DATABASE HOMEWORK 1

16th May,2021

AYETIJHYA DESMUKHYA

309127

"I certify that this assignment is entirely my own work, performed independently and without any help from the sources which are not allowed."

Ayetijhya Desmukhya

OVERVIEW

In this project we created a Management System using java servlets using JDeveloper and SQL Database. Our application (containing mainly HTTP servlets) is connected through a database connection to our "MovieSystem" Database which contains the "Movies" (Primary Key -> MovieID) and "Orders" (Primary Key -> Order ID, Foreign Key -> Movies (MovieID)) tables. Database connection is made through DriverManager.getConnection(<proper parameters provided>) function in the servlets

SHORT DESCRIPTION OF THE CLASSES AND SERVLETS

- ### MovieViewer.java

  It is the application to be tested first and connects to all the other servlets (Orders, Filter Movies) through links. MovieViewer.java is a HTTP Servlet which connects to the our "MovieSystem" database and prints out(using PrintWriter) all the entries of dbo.Movies[1] in a table format(using html and a little bit of css).Each entry has a link to rent a movie.

- ### OrderViewer.java

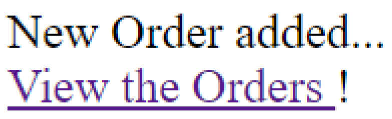
  OrderViewer.java is a HTTP Servlet which connects to the our "MovieSystem" database and prints out (using PrintWriter) all the entries of dbo.Orders[1]] in a table format(using html and a little bit of css).Each entry has a link to update or delete a movie.It also connects to other servlets (Movies, Filter Movies) through links.

- ### FilterForm.html and Filter.java

  FilterForm.html is basically a html file displaying a form containing Release Date From, Release Date To, Genre and Minimum Rating. After filling appropriately and clicking the Submit Button the form gets connected to Filter.java servlet (through action attribute of form).
  Filter.java filters movies using the specifications inputted by the user using SQL query and prints out the filtered result. The result format is similar to MovieViewer.java.

- ### RentMovie.java

  This servlet processes a query to insert orders into the Orders table according to specific instructions given in the task. Upon successful execution:



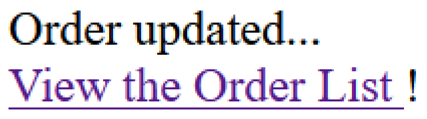
New Order added...
View the Orders !

Otherwise, it prints an error.

[1] The sample entries are in "Insert_File.sql"

- ProcessUpdate.java and UpdateOrder.java

  Similar to the filtering part, process update servlet displays a form containing all the columns of the specific order selected for updating with the values that were already inserted into the columns. After filling appropriate values, clicking the submit button will take us to the UpdateOrder.java servlet through linking which further processes the details as a query and updates the specific entry with the new details. Upon successful execution of update statement:

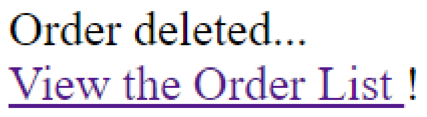


Order updated...
View the Order List !

Otherwise, it prints an error.

- DeleteOrder.java

  This is a self-explanatory servlet since it processes a query to delete a specific order. Upon successful execution of update statement:



Order deleted...
View the Order List !

Otherwise, it prints an error.

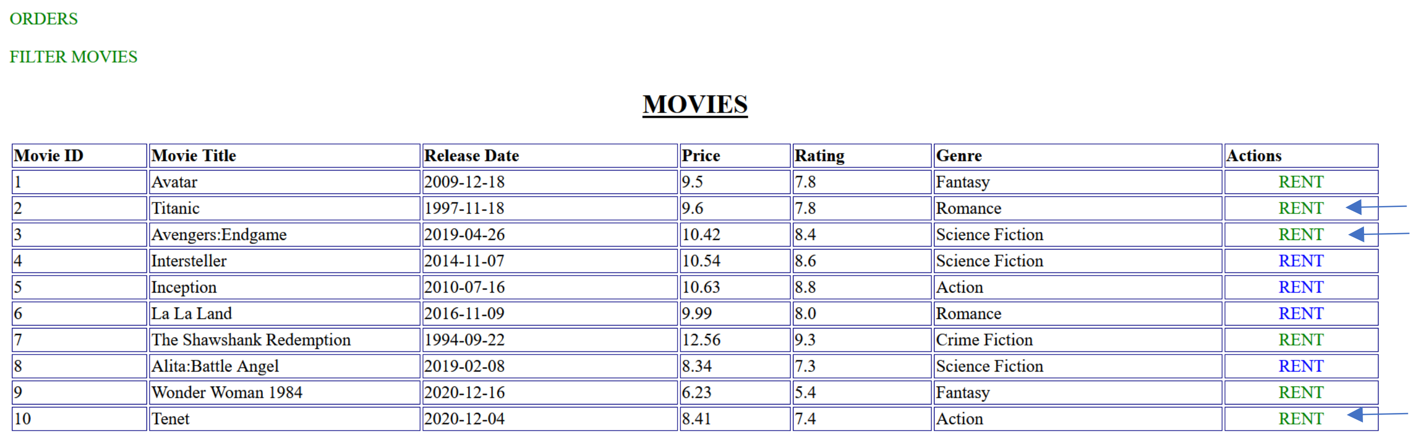
- CommonFunctions.java

  A simple java class for re-using some basic functions required by all the other servlets.
  It contains parseInt(), parseDouble() and parseDate()functions returning the desired type on casting. If it is unsuccessful, it returns an error. On the other hand, there is a getConnection() function which returns Connection con on successfully connecting to the database.

SIMULATION RESULTS

ADDING ORDERS

First, we run the MovieViewer.java servlet and we obtain a screen like this:



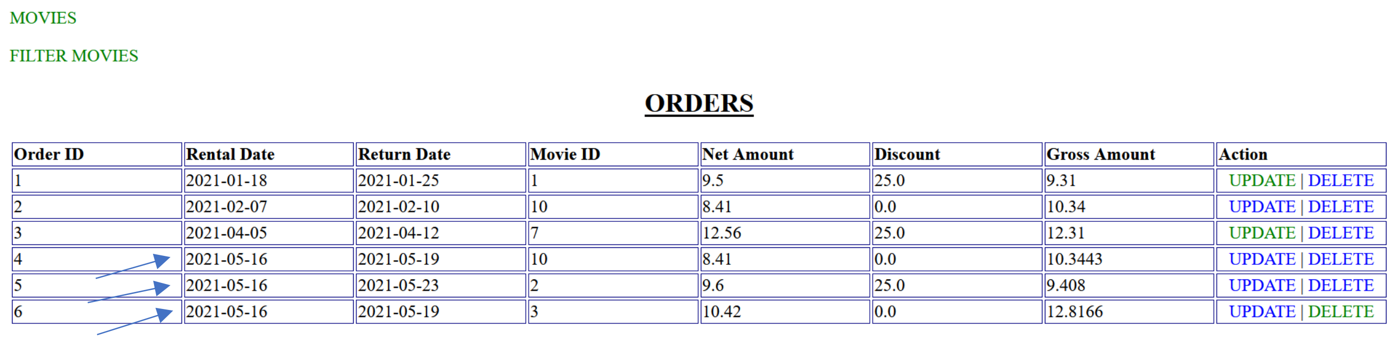
ORDERS

FILTER MOVIES

MOVIES

Movie ID	Movie Title	Release Date	Price	Rating	Genre	Actions
1	Avatar	2009-12-18	9.5	7.8	Fantasy	RENT
2	Titanic	1997-11-18	9.6	7.8	Romance	RENT
3	Avengers:Endgame	2019-04-26	10.42	8.4	Science Fiction	RENT
4	Intersteller	2014-11-07	10.54	8.6	Science Fiction	RENT
5	Inception	2010-07-16	10.63	8.8	Action	RENT
6	La La Land	2016-11-09	9.99	8.0	Romance	RENT
7	The Shawshank Redemption	1994-09-22	12.56	9.3	Crime Fiction	RENT
8	Alita:Battle Angel	2019-02-08	8.34	7.3	Science Fiction	RENT
9	Wonder Woman 1984	2020-12-16	6.23	5.4	Fantasy	RENT
10	Tenet	2020-12-04	8.41	7.4	Action	RENT

