CSC 352 / 452: Database Programming

Project  
(260 Points)

**Due on Monday, 11/21/2016 at midnight**

**Late projects will not be accepted or graded.**

**You’ll receive no points if your programs do not compile.**

* **Please, remember that this is NOT a group project. So do not consult with anybody in class or outside class.**
* **Please submit a text file containing all your programs to D2L before or on due date.**
* **Please review your file before submitting it to make sure you have the correct one. It is your responsibility to ensure that you upload the correct file. If you submit a blank/wrong file, you will simply receive a grade of zero.**
* **The project is due by 11:59 pm Monday, November 21, 2016. Do not wait until the last day to submit your file! Submit ahead in case of unforeseen circumstances such as a computer crash or illness. There are no exceptions to the deadline.**

Your task is to build a PL/SQL-based application to insert and update records in a video rental store database and generate some reports.

The database consists of only the following essential tables.

* CUSTOMER(**CUSTOMER\_ID,** CUSTOMER\_NAME, EMAIL, PHONE\_NUMBER,

REGISTRATION\_DATE, EXPIRATION\_DATE, LAST\_UPDATE\_DATE);

* VIDEO(**VIDEO\_ID**, VIDEO\_NAME, FORMAT, PUBLISH\_DATE,

MAXIMUM\_RENTAL\_DAYS);

* VIDEO\_COPY(**COPY\_ID**, VIDEO\_ID\*, COPY\_STATUS, NOTE);
* RENTAL\_HISTORY (**CUSTOMER\_ID\*, COPY\_ID\*, CHECKOUT\_DATE,**

DUE\_DATE, RETURN\_DATE);

The primary keys are marked in red and the foreign keys are marked with asterisks.

VIDEO\_COPY(COPY\_STATUS): A – Available, R – Rented, D – Damaged

Each video in the VIDEO table has at least one video copy in the VIDEO\_COPY table.

**Part 1)**

Create and populate the CUSTOMER, VIDEO, VIDEO\_COPY, and RENTAL\_HISTORY tables by using the following SQL statements.

CREATE TABLE customer

( CUSTOMER\_ID NUMBER PRIMARY KEY,

CUSTOMER\_NