University of Calgary

Progress Report:

HIPO & DFD Diagrams, SQL Foundation

Group 22: Dylan Dizon, Keenan Hanearin-Balczer, and Anmol Ratol

CPSC 471 W25: Database Management Systems

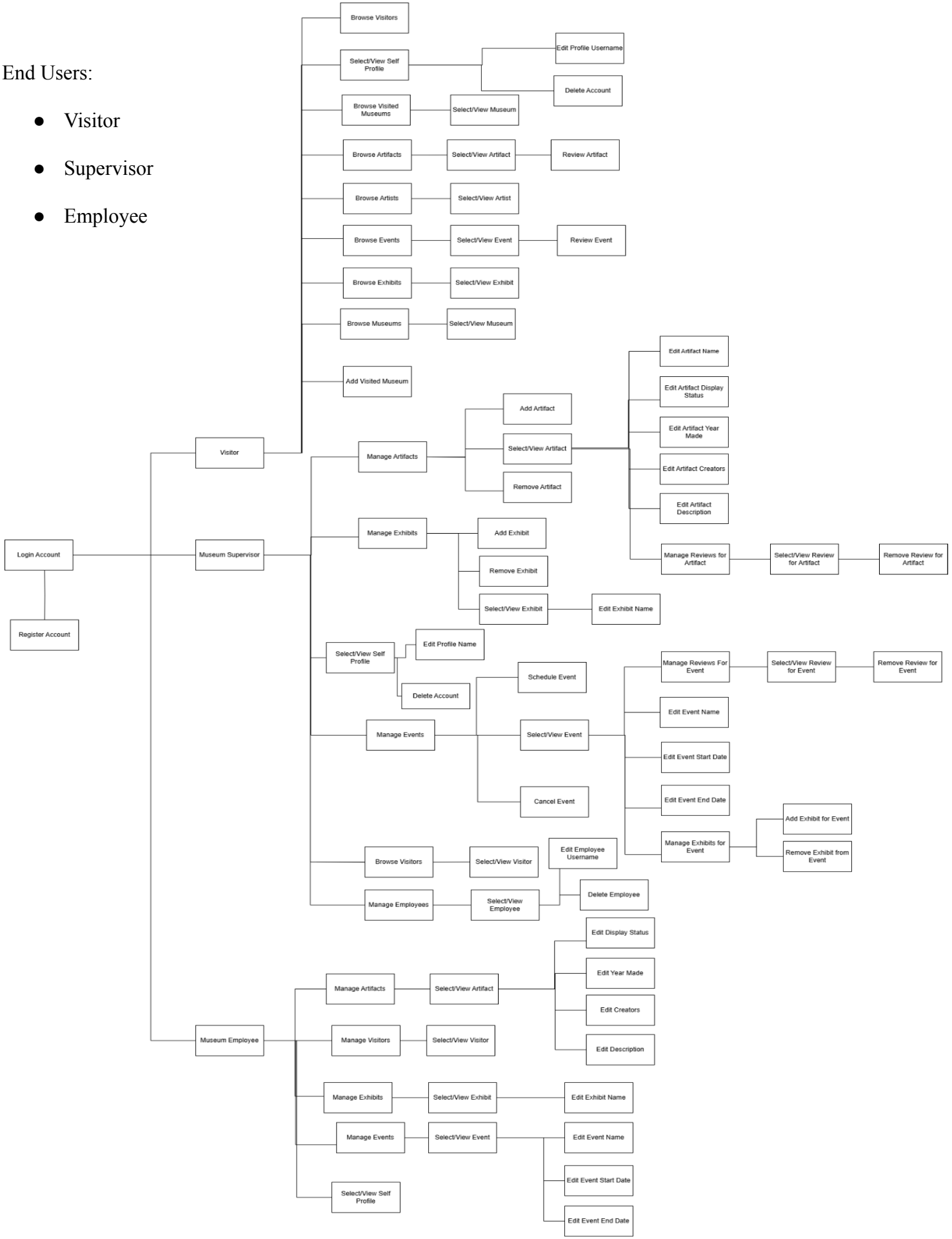
Sara Imani and Dr. Reda Alhajj

21. March 2025

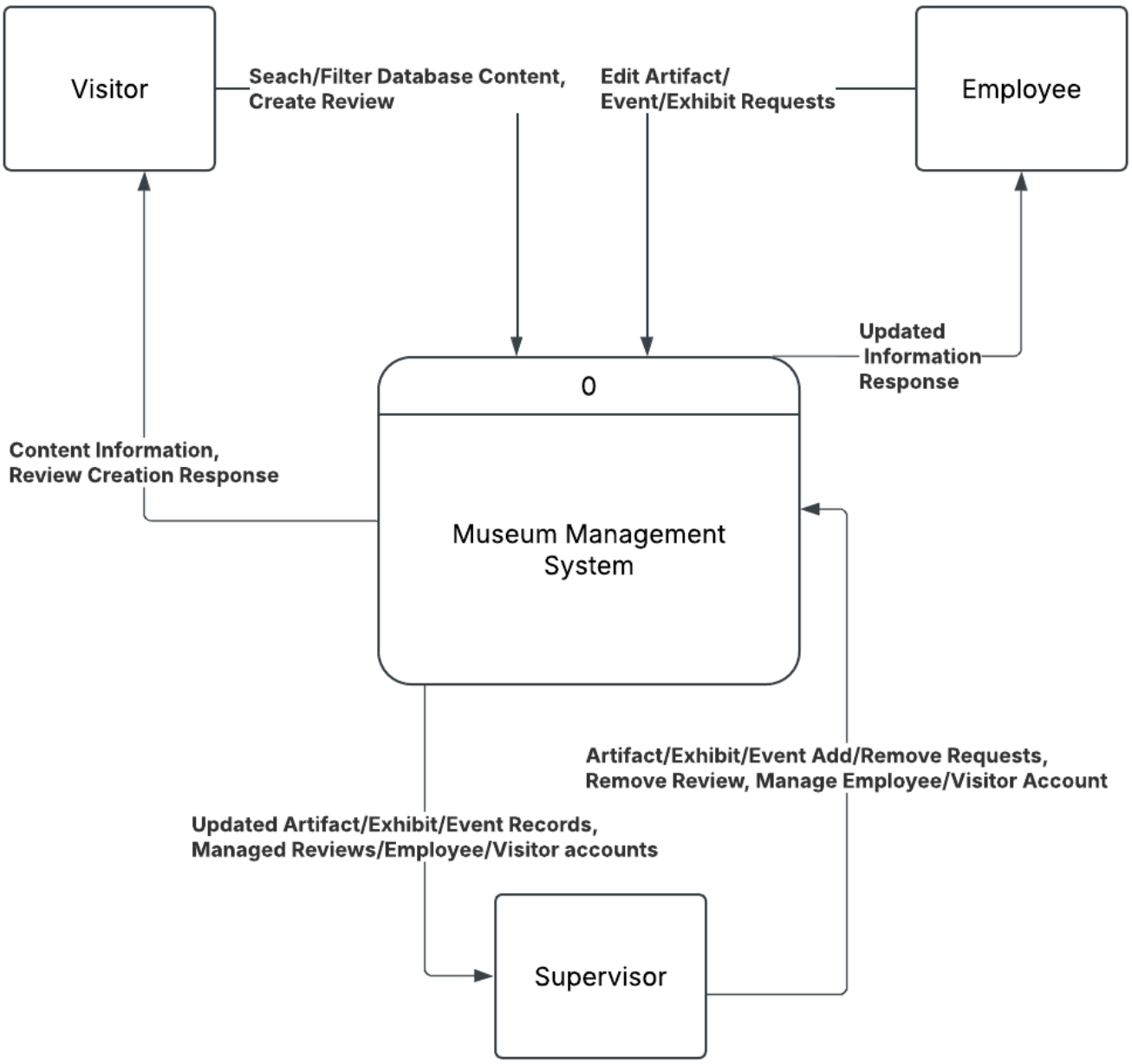
HIPO of Museum & Artifacts DBMS

End Users:

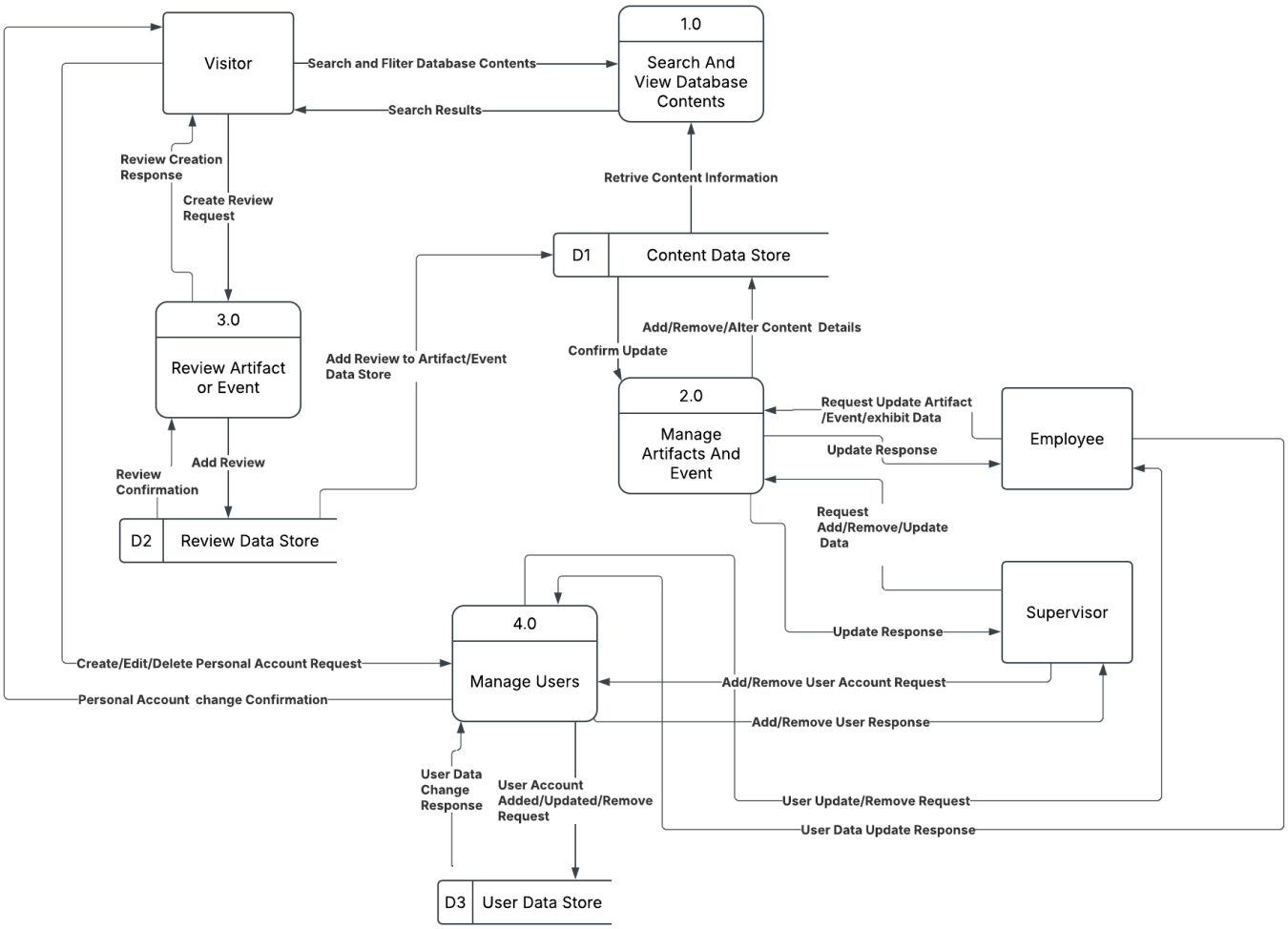
- Visitor
- Supervisor
- Employee



DFD Context Diagram of Museum & Artifacts DBMS



DFD Level-0 Diagram of Museum & Artifacts DBMS



External Entities

Visitor:

- Search/Filter Database Content: Finds artifacts and events.
- Create Review: Submits reviews for artifacts or events.
- Account Management: Creates, updates, or deletes a personal account.

Employee:

- Edit Artifacts/Events/Exhibits: Updates content in the system.
- Receive Update Responses: Confirms updates made.
- Account Management: updates, or deletes a personal account.

Supervisor:

- Manage Artifacts/Exhibits/Events: Adds, removes, or updates content.
- Manage Users: Adds, updates, or removes employee or visitor accounts.
- Remove Reviews: Deletes inappropriate reviews.
- Receive Update Responses: Confirms successful content and user updates.

Processes

Search and View Database Contents (1.0):

- Visitor searches for artifacts or events.
- Retrieves data from the Content Data Store (D1).
- Shows search results and detailed information.

Manage Artifacts and Events (2.0):

- Employees submit update requests for artifacts or events.
- Supervisors can directly add or remove artifacts, events, or exhibits.
- Updates are stored in the Content Data Store (D1).
- Supervisors also approve employee updates.

Review Artifact or Event (3.0):

- Visitors add reviews for artifacts or events.
- Reviews are saved in the Review Data Store (D2).
- The system sends review confirmations.

Manage Users (4.0):

- Supervisors, Employees, and Visitors can create, update, or delete their personal accounts.
- Supervisors can delete all user accounts.
- User data is stored in the User Data Store (D3).
- The system sends account change confirmations.

Data Stores

Content Data Store (D1):

- Holds information about artifacts, events, exhibits (it also holds information on museums and artists, but these can't be interacted with directly).

Review Data Store (D2):

- Stores visitor reviews.

User Data Store (D3):

- Maintains user account information.

SQL Database

```
CREATE DATABASE MuseoTrackDB;
USE MuseoTrackDB;

CREATE TABLE MUSEUM (
    Address VARCHAR(255) PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
);

CREATE TABLE SUPERVISOR (
    SEmail VARCHAR(255),
    MuseumAddress VARCHAR(255),
    PRIMARY KEY(SEmail),
    FOREIGN KEY(SEmail) REFERENCES USER(Email) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY(MuseumAddress) REFERENCES MUSEUM(Address)
);

CREATE TABLE EVENT (
    EvID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Start_Date VARCHAR(255) NOT NULL,
    End_Date VARCHAR(255) NOT NULL,
    Address VARCHAR(255),
);

CREATE TABLE ARTIST (
    AID INT PRIMARY KEY,
    Date_of_Birth VARCHAR(255),
    First_Name VARCHAR(255) NOT NULL,
    Middle_Name VARCHAR(255),
    Last_Name VARCHAR(255) NOT NULL
);

CREATE TABLE ARTIST_WORKS (
    AID INT,
    Works VARCHAR(255),
    PRIMARY KEY(AID, Works),
    FOREIGN KEY(AID) REFERENCES ARTIST(AID) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE EXHIBIT (
    ExID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
```

```
    Address VARCHAR(255),
);

CREATE TABLE ARTIFACT (
    ArtID INT PRIMARY KEY,
    Description VARCHAR(255),
    Year_Made VARCHAR(255),
    Display_Status VARCHAR(255),
    Name VARCHAR(255) NOT NULL,
    SEmail VARCHAR(255) NOT NULL,
    ExID INT NOT NULL,
    FOREIGN KEY(SEmail) REFERENCES SUPERVISOR(SEmail) ON DELETE CASCADE ON UPDATE
CASCADE,
    FOREIGN KEY(ExID) REFERENCES EXHIBIT(ExID) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE ARTIFACT_CREATORS (
    ArtID INT,
    Creators VARCHAR(255),
    PRIMARY KEY(ArtID, Creators),
    FOREIGN KEY(ArtID) REFERENCES ARTIFACT(ArtID) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE ARTIFACT_RATINGS (
    ArtID INT,
    Ratings INT,
    PRIMARY KEY(ArtID, Ratings),
    FOREIGN KEY(ArtID) REFERENCES ARTIFACT(ArtID)
);

CREATE TABLE CREATES (
    ArtID INT,
    AID INT,
    PRIMARY KEY(ArtID, AID),
    FOREIGN KEY(ArtID) REFERENCES ARTIFACT(ArtID) ON DELETE CASCADE ON UPDATE
CASCADE,
    FOREIGN KEY(AID) REFERENCES ARTIST(AID) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE USER (
    Email VARCHAR(255) PRIMARY KEY,
    First_Name VARCHAR(255) NOT NULL,
```

```
    Middle_Name VARCHAR(255),
    Last_Name VARCHAR(255) NOT NULL,
    Username VARCHAR(255) NOT NULL,
    Password VARCHAR(255) NOT NULL,
    Year_of_Birth INT
);

CREATE TABLE EMPLOYEE (
    EEmail VARCHAR(255),
    SEmail VARCHAR(255),
    MuseumAddress VARCHAR(255),
    PRIMARY KEY(EEmail),
    FOREIGN KEY(EEmail) REFERENCES USER(Email) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY(SEmail) REFERENCES SUPERVISOR(SEmail) ON DELETE CASCADE ON UPDATE
CASCADE
    FOREIGN KEY(MuseumAddress) REFERENCES MUSEUM(Address)
);

CREATE TABLE VISITOR (
    VEmail VARCHAR(255),
    PRIMARY KEY(VEmail),
    FOREIGN KEY(VEmail) REFERENCES USER(Email) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE EDITS (
    EEmail VARCHAR(255),
    ArtID INT,
    PRIMARY KEY(EEmail, ArtID),
    FOREIGN KEY(EEmail) REFERENCES EMPLOYEE(EEmail) ON DELETE CASCADE ON UPDATE
CASCADE,
    FOREIGN KEY(ArtID) REFERENCES ARTIFACT(ArtID) ON DELETE CASCADE ON UPDATE
CASCADE
);


CREATE TABLE REVIEW_EVENT (
    VEmail VARCHAR(255),
    EvID INT,
    Review_Desc VARCHAR(255),
    PRIMARY KEY(VEmail, EvID),
    FOREIGN KEY(VEmail) REFERENCES VISITOR(VEmail) ON UPDATE CASCADE,
    FOREIGN KEY(EvID) REFERENCES EVENT(EvID) ON UPDATE CASCADE
);
```

```
CREATE TABLE REVIEW_ARTIFACT (
    VEmail VARCHAR(255),
    ArtID INT,
    Review_Desc VARCHAR(255),
    PRIMARY KEY(VEmail, ArtID),
    FOREIGN KEY(VEmail) REFERENCES VISITOR(VEmail) ON UPDATE CASCADE,
    FOREIGN KEY(ArtID) REFERENCES ARTIFACT(ArtID) ON UPDATE CASCADE
);

CREATE TABLE PART_OF (
    EvID INT,
    ExID INT,
    PRIMARY KEY(EvID, ExID),
    FOREIGN KEY(EvID) REFERENCES EVENT(EvID) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY(ExID) REFERENCES EXHIBIT(ExID) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE VISITS (
    VEmail VARCHAR(255),
    Address VARCHAR(255),
    PRIMARY KEY(VEmail, Address),
    FOREIGN KEY(VEmail) REFERENCES VISITOR(VEmail) ON UPDATE CASCADE,
    FOREIGN KEY(Address) REFERENCES MUSEUM(Address) ON UPDATE CASCADE
);
```

Other SQL Statements

```
-- registerAccountForVisitor(@VEmail, @First_Name, @Middle_Name, @Last_Name,
@Username, @Password, @Year_of_Birth)
INSERT INTO USER
VALUES (@VEmail, @First_Name, @Middle_Name, @Last_Name, @Username, @Password,
@Year_of_Birth);

INSERT INTO VISITOR (VEmail)
VALUES (@VEmail);

-- registerAccountForSupervisor(@SEmail, @First_Name, @Middle_Name, @Last_Name,
@Username, @Password, @Year_of_Birth)
INSERT INTO USER
VALUES (@SEmail, @First_Name, @Middle_Name, @Last_Name, @Username, @Password,
@Year_of_Birth);
```

```
INSERT INTO SUPERVISOR
VALUES (@SEmail);

-- registerAccountForEmployee(@EEmail, @First_Name, @Middle_Name, @Last_Name,
@Username, @Password, @Year_of_Birth, @SEmail)
INSERT INTO USER
VALUES (@EEmail, @First_Name, @Middle_Name, @Last_Name, @Username, @Password,
@Year_of_Birth);

INSERT INTO EMPLOYEE
VALUES (@EEmail, @SEmail);

-- browseVisitor(@VEmail)
SELECT V.VEmail, U.First_Name, U.Middle_Name, U.Last_Name, U.Username,
U.Year_of_Birth
FROM VISITOR AS V
     JOIN USER AS U ON U.Email = V.VEmail
WHERE V.VEmail = @VEmail;

-- selectSelfProfile(@Email)
SELECT Email, First_Name, Middle_Name, Last_Name, Username, Year_of_Birth
FROM USER
WHERE Email = @Email;

-- editProfileUserName(@Email)
UPDATE USER
SET Username = @Username
WHERE Email = @Email;

-- deleteAccount(@Email, @Fname, @Mname, @Lname, @Username)
DELETE FROM USER
WHERE Email = @Email
    AND First_Name = @Fname
    AND Middle_Name = @Mname
    AND Last_Name = @Lname
    AND Username = @Username;

-- addVisitedMuseum(@VEmail, @Address)
INSERT INTO VISITS
VALUES (@VEmail, @Address);

-- browseVisitedMuseums(@VEmail)
SELECT M.Name, M.Address
```

```
FROM VISITS AS V
JOIN MUSEUM AS M ON V.Address = M.Address
WHERE V.VEmail = ?;

-- selectMuseum(@Address)
SELECT Name, Address
FROM MUSEUM
WHERE Address = @Address;

-- browseArtifacts(@Name)
SELECT ArtID, Name, Description, Year_Made, ExID
FROM ARTIFACT
WHERE Name = @Name;

-- selectArtifact(@ArtID)
SELECT A.ArtID, A.Name, A.Description, A.Year_Made,
       M.Name AS Museum_Name, E.Name AS Exhibit_Name, S.First_Name, S.Last_Name
FROM ARTIFACT AS A
JOIN EXHIBIT AS E ON A.ExID = E.ExID
JOIN MUSEUM AS M ON E.Address = M.Address
JOIN SUPERVISOR AS S ON A.SEmail = S.SEmail
WHERE A.ArtID = @ArtID;

-- reviewArtifact(@VEmail, @ArtID, @Review_Desc)
INSERT  INTO REVIEW_ARTIFACT
VALUES(@VEmail, @ArtID, @Review_Desc);

-- browseArtists(@First_Name, @Last_Name)
SELECT A.AID, A.Date_of_Birth, A.Date_of_Death, A.First_Name, A.Middle_Name,
A.Last_Name, ART.ArtID, ART.ArtName
FROM ARTIST AS A
    JOIN ARTIST_WORKS AS ART ON ART.AID = A.AID
WHERE @First_Name = A.First_Name
    AND @Last_Name = A.Last_Name;

-- selectArtist(@AID)
SELECT AID, First_Name, Middle_Name, Last_Name, Date_of_Birth
FROM ARTIST