Here, when we visit links, the link's text foreground will change to green. So today, I rented 3 more movies as indicated by arrows in the above picture. After getting "Order updated messages" as shown before we move on to see the list of orders:



MOVIES

FILTER MOVIES

ORDERS

Order ID	Rental Date	Return Date	Movie ID	Net Amount	Discount	Gross Amount	Action
1	2021-01-18	2021-01-25	1	9.5	25.0	9.31	UPDATE | DELETE
2	2021-02-07	2021-02-10	10	8.41	0.0	10.34	UPDATE | DELETE
3	2021-04-05	2021-04-12	7	12.56	25.0	12.31	UPDATE | DELETE
4	2021-05-16	2021-05-19	10	8.41	0.0	10.3443	UPDATE | DELETE
5	2021-05-16	2021-05-23	2	9.6	25.0	9.408	UPDATE | DELETE
6	2021-05-16	2021-05-19	3	10.42	0.0	12.8166	UPDATE | DELETE

So, we see the three new orders at the end indicated by arrows. As per our expectations, rental date is today, return date is calculated depending on release date -> 3 days for new releases otherwise 7 days.

E.g., OrderID 5 has duration for 7 days since MovieID 2 -> Titanic was released in 1997 whereas OrderID 4 or 6 have durations for 3 days since the respective movies were released recently and doesn't exceed the 5 years duration.

MovieID, Net Amount are directly taken from the requested movie entry. 25% discount is given to movies with release date older than 5 years from current date. Accordingly, Gross Amount is calculated ->

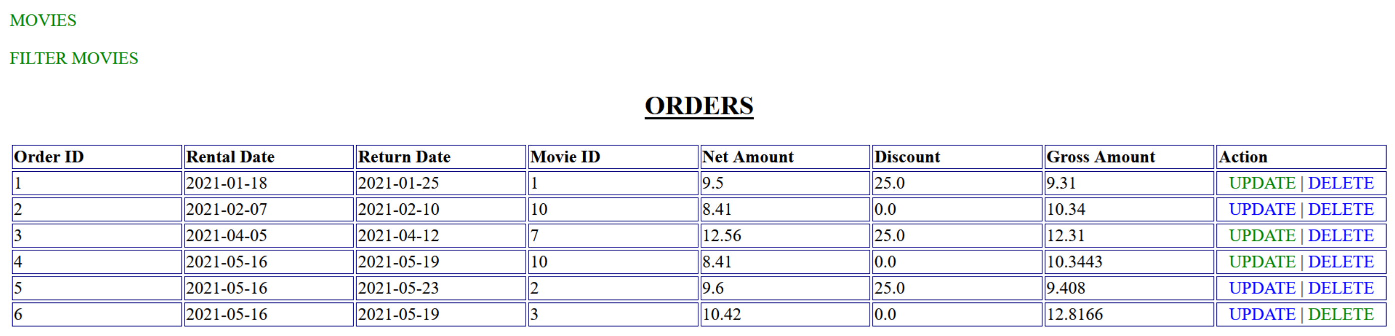
If there is a discount, Gross Amount = Net Amount – Discount + VAT = 0.98*Net Amount

If there is no discount, Gross Amount = Net Amount + VAT = 1.23*Net Amount

UPDATING ORDERS

Now we are going to update 3 orders.

Before updating:



MOVIES

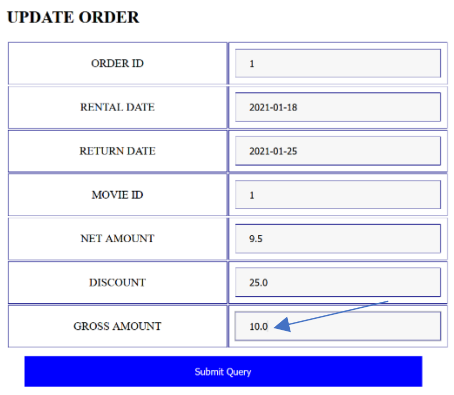
FILTER MOVIES

ORDERS

Order ID	Rental Date	Return Date	Movie ID	Net Amount	Discount	Gross Amount	Action
1	2021-01-18	2021-01-25	1	9.5	25.0	9.31	UPDATE | DELETE
2	2021-02-07	2021-02-10	10	8.41	0.0	10.34	UPDATE | DELETE
3	2021-04-05	2021-04-12	7	12.56	25.0	12.31	UPDATE | DELETE
4	2021-05-16	2021-05-19	10	8.41	0.0	10.3443	UPDATE | DELETE
5	2021-05-16	2021-05-23	2	9.6	25.0	9.408	UPDATE | DELETE
6	2021-05-16	2021-05-19	3	10.42	0.0	12.8166	UPDATE | DELETE

Updating orders:

Updating Order 1

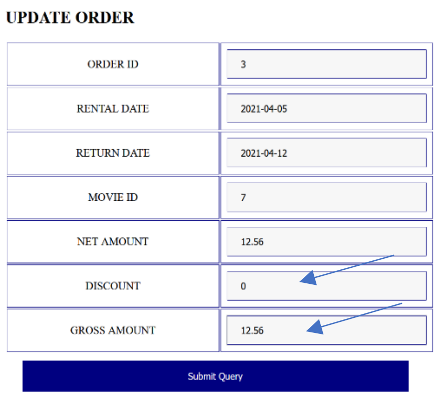


UPDATE ORDER

ORDER ID	1
RENTAL DATE	2021-01-18
RETURN DATE	2021-01-25
MOVIE ID	1
NET AMOUNT	9.5
DISCOUNT	25.0
GROSS AMOUNT	10.0

Submit Query

Updating Order 3

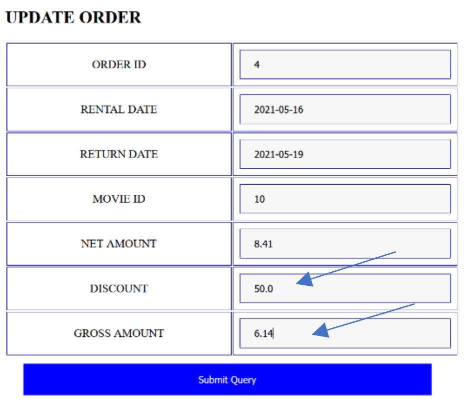


UPDATE ORDER

ORDER ID	3
RENTAL DATE	2021-04-05
RETURN DATE	2021-04-12
MOVIE ID	7
NET AMOUNT	12.56
DISCOUNT	0
GROSS AMOUNT	12.56

Submit Query

Updating Order 4



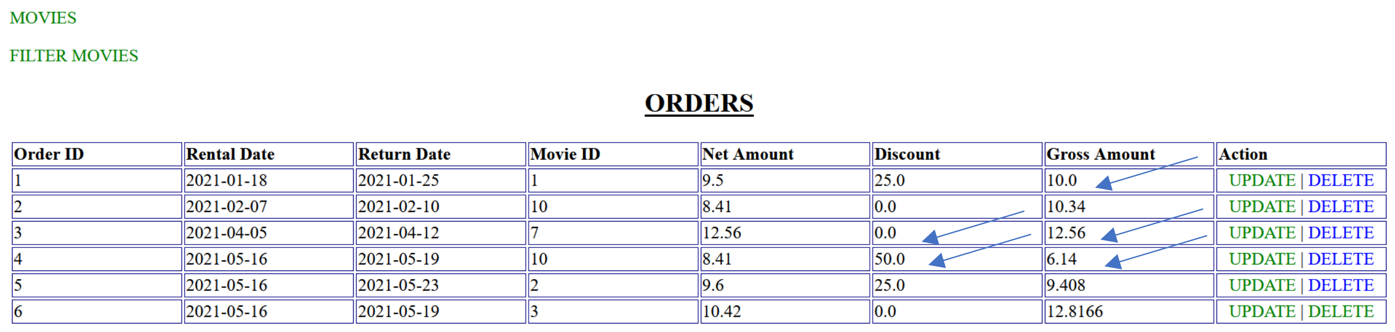
UPDATE ORDER

ORDER ID	4
RENTAL DATE	2021-05-16
RETURN DATE	2021-05-19
MOVIE ID	10
NET AMOUNT	8.41
DISCOUNT	50.0
GROSS AMOUNT	6.14

Submit Query

You might notice a difference in the colour of Submit Query input, that is because the cursor was hovering over the button. This is because I added a hover style property for the input.

