    scid INTEGER,
    PRIMARY KEY (mid, scid),
    FOREIGN KEY (mid) REFERENCES Manager(mid)
        ON DELETE CASCADE,
    FOREIGN KEY (scid) REFERENCES StageCrew(scid)
        ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE Tickets
    (seatNum INTEGER,
    row CHAR[2],
    type VARCHAR(255),
    sName VARCHAR(255),
    sDate DATE,
    sTime INTEGER,
    email VARCHAR(255),
    PRIMARY KEY (seatNum, row, sName, sDate, sTime),
    FOREIGN KEY (sName, sDate, sTime) REFERENCES Performances(sName, sDate, sTime)
        ON DELETE CASCADE,
    FOREIGN KEY (email) REFERENCES Audience(email)
        ON DELETE NO ACTION);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

```
CREATE TABLE Audience
    (email VARCHAR(255) PRIMARY KEY,
    aName VARCHAR(255) NOT NULL);

CREATE TABLE Performers
    (pid INTEGER PRIMARY KEY,
    pName VARCHAR(255) NOT NULL,
    instrument VARCHAR(255),
    genre VARCHAR(255));

CREATE TABLE Perform
    (pid INTEGER,
    sName VARCHAR(255),
    sDate DATE,
    sTime INTEGER,
    PRIMARY KEY (pid, pName, sDate, sTime),
    FOREIGN KEY (sName, sDate, sTime) REFERENCES Performances(sName, sDate, sTime)
          ON DELETE CASCADE,
    FOREIGN KEY (pid) REFERENCES Performers(pid)
          ON DELETE CASCADE);
```

** Oracle doesn't support ON UPDATE CASCADE but this table needs a similar function*

## 8. INSERT statements to populate each table with at least 5 tuples

```
INSERT INTO Recordings VALUES (1, 120, 'Concert in the Park', '2023-10-15');

INSERT INTO Recordings VALUES (2, 90, 'Acoustic Jam', '2023-09-20');

INSERT INTO Recordings VALUES (3, 150, 'Pop Sensation Showcase', '2023-11-05');

INSERT INTO Recordings VALUES (4, 90, 'Classical Crescendo Concert', '2023-09-20');

INSERT INTO Recordings VALUES (5, 150, 'Soulful Melodies Showcase', '2023-11-05');


INSERT INTO Recording1 VALUES ('Concert in the Park', 1);

INSERT INTO RecordingsName VALUES ('Acoustic Jam', 2);

INSERT INTO RecordingsName VALUES (‘Pop Sensation Showcase', 3);

INSERT INTO RecordingsName VALUES ( 'Classical Crescendo Concert', 4);

INSERT INTO RecordingsName VALUES ( 'Soulful Melodies Showcase', 5);


INSERT INTO RecordingsAttributesVALUES (120, 'Concert in the Park', '2023-10-15');
INSERT INTO RecordingsAttributesVALUES (90, 'Acoustic Jam', '2023-09-20');

INSERT INTO RecordingsAttributesVALUES (150, 'Pop Sensation Showcase', '2023-11-05');

INSERT INTO RecordingsAttributesVALUES (90, 'Classical Crescendo Concert', '2023-09-20');

INSERT INTO RecordingsAttributesVALUES (150, 'Soulful Melodies Showcase', '2023-11-05');


INSERT INTO Videographers VALUES (1, 'Amelia Parker’);

INSERT INTO Videographers VALUES (2, ‘Benjamin Smith’);

INSERT INTO Videographers VALUES (3, ‘Chloe Davis’);

INSERT INTO Videographers VALUES (4, ‘Daniel Williams’);

INSERT INTO Videographers VALUES (5, ‘Emily Johnson’);


INSERT INTO Upload VALUES (1, 2);

INSERT INTO Upload VALUES (2, 2);

INSERT INTO Upload VALUES (3, 3);

INSERT INTO Upload VALUES (4, 5);

INSERT INTO Upload VALUES (5, 1);


INSERT INTO Performances VALUES (‘Moonlight Sonata’, ‘2023-10-23’, 1930, ‘123 Maple Street,
Toronto, ON M5V 2N7’, 1, ‘John Doe’, NULL);

INSERT INTO Performances VALUES (‘Rhythmic Fusion’, ‘2023-11-12’, 2000, ‘456 Cedar Avenue,
Vancouver, BC V6B 2P4’, 1, ‘Michelle Jeans’, NULL);

INSERT INTO Performances VALUES (‘Jazz Vibes Extravaganza’, ‘2023-12-05’, 1845, ‘789 Birch
Lane, Calgary, AB T2P 3H9’, 1, ‘Sophia Anderson’, NULL);

INSERT INTO Performances VALUES (‘HarmonyFest2023’, ‘2023-01-18’, 1400, ‘567 Oak Road,
Ottawa, ON K1A 0G9‘, NULL, NULL, ‘Raymond Johnson’);
```

```
INSERT INTO Performances VALUES (‘Rock Legends Reunion’, ‘2024-02-07’, 2315, ‘567 Oak Road,
Ottawa, ON K1A 0G9‘, 1, ‘Carol Brooks’, NULL,);


INSERT INTO Film VALUES (‘Moonlight Sonata’, ‘2023-10-23’, 1930, 5);

INSERT INTO Film VALUES (‘Rhythmic Fusion’, ‘2023-11-12’, 2000, 1);

INSERT INTO Film VALUES (‘Jazz Vibes Extravaganza’, ‘2023-12-05’, 1845, 2);

INSERT INTO Film VALUES (‘HarmonyFest2023’, ‘2023-01-18’, 1400, 3);

INSERT INTO Film VALUES (‘Rock Legends Reunion’, ‘2024-02-07’, 2315, 1);


INSERT INTO Venues VALUES (‘123 Maple Street, Toronto, ON M5V 2N7’, ‘Rogers Arena’, 200, 1);

INSERT INTO Venues VALUES (‘456 Cedar Avenue, Vancouver, BC V6B 2P4’, ‘Oracle Centre’, 100,
2);

INSERT INTO Venues VALUES (‘789 Birch Lane, Calgary, AB T2P 3H9’, ‘TD Garden’, 400, 3);

INSERT INTO Venues VALUES (‘567 Oak Road, Ottawa, ON K1A 0G9‘, ‘Scotiabank Arena’, 50, 4);

INSERT INTO Venues VALUES (‘321 Pine Drive, Montreal, QC H2Z 1J4’, ‘The Sphere’, 250, 5);


INSERT INTO VenuesMID VALUES (1, ‘123 Maple Street, Toronto, ON M5V 2N7’);

INSERT INTO VenuesMID VALUES (2, ‘456 Cedar Avenue, Vancouver, BC V6B 2P4’);

INSERT INTO VenuesMID VALUES (3, ‘789 Birch Lane, Calgary, AB T2P 3H9’);

INSERT INTO VenuesMID VALUES (4, 567 Oak Road, Ottawa, ON K1A 0G9‘);

INSERT INTO VenuesMID VALUES (5, ‘321 Pine Drive, Montreal, QC H2Z 1J4’,);


INSERT INTO VenuesAttributesVALUES (‘Rogers Arena’, 200, 1);
INSERT INTO VenuesAttributesVALUES (‘Oracle Centre’, 100, 2);

INSERT INTO VenuesAttributesVALUES (‘TD Garden’, 400, 3);

INSERT INTO VenuesAttributesVALUES (‘Scotiabank Arena’, 50, 4);

INSERT INTO VenuesAttributesVALUES (‘The Sphere’, 250, 5);


INSERT INTO Managers VALUES (1, ‘Finn Anderson’, ‘123 Maple Street, Toronto, ON M5V 2N7’);

INSERT INTO Managers VALUES (2, ‘Grace Martinez’, ‘456 Cedar Avenue, Vancouver, BC V6B 2P4’);

INSERT INTO Managers VALUES (3, ‘Henry Taylor’, ‘789 Birch Lane, Calgary, AB T2P 3H9’);

INSERT INTO Managers VALUES (4, ‘Isabella Brown’, ‘321 Pine Drive, Montreal, QC H2Z 1J4’);

INSERT INTO Managers VALUES (5, ‘Jackson Wilson’, ‘567 Oak Road, Ottawa, ON K1A 0G9‘);


INSERT INTO ManagersAddress VALUES (‘123 Maple Street, Toronto, ON M5V 2N7’, 1);

INSERT INTO ManagersAddress VALUES (‘456 Cedar Avenue, Vancouver, BC V6B 2P4’, 2);

