### Saskatchewan Polytechnic

Moose Jaw, Saskatchewan

Information and Communications Technology

# **COMP 234 Database**

# **Final Examination**

Date: April 18th, 2019

Time: 3 Hours

Instructor: M. Barclay

Name: Oluwabunmi Olanrewaju

**INSTRUCTIONS:**

1. PUT YOUR NAME ON THIS EXAM AND SAVE THIS DOCUMENT TO YOUR NETWORK DRIVE (M:\ drive) NOW.
2. All of the CREATE and INSERT commands for the required tables are on PCCOMMON in the Final Folder in the file *FinalSQL.txt*. Use the commands in this file to create the tables and enter the data that you require for the exam.
3. Using the tables that you created, **answer each of the questions and put the answer or SQL statement into the exam**. Do not copy what is retrieved by the command (the output).
4. Use DISTINCT and case conversion (upper( ) and lower( )) functions where appropriate.
5. This is an open-book, on-line exam. No communication with other people is allowed.
6. When completed, submit your exam to your instructor using your SaskPolyTech email. Use the email address: **barclay7342@saskpolytech.ca**

**MARK: \_\_\_\_\_\_\_/50Questions** (marks as indicated)

1. (6 marks) List the names of each member and the total amount they have spent renting videos between October 1, 2016 and today.

Select MEMBER.NAME, SUM(VIDEO.RENTPRICE) from MEMBER

INNER JOIN RENTAL ON MEMBER.MID = RENTAL.MID

INNER JOIN VIDEO ON RENTAL.VID= VIDEO.VID

WHERE RENTAL.DATEOUT BETWEEN '01-OCT-16' AND CURRENT\_DATE

Group by MEMBER.NAME;

1. ( 6 marks) Our company gives a 10% discount on the rental price if the number of days (in whole numbers) elapsed between DateOut and DateIn is less than two days. Write a SQL query to display the member name and discounted rental cost for those videos that receive the discount.

Select unique MEMBER.NAME, SUM(VIDEO.RENTPRICE- (.10\*VIDEO.RENTPRICE)) as "Discounted Cost" from MEMBER

INNER JOIN RENTAL ON MEMBER.MID = RENTAL.MID

INNER JOIN VIDEO ON RENTAL.VID= VIDEO.VID

where RENTAL.DATEIN-RENTAL.DATEOUT < 2

Group by MEMBER.NAME;

1. (5 marks) Write a SELECT statement that displays the following sentence for each rental that was made by our members and returned:

Video “Black Hawk Down” was rented by Sheila Hasbro on October 10th, 2016 at the cost of $6.50.

SELECT ('Video "') || VIDEO.TITLE ||'" was rented by '|| MEMBER.NAME ||'on '||

TO\_CHAR (DATEOUT, 'FMMonth ddth, YYYY ') ||'at the cost of $'|| VIDEO.RENTPRICE

||('.') as "Returned Boooks" from MEMBER

INNER JOIN RENTAL ON MEMBER.MID = RENTAL.MID

INNER JOIN VIDEO ON RENTAL.VID= VIDEO.VID

where DATEIN is not Null;

1. (6 marks) List each video title along with the total rental revenue generated by that title.

Select VIDEO.TITLE, sum(VIDEO.RENTPRICE)as "Total Rental Revenue" from VIDEO

Group by VIDEO.TITLE;

1. (6 marks) For each employee show separately his or her name and the total value of video sales generated by that employee.

Select substr(NAME, 0, InStr(NAME, ' ')) as "First Name",

substr(NAME, InStr(NAME, ' ')) as "Last Name", sum(VIDEO.SALEPRICE) as "video sales"

from EMPLOYEES

INNER JOIN SALE ON EMPLOYEES.EID = SALE.EID

INNER JOIN VIDEO ON SALE.VID= VIDEO.VID

Group by EMPLOYEES.NAME;

1. (6 marks) Write a SQL query to display the member Id and the member name of members located in Saskatchewan, and the video Id and title of videos which are a type of drama (remember your wildcards).

Select MID AS "ID(MID OR VID)", NAME AS "Video or Member Name" from MEMBER

where PROV = 'SK'

UNION Select VID, TITLE from VIDEO

WHERE CATEGORY LIKE '%Drama%' OR CATEGORY LIKE 'Drama%'

OR CATEGORY LIKE '%Drama';

**SECTION 2 PL/SQL** (Please answer in the space provided. Marks are indicated next to each question.)

1. Write a trigger that will update the Video table when a new record is inserted into the Sale table. When a sale is made, a new record is inserted into the Sale table. When that occurs, your trigger should subtract one from the quantity on hand in the Video table, the ***QOH****,* to reflect the fact that a video has been sold. (15 marks)

SET SERVEROUTPUT ON;

CREATE OR REPLACE TRIGGER VIDEO\_TABLE\_UPDATE

After INSERT on SALE

DECLARE

new\_TRANSNO INTEGER;

BEGIN

SELECT count(\*) into new\_TRANSNO from SALE;

IF new\_TRANSNO < new\_TRANSNO

then

update VIDEO

set QOH = QOH-1

where 'VIDEO.VID' = Sale(New.VID);

end if;

END;

/

**\*\*\* THIS EXAMINATION ENDS HERE \*\*\***