|  |  |
| --- | --- |
|  | **CODE: BSQL\_Assignment2\_Opt3**  **TYPE: n/a**  **LOC: n/a**  **DURATION: 180 MINUTES** |

# For the following assignments:

* Print out respectively the screenshots to show the query results.
* Pack **screenshots** and **SQL scripts** or **your answers** into the zip file named BSQL\_Assignment<i>\_AccountName.zip (for instance: BSQL\_Assignment1\_NamNT.zip) then handle to the evaluator via email ([XYZ@fsoft.com.vn](mailto:XYZ@fsoft.com.vn) ) or follow the guidance of the class admin.

# Day 2. Lesson 2: SQL Basic

## Assignment 2\_Opt3: Movie Management

**Barem**: Q1 - 40%, Q2.a – 10%, Q2.b – 10%, Q3.each subquestion - 10%.

**Objective**: H5SD - SQL skills

**Problem Description**:

Building a Movie Collection database to store information about movies.

**Questions to answer**:

**Q1**: Create your tables

* 1. Create a table called **Movie** to store information about movies. Add columns in your table for *movie name*, *duration*, *genre*, *director*, *amount of money* made at the box office and *comments*.
* Make sure you one of your columns works as a PRIMARY KEY.
* Genre: accepts value range from 1 to 8 only (1: Action, 2: Adventure, 3: Comedy, 4: Crime (gangster), 5: Dramas, 6: Horror, 7: Musical/dance, 8: War)
* Duration: must be greater than or equal 1 hours
  1. Create another table called **Actor** to store information about actors. Just like you did with Movie, add several columns to store actor data for the *actor's name*, *age*, *average movie salary*, and *Nationality*. Again, make sure there is a PRIMARY KEY in your table.
  2. Create a final table called **ActedIn** to store information about which movies certain actors have acted in. Think carefully about what the columns of this table should be. This table should make use of FOREIGN KEYS.

Create above tables with the most appropriate/economic field/column constraints & types, all fields are mandatory except Comments field.

**Q2.** Polulate tables

1. Add an ImageLink field to Movie table and make sure that the database will not allow the value for ImageLink to be inserted into a new row if that value has already been used in another row.
2. Populate your tables with some data using the INSERT statement. Make sure you have at least 5 tuples per table.

You accidentally mis-typed one of the actors' names. Fix your typo by using an UPDATE statement.

**Q3**. Query tables

1. Write a query to retrieve all the data in the Actor table for actors that are older than 50.
2. Write a query to retrieve all actor names and average salaries from ACTOR and sort the results by average salary.
3. Using an actor name from your table, write a query to retrieve the names of all the movies that actor has acted in.
4. Write a query to retrieve the names of all the action movies that amount of actor be greater than 3

**Estimated Time to complete**:180 mins.

My answer

CREATE DATABASE BSQL\_Assignment\_Day02\_03;

GO

USE BSQL\_Assignment\_Day02\_03;

Go

--Q1.1

--Create a table called Movie to store information about movies.

--Add columns in your table for:

-->Movie name

-->Duration

-->Genre

-->Director

-->Amount of money made at the box office

-->Comments.

--Make sure one of your columns works as a PRIMARY KEY.

--Genre: accepts value range from 1 to 8 only (1: Action, 2: Adventure, 3: Comedy, 4: Crime (gangster), 5: Dramas, 6: Horror, 7: Musical/dance, 8: War)

--Duration: must be greater than or equal 1 hours



DROP TABLE IF EXISTS Movie;

CREATE TABLE Movie (

MovieID char(4),

MovieName nvarchar(100),

Duration time,

Genre nvarchar(100),

Director nvarchar(100),

AmountMoney money,

Comments nvarchar(max),

CONSTRAINT PK\_MovieID PRIMARY KEY (MovieID),

CONSTRAINT CHK\_Genre CHECK (Genre in ('Action','Adventure','Comedy','Crime (Gangster)', 'Dramas' ,'Horror','Musical/Dance', 'War')),

CONSTRAINT CHK\_Duration CHECK (Duration >= '1:00:00')

);

Go

--Q1.2

--Create another table called Actor to store information about actors.

--Just like you did with Movie, add several columns to store actor data for the actor's:

-->Name

-->Age

-->Average movie salary

-->Nationality

--Again, make sure there is a PRIMARY KEY in your table.

DROP TABLE IF EXISTS Actor;

CREATE TABLE Actor (

ActorID char(4),

ActorName nvarchar(100),

ActorAge nvarchar(100),

AverageMovieSalary money,

Nationality nvarchar(100),

CONSTRAINT PK\_ActorID PRIMARY KEY (ActorID)

);

GO

--Q1.3

--Create a final table called ActedIn to store information about which movies certain actors have acted in.

--Think carefully about what the columns of this table should be.

--This table should make use of FOREIGN KEYS.

