

*SQL BASICS*

Training Assignments

|  |  |
| --- | --- |
| Program Code | BSQL |
| Version | 3.1 |
| Effective Date | 01/11/2016 |

**Hanoi, 11/2016**

RECORD OF CHANGES

\*A - Added M - Modified D - Deleted

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Changes | A\* M, D | Contents | Version |
| 14-Oct-2016 | Create | A | Add the new assignments. | v1.0 |
| 14-Oct-2018 | Update | M | Template. | v1.1 |
| 01-Jun-2019 | Update | M | Update Objective | v1.2 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[For the following assignments: 4](#_Toc14121727)

[Day 2. Lesson 2: SQL Basic 4](#_Toc14121728)

[Assignment 2\_Opt3: Movie Management 4](#_Toc14121729)

|  |  |
| --- | --- |
|  | **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.

**-- THE END --**