```
-- Daniel Carpenter
-- 113009743

--
================================================================
-- PROBLEM 2(a) -- Create Relations
--
================================================================

USE [cs-dsa-4513-sql-db];

-- Drop tables since this is a repeated excercise for database
design
DROP TABLE IF EXISTS [Acted];
DROP TABLE IF EXISTS [Movie];
DROP TABLE IF EXISTS [Director];
DROP TABLE IF EXISTS [Performer];

-- CREATE TABLES
==================================================

   -- Performer table
--------------------------------------------
   CREATE TABLE [Performer] (
       pid                 INT PRIMARY KEY,
       pname               VARCHAR(64) NOT NULL,
       years_of_experience INT NOT NULL,
       age                 INT NOT NULL
   )

   -- Director Table
---------------------------------------------
   CREATE TABLE [Director] (
       did      INT PRIMARY KEY,
       dname    VARCHAR(64) NOT NULL,
       earnings REAL NOT NULL
   )

   -- Movie Table
------------------------------------------------
   CREATE TABLE [Movie] (
       [mname]         VARCHAR(64) PRIMARY KEY,
       [genre]         VARCHAR(64) NOT NULL,
       [minutes]       INT NOT NULL,
       [release_year]  INT NOT NULL,
       [did]           INT FOREIGN KEY REFERENCES [Director]
   )

   -- Movie Table            Primary key?   -0.5
------------------------------------------------
   CREATE TABLE [Acted] (
       pid     INT         FOREIGN KEY REFERENCES [Performer],
       mname   VARCHAR(64) FOREIGN KEY REFERENCES [Movie]
   )


--
================================================================
-- PROBLEM 2(b) -- Populate Tables
--
================================================================

   -- Performer table
--------------------------------------------
   INSERT INTO Performer
       (pid, pname, years_of_experience, age)
   VALUES
       (1, 'Morgan',48, 67),
       (2, 'Cruz',  14, 28),
       (3, 'Adams', 1,  16),
       (4, 'Perry', 18, 32),
       (5, 'Hanks', 36, 55),
       (6, 'Hanks', 15, 24),                   ✓
       (7, 'Lewis', 13, 32)

   -- Director table --------------------------------------------
   INSERT INTO Director
       (did, dname, earnings)
   VALUES
       (1, 'Parker', 580000),
       (2, 'Black',  2500000),
       (3, 'Black',  30000),
       (4, 'Stone',  820000)

   -- Movie table -----------------------------------------------
   INSERT INTO Movie
       ([mname], [genre], [minutes], [release_year], [did])
   VALUES
       ('Jurassic Park',        'Action',      125, 1984, 2),
       ('Shawshank Redemption', 'Drama',       105, 2001, 2),
       ('Fight Club',           'Drama',       144, 2015, 2),
       ('The Departed',         'Drama',       130, 1969, 3),
       ('Back to the Future',   'Comedy',      89,  2008, 3),
       ('The Lion King',        'Animation',   97,  1990, 1),
       ('Alien',                'Sci-Fi',      115, 2006, 3),
       ('Toy Story',            'Animation',   104, 1978, 1),
       ('Scarface',             'Drama',       124, 2003, 1),
       ('Up',                   'Animation',   111, 1999, 4)

   -- Acted table -----------------------------------------------
   INSERT INTO Acted
       (pid, mname)
   VALUES
       (4, 'Fight Club'),
       (5, 'Fight Club'),
       (6, 'Shawshank Redemption'),
       (4, 'Up'),
       (5, 'Shawshank Redemption'),
       (1, 'The Departed'),
       (2, 'Fight Club'),
       (3, 'Fight Club'),
       (4, 'Alien')
```

✓

```
--
=================================================================
-- PROBLEM 2(c) -- SQL Queries (1 - 9)
--
=================================================================

-- 1. Display all the data you store in the database to verify
that you have populated the relations correctly.
   SELECT
       [perf].[pid],
       [pname],
       [years_of_experience],
       [age],
       [movi].[mname],
       [genre],
       [minutes],
       [release_year],
       [direct].[did],
       [dname],
       [earnings]
```

Have 4 separate SELECT * FROM X queries for every

```
   FROM
       -- Performer data
       [cs-dsa-4513-sql-db].[dbo].[Performer]          AS [perf]

       -- Acted data
       LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Acted]    AS [act]
           ON [perf].pid = [act].pid

       -- Movie Data
       RIGHT JOIN [cs-dsa-4513-sql-db].[dbo].[Movie]    AS [movi]
           ON [act].mname = [movi].mname

       -- Director Data
       LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Director] AS
[direct]
           ON [movi].did = [direct].did


-- 2. Find the names of all Action movies.
   SELECT mname
   FROM [cs-dsa-4513-sql-db].[dbo].[Movie]
   WHERE genre = 'Action'
```