CREATE TABLE ActedIn (

MovieID char(4),

ActorID char(4),

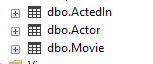
CONSTRAINT PK\_MovieID\_ActorID PRIMARY KEY (MovieID, ActorID),

CONSTRAINT FK\_MovieID FOREIGN KEY (MovieID) REFERENCES Movie(MovieID),

CONSTRAINT FK\_ActorID FOREIGN KEY (ActorID) REFERENCES Actor(ActorID)

);

GO



--Q2.1

--Add an ImageLink field to Movie table

--And make sure that the database will not allow the value for ImageLink to be inserted into a new row if that value has already been used in another row.

ALTER TABLE Movie

ADD ImageLink nvarchar(2048);

GO

ALTER TABLE Movie

ADD CONSTRAINT UNI\_ImageLink UNIQUE (ImageLink);

--Q2.2

--Populate your tables with some data using the INSERT statement.

--Make sure you have at least 5 tuples per table.

GO

INSERT INTO Movie VALUES

('1', 'Movie A', '2:30:00','Action', 'Nguyen A', '1000000', 'A good movie', 'www.moviea.com/picturea'),

('2', 'Movie B', '1:30:00', 'Adventure', 'Nguyen B', '2000000', 'A good movie', 'www.movieb.com/pictureb'),

('3', 'Movie C', '1:45:00', 'Comedy', 'Nguyen C', '3000000', 'A good movie', 'www.moviec.com/picturec'),

('4', 'Movie D', '2:00:00', 'Crime (Gangster)', 'Nguyen D', '4000000', 'A good movie', 'www.movied.com/pictured'),

('5', 'Movie E', '2:15:00', 'Dramas', 'Nguyen E', '5000000', 'A good movie', 'www.moviee.com/picturee');

GO

INSERT INTO Actor VALUES

('1', 'Tran A', '25', '500', 'Viet Nam'),

('2', 'Tran B', '30', '600', 'USA'),

('3', 'Tran C', '40', '1000', 'UK'),

('4', 'Tran D', '50', '1200', 'Japan'),

('5', 'Tran E', '60', '2000', 'Korea');

GO

INSERT INTO ActedIn VALUES

('1', '1'),

('1', '2'),

('1', '3'),

('1', '4'),

('1', '5'),

('2', '2'),

('2', '3'),

('3', '3'),

('3', '4'),

('3', '5'),

('4', '4'),

('5', '5'),

('5', '3'),

('5', '4');

GO

UPDATE Movie SET Director = 'Nguyen AA' WHERE MovieID = '1';

--Q3.1

--Write a query to retrieve all the data in the Actor table for actors that are older than 50.

GO

SELECT \* FROM Actor

WHERE ActorAge > 50;

--Q3.2

--Write a query to retrieve all actor names

--And average salaries

--From ACTOR

--And sort the results by average salary.

GO

SELECT ActorName, AverageMovieSalary FROM Actor

ORDER BY AverageMovieSalary;

--Q3.3

--Using an actor name from your table,

--Write a query to retrieve the names of all the movies that the actor has acted in.

GO

SELECT MovieName FROM Movie

WHERE MovieID IN (

SELECT MovieID FROM ActedIn

WHERE ActorID = ( SELECT ActorID FROM Actor WHERE ActorName = 'Tran B' )

);

--Q3.4\*

--Write a query to retrieve the names of all the action movies

--That amount of actor be greater than 3

GO

SELECT MovieID, COUNT(ActorID) AS NumActor FROM ActedIn

GROUP BY MovieID

having COUNT(ActorID)>3

select M.MovieName from Movie M

inner join (

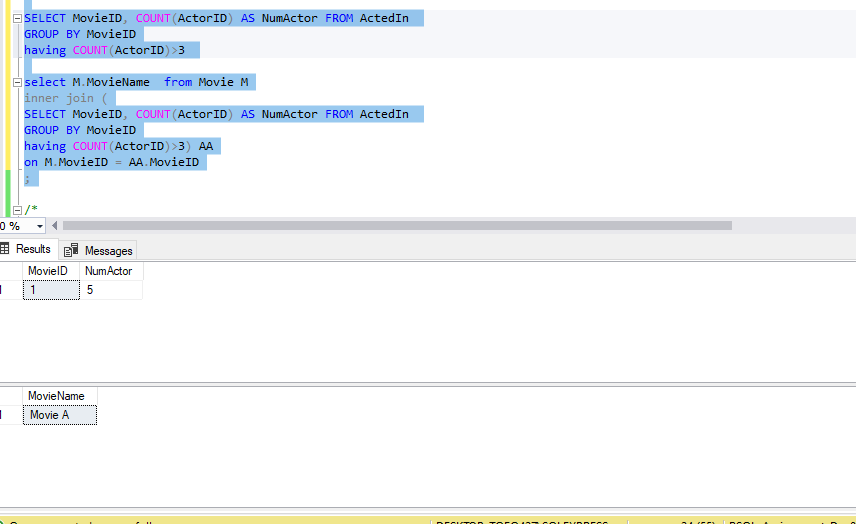
SELECT MovieID, COUNT(ActorID) AS NumActor FROM ActedIn

GROUP BY MovieID

having COUNT(ActorID)>3) AA

on M.MovieID = AA.MovieID

;



**-- THE END --**