Anyway, after updating, we view Orders again to see if they were updated or not:



MOVIES

FILTER MOVIES

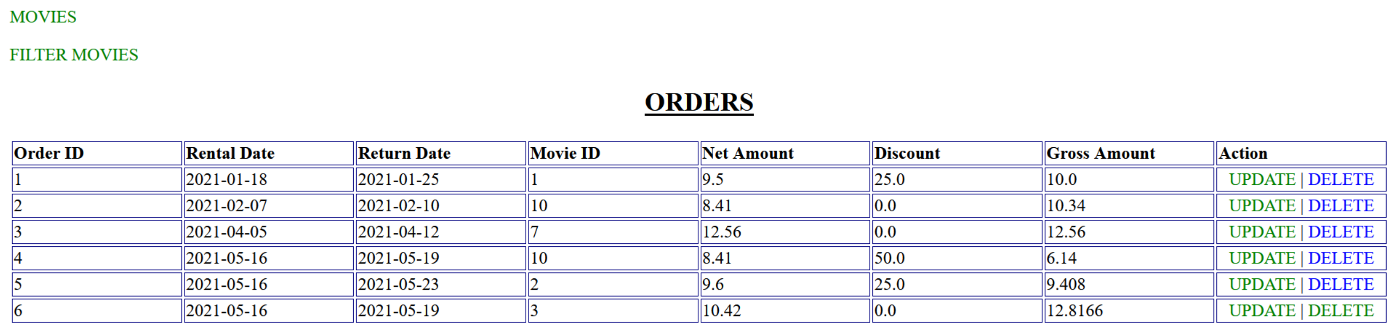
ORDERS

Order ID	Rental Date	Return Date	Movie ID	Net Amount	Discount	Gross Amount	Action
1	2021-01-18	2021-01-25	1	9.5	25.0	10.0	UPDATE | DELETE
2	2021-02-07	2021-02-10	10	8.41	0.0	10.34	UPDATE | DELETE
3	2021-04-05	2021-04-12	7	12.56	0.0	12.56	UPDATE | DELETE
4	2021-05-16	2021-05-19	10	8.41	50.0	6.14	UPDATE | DELETE
5	2021-05-16	2021-05-23	2	9.6	25.0	9.408	UPDATE | DELETE
6	2021-05-16	2021-05-19	3	10.42	0.0	12.8166	UPDATE | DELETE

So, comparing, we see the queries were successfully executed and printed.

DELETING ORDERS

So, before deleting we have this:



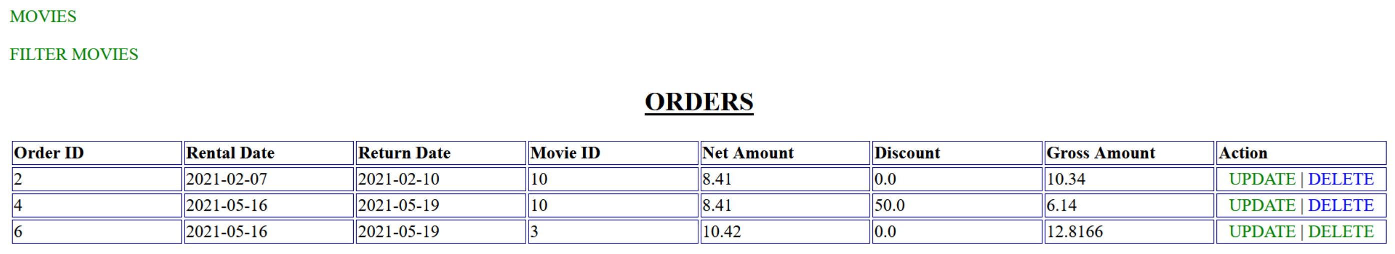
MOVIES

FILTER MOVIES

ORDERS

Order ID	Rental Date	Return Date	Movie ID	Net Amount	Discount	Gross Amount	Action
1	2021-01-18	2021-01-25	1	9.5	25.0	10.0	UPDATE | DELETE
2	2021-02-07	2021-02-10	10	8.41	0.0	10.34	UPDATE | DELETE
3	2021-04-05	2021-04-12	7	12.56	0.0	12.56	UPDATE | DELETE
4	2021-05-16	2021-05-19	10	8.41	50.0	6.14	UPDATE | DELETE
5	2021-05-16	2021-05-23	2	9.6	25.0	9.408	UPDATE | DELETE
6	2021-05-16	2021-05-19	3	10.42	0.0	12.8166	UPDATE | DELETE

After deleting orders, for e.g., 1,3,5, we get:



MOVIES

FILTER MOVIES

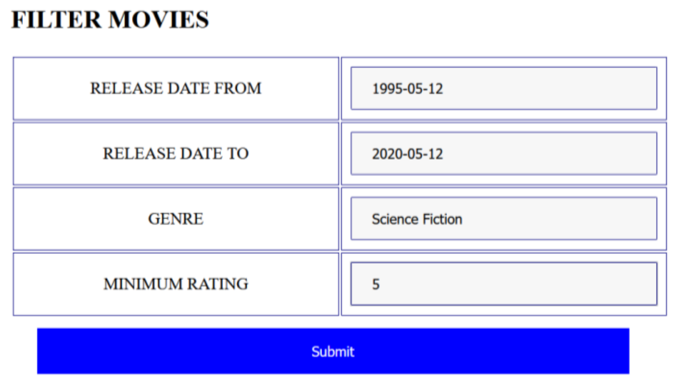
ORDERS

Order ID	Rental Date	Return Date	Movie ID	Net Amount	Discount	Gross Amount	Action
2	2021-02-07	2021-02-10	10	8.41	0.0	10.34	UPDATE | DELETE
4	2021-05-16	2021-05-19	10	8.41	50.0	6.14	UPDATE | DELETE
6	2021-05-16	2021-05-19	3	10.42	0.0	12.8166	UPDATE | DELETE

So, we achieved the expected result. [2]

FILTERING MOVIES

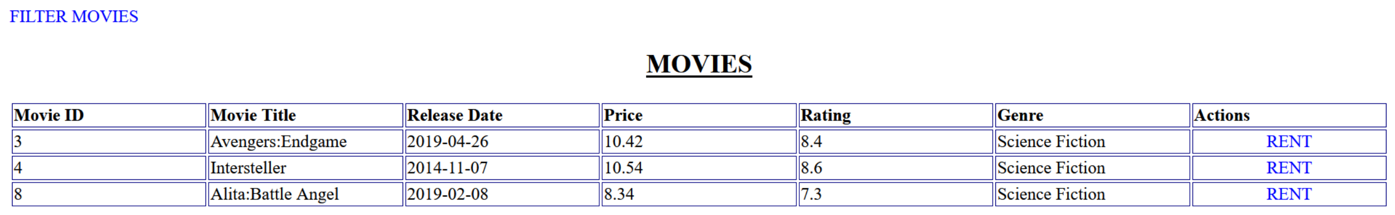
Lastly, on clicking the "FILTER MOVIES" link, we get something like this and then we add details to it:



FILTER MOVIES

RELEASE DATE FROM	1995-05-12
RELEASE DATE TO	2020-05-12
GENRE	Science Fiction
MINIMUM RATING	5

Clicking Submit we get the movies filtered according to the details entered:



FILTER MOVIES

MOVIES

Movie ID	Movie Title	Release Date	Price	Rating	Genre	Actions
3	Avengers:Endgame	2019-04-26	10.42	8.4	Science Fiction	RENT
4	Intersteller	2014-11-07	10.54	8.6	Science Fiction	RENT
8	Alita:Battle Angel	2019-02-08	8.34	7.3	Science Fiction	RENT

[2]After every delete there will be a "Order deleted"message printed showing whether the query was successfully executed or not.