INSERT INTO ManagersAddress VALUES (‘789 Birch Lane, Calgary, AB T2P 3H9’, 3);

INSERT INTO ManagersAddress VALUES (‘321 Pine Drive, Montreal, QC H2Z 1J4’, 4);

INSERT INTO ManagersAddress VALUES (‘567 Oak Road, Ottawa, ON K1A 0G9‘, 5);


INSERT INTO ManagersAttributes VALUES (‘Finn Anderson’, ‘123 Maple Street, Toronto, ON M5V 2N7’);

INSERT INTO ManagersAttributes VALUES (‘Grace Martinez’, ‘456 Cedar Avenue, Vancouver, BC V6B 2P4’);

INSERT INTO ManagersAttributes VALUES (‘Henry Taylor’, ‘789 Birch Lane, Calgary, AB T2P 3H9’);

INSERT INTO ManagersAttributes VALUES ( ‘Isabella Brown’, ‘321 Pine Drive, Montreal, QC H2Z 1J4’);
INSERT INTO ManagersAttributes VALUES (‘Jackson Wilson’, ‘567 Oak Road, Ottawa, ON K1A
0G9‘);


INSERT INTO StageCrew VALUES (1, 'Oliver Smith', 'Lighting Technician');
INSERT INTO StageCrew VALUES (2, 'Emma Johnson', 'Sound Engineer');
INSERT INTO StageCrew VALUES (3, 'Liam Davis', 'Stage Manager');
INSERT INTO StageCrew VALUES (4, 'Ava Wilson', 'Props Coordinator');
INSERT INTO StageCrew VALUES (5, 'Noah Martinez', 'Costume Designer');


INSERT INTO Manage VALUES (1, 1);
INSERT INTO Manage VALUES (2, 2);
INSERT INTO Manage VALUES (4, 3);
INSERT INTO Manage VALUES (1, 4);
INSERT INTO Manage VALUES (3, 5);


INSERT INTO Tickets VALUES (1, 11, ‘Balcony’, ‘Moonlight Sonata’, ‘2023-10-23’, 1930,
'john@gmail.com');
INSERT INTO Tickets VALUES (2, 11, ‘Balcony’, ‘2023-10-23’, ‘Rhythmic Fusion’, ‘2023-11-12’,
2000,'sarah@gmail.com');
INSERT INTO Tickets VALUES (3, 11’, ‘2023-10-23’, ‘Jazz Vibes Extravaganza’, ‘2023-12-05’, 1845,
'michael@gmail.com');
INSERT INTO Tickets VALUES (4, 11, ‘Balcony’, ‘2023-10-23’, ‘HarmonyFest2023’, ‘2023-01-18’,
1400,, 'emily@gmail.com');
INSERT INTO Tickets VALUES (5, 11, ‘Balcony’, ‘2023-10-23’, ‘Rock Legends Reunion’,
‘2024-02-07’, 2315, 'david@gmail.com');
```

```
INSERT INTO Audience VALUES ('john@gmail.com', 'John Smith');

INSERT INTO Audience VALUES ('sarah@gmail.com', 'Sarah Johnson');

INSERT INTO Audience VALUES ('michael@gmail.com', 'Michael Brown');

INSERT INTO Audience VALUES ('emily@gmail.com', 'Emily Davis');

INSERT INTO Audience VALUES ('david@gmail.com', 'David Wilson');


INSERT INTO Performers VALUES (1, 'Sophia Adams', 'Violin', NULL);

INSERT INTO Performers VALUES (2, 'Ethan Parker', NULL, 'Pop');

INSERT INTO Performers VALUES (3, 'Ava Mitchell', 'Clarinet', NULL);

INSERT INTO Performers VALUES (4, 'Logan Anderson', ‘Flute’, NULL);

INSERT INTO Performers VALUES (5, 'Olivia Turner', NULL, ‘Punk’);


INSERT INTO Perform VALUES (1, ‘Moonlight Sonata’, ‘2023-10-23’, 1930);

INSERT INTO Perform VALUES (3, ‘Rhythmic Fusion’, ‘2023-11-12’, 2000);

INSERT INTO Perform VALUES (4, ‘Jazz Vibes Extravaganza’, ‘2023-12-05’, 1845);

INSERT INTO Perform VALUES (2, ‘HarmonyFest2023’, ‘2023-01-18’, 1400, ‘567 Oak Road,
Ottawa, ON K1A 0G9‘);

INSERT INTO Perform VALUES (5, ‘Rock Legends Reunion’, ‘2024-02-07’, 2315, ‘567 Oak Road,
Ottawa, ON K1A 0G9‘);
```