WHERE AID = @AID;

-- browseEvents(@Name)
SELECT EV.EvID, EV.Name, EV.Start_Date, EV.End_Date, EV.Address, EX.Name
FROM EVENT AS EV
```

```
    JOIN PART_OF AS PO ON PO.EvID = EV.EvID
    JOIN EXHIBIT AS EX ON EX.ExID = PO.ExID
WHERE @Name = EV.Name;

-- selectEvent(@EvID)
SELECT EvID, Name, Start_Date, End_Date, Address
FROM EVENT
WHERE EvID = @EvID;

-- reviewEvent(@VEmail, @EvID, @Review_Desc)
INSERT INTO REVIEW_EVENT
VALUES (@VEmail, @EvID, @Review_Desc);

-- browseExhibits(@Name)
SELECT EX.ExID, EX.Name, EX.Address, ART.Name
FROM EXHIBIT AS EX
    JOIN ARTIFACT AS ART ON EX.ExID = EX.ExID
WHERE @Name = EX.Name;

-- selectExhibit(@ExID)
SELECT E.ExID, E.Name, E.Address
FROM EXHIBIT AS
WHERE ExID = @ExID;

-- browseMuseums(@Name)
SELECT M.Name, M.Address, EX.Name
FROM MUSEUM AS M
    JOIN EXHIBIT AS EX ON EX.Address = M.Address
WHERE @Name = M.Name

-- manageArtifacts(@SEmail)
SELECT A.ArtID, A.Name AS Artifact_Name, A.Description, A.Year_Made, E.Name AS
Exhibit_Name
FROM ARTIFACT AS A
JOIN EXHIBIT AS E ON A.ExID = E.ExID
JOIN SUPERVISOR AS S ON A.SEmail = S.SEmail
WHERE S.SEmail = @SEmail;

-- addArtifact(@ArtID, @Name, @Description, @Year_Made, @SEmail, @ExID)
INSERT INTO ARTIFACT
VALUES (@ArtID, @Name, @Description, @Year_Made, @SEmail, @ExID);

-- selectArtifact(@ArtID)
```

```sql
SELECT A.ArtID, A.Name, A.Description, A.Year_Made, E.Name AS ExhibitName, M.Name
AS MuseumName
FROM ARTIFACT A
JOIN EXHIBIT E ON A.ExID = E.ExID
JOIN MUSEUM M ON E.Address = M.Address
WHERE A.ArtID = @ArtID;

-- editArtifactName(@ArtID, @Name)
UPDATE ARTIFACT
SET Name = @Name
WHERE ArtID = @ArtID;

-- editArtifactDisplayStatus(@ArtID, @Display_Status)
UPDATE ARTIFACT
SET Display_Status = @Display_Status
WHERE ArtID = @ArtID;

-- editArtifactYearMade(@ArtID, @Year_Made)
UPDATE ARTIFACT
SET Year_Made = @Year_Made
WHERE ArtID = @ArtID;

-- editArtifactCreators(@ArtID, @Creators)
UPDATE ARTIFACT
SET Creators = @Creators
WHERE ArtID = @ArtID;

-- editArtifactDescription(@ArtID, @Description)
UPDATE ARTIFACT
SET Description = @Description
WHERE ArtID = @ArtID;

-- removeArtifact(@ArtID, @Name, @ExID)
DELETE FROM ARTIFACT
WHERE ArtID = @ArtID
    AND Name = @Name
    AND ExID = @ExID;

-- manageExhibits(@SEmail)
SELECT E.ExID, E.Name, E.Address
FROM EXHIBIT AS E
JOIN MUSEUM AS M ON E.Address = M.Address
JOIN SUPERVISOR AS S ON M.Address = S.MuseumAddress
```

```
WHERE S.SEmail = @SEmail;

-- addExhibit(@ExID, @Name, @Address)
INSERT INTO EXHIBIT
VALUES (@ExID, @Name, @Address)

-- removeExhibit(@ExID, @Name, @Address)
DELETE FROM EXHIBIT
WHERE ExID = @ExID
    AND Name = @Name
    AND Address = @Address;

-- selectExhibit(@ExID)
SELECT *
FROM EXHIBIT
WHERE ExID = @ExID;

-- editExhibitName(@ExID, @Name)
UPDATE EXHIBIT
SET Name = @Name
WHERE ExID = @ExID;

-- manageEvents(@SEmail)
SELECT E.EvID, E.Name, E.Start_Date, E.End_Date, E.Address
FROM EVENT E
JOIN PART_OF P ON E.EvID = P.EvID
JOIN EXHIBIT EX ON P.ExID = EX.ExID
JOIN MUSEUM M ON EX.Address = M.Address
JOIN SUPERVISOR S ON M.Address = S.MuseumAddress
WHERE S.SEmail = @SEmail;

-- scheduleEvent(@EvID, @Name, @Start_Date, @End_Date, @Address)
INSERT INTO EVENT
VALUES (@EvID, @Name, @Start_Date, @End_Date, @Address);

-- selectEvent(@EvID)
SELECT *
FROM EVENT
WHERE EvID = @EvID;

-- editEventName(@EvID, @Name)
UPDATE EVENT
SET Name = @Name
```

```
WHERE EvID = @EvID;

-- editEventStartDate(@EvID, @Start_Date)
UPDATE EVENT
SET Start_Date = @Start_Date
WHERE EvID = @EvID;

-- editEventEndDate(@EvID, @End_Date)
UPDATE EVENT
SET End_Date = @End_Date
WHERE EvID = @EvID;

-- manageExhibitsForEvent(@SEmail)
SELECT E.ExID, E.Name, E.Address
FROM EXHIBIT AS E
JOIN PART_OF AS P ON E.ExID = P.ExID
JOIN EVENT AS EV ON P.EvID = EV.EvID
JOIN SUPERVISOR AS S ON EV.Address = S.MuseumAddress
WHERE S.SEmail = @SEmail;

-- addExhibitForEvent(@EvID, @ExID)
INSERT INTO PART_OF
VALUES (@EvID, @ExID);

-- removeExhibitFromEvent(@EvID, @ExID)
DELETE FROM PART_OF
WHERE EvID = @EvID
    AND ExID = @ExID;

-- manageEmployees(@SEmail)
SELECT E.EEmail, U.First_Name, U.Middle_Name, U.Last_Name, E.MuseumAddress
FROM EMPLOYEE AS E
     JOIN USER AS U ON E.EEmail = U.Email
     JOIN SUPERVISOR AS S ON E.MuseumAddress = S.MuseumAddress
WHERE S.SEmail = @SEmail;

-- selectEmployee(@EEmail)
SELECT E.EEmail, U.First_Name, U.Middle_Name, U.Last_Name, E.MuseumAddress
FROM EMPLOYEE AS E
     JOIN USER AS U ON E.EEmail = U.Email
     JOIN SUPERVISOR AS S ON E.MuseumAddress = S.MuseumAddress
WHERE E.EEmail = @EEmail;
```

```sql
-- manageReviewsForArtifact(@SEmail)
SELECT R.VEmail, U.First_Name, U.Middle_Name, U.Last_Name,
       R.ArtID, A.Name AS Artifact_Name, R.Review_Desc
FROM REVIEW_ARTIFACT AS R
JOIN ARTIFACT AS A ON R.ArtID = A.ArtID
JOIN EXHIBIT AS E ON A.ExID = E.ExID
JOIN MUSEUM AS M ON E.Address = M.Address
JOIN SUPERVISOR AS S ON M.Address = S.MuseumAddress
JOIN USER AS U ON R.VEmail = U.Email
WHERE S.SEmail = @SEmail;

-- selectReviewForArtifact(@ArtID)
SELECT R.VEmail, U.First_Name, U.Middle_Name, U.Last_Name,
       R.ArtID, A.Name AS Artifact_Name, R.Review_Desc
FROM REVIEW_ARTIFACT AS R
JOIN ARTIFACT AS A ON R.ArtID = A.ArtID
JOIN USER AS U ON R.VEmail = U.Email
WHERE A.ArtID = @ArtID;

-- removeReview(@VEmail, @ArtID)
DELETE FROM REVIEW_ARTIFACT
WHERE VEmail = @VEmail AND ArtID = @ArtID;

-- deleteEmployee(@EEmail)
DELETE FROM EMPLOYEE
WHERE EEmail = @EEmail;

-- editEmployeeUserame(@EEmail, @NewUsername)
UPDATE USER
SET Username = @NewUsername
WHERE Email = @EEmail
    AND Email IN (
        SELECT EEmail
        FROM EMPLOYEE;
    );
```