✓

```
-- 3. For each genre, display the genre and the average length
(minutes) of movies for that genre.
   SELECT
       genre,
       AVG([minutes]) AS avgMinutes

   FROM [cs-dsa-4513-sql-db].[dbo].[Movie]
   GROUP BY genre
```

✓

```
-- 4. Find the names of all performers with at least 20 years of
experience who have acted in a movie directed by Black.
   SELECT DISTINCT
       [perf].pid,
       [perf].pname

   FROM
       -- Performer data
       [cs-dsa-4513-sql-db].[dbo].[Performer]          AS [perf]

       -- Acted data
```

```
        LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Acted]    AS [act]
            ON [perf].pid = [act].pid

        -- Movie Data
        RIGHT JOIN [cs-dsa-4513-sql-db].[dbo].[Movie]    AS [movi]
            ON [act].mname = [movi].mname

        -- Director Data
        LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Director] AS
[direct]
            ON [movi].did = [direct].did

    WHERE
        -- performers with at least 20 years of experience
            years_of_experience >= 20

        -- directed by Black.
        AND dname = 'Black'


-- 5. Find the age of the oldest performer who is either named or
has acted in a movie named .
    SELECT MAX(age) AS maxAge
    FROM (
        SELECT
            age
        FROM
            -- Performer data
            [cs-dsa-4513-sql-db].[dbo].[Performer]          AS
[perf]


            -- Acted data
            LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Acted]    AS
[act]
                ON [perf].pid = [act].pid

            -- Movie Data
            RIGHT JOIN [cs-dsa-4513-sql-db].[dbo].[Movie]    AS
[movi]
                ON [act].mname = [movi].mname
        WHERE
            pname = 'Hanks'
```

```
       OR  [movi].mname = 'The Departed'
   ) AS maxAgeTbl


-- 6. Find the names of all movies that are either a Comedy or
have had more than one performer act in them.
   SELECT
       [movi].mname
       -- genre,
       -- COUNT([movi].mname) AS numPerformers
   FROM
       -- Performer data
       [cs-dsa-4513-sql-db].[dbo].[Performer]          AS [perf]

       -- Acted data
       LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Acted]    AS [act]
           ON [perf].pid = [act].pid

       -- Movie Data
       RIGHT JOIN [cs-dsa-4513-sql-db].[dbo].[Movie]    AS [movi]
           ON [act].mname = [movi].mname

   GROUP BY
       [movi].mname,
       genre

   HAVING
       -- Had more than one performer act in a movie, OR
           COUNT([perf].pid) > 1

       -- Is of the genre Comedy
       OR  genre = 'Comedy'


-- 7. Find the names and pid's of all performers who have acted
in at least two movies that have the same genre.
   SELECT
       [perf].pid,
       pname
       -- genre,
       -- COUNT([movi].mname) AS countOfMovies
   FROM
       -- Performer data
       [cs-dsa-4513-sql-db].[dbo].[Performer]          AS [perf]

       -- Acted data
       LEFT JOIN [cs-dsa-4513-sql-db].[dbo].[Acted]    AS [act]
           ON [perf].pid = [act].pid

       -- Movie Data
       RIGHT JOIN [cs-dsa-4513-sql-db].[dbo].[Movie]    AS [movi]
           ON [act].mname = [movi].mname

   -- Account for movies with no performers listed
   WHERE [perf].pid IS NOT NULL

   GROUP BY
       [perf].pid,
       pname,
       genre

   -- acted in at least two movies
   HAVING COUNT([movi].mname) >= 2


-- 8. Decrease the earnings of all directors who directed by 10%.

   -- Inputs for decrease % for salaries, and the name of the
movie who directed
   DECLARE @decreaseAmt AS REAL
   DECLARE @movieName   AS VARCHAR(64)

   SET @decreaseAmt = -0.10
   SET @movieName   = 'Up'

   -- Update table to show new earnings
   UPDATE Director
       SET earnings = earnings * (1 + @decreaseAmt)

       -- Get director's did who directed selected Movie ('Up')
       WHERE
           did = (
               SELECT [direct].did

               FROM
                   [cs-dsa-4513-sql-db].[dbo].[Movie]    AS
[movi]

                   -- Director Data
                   LEFT JOIN [cs-dsa-4513-sql-db].[dbo].
[Director] AS [direct]
                       ON [movi].did = [direct].did

               WHERE mname = @movieName
       )


-- 9. Delete all movies released in the 70's and 80's (1970 <=
release_year <= 1989).

   DELETE FROM Movie
   WHERE Movie.release_year BETWEEN 1970 AND 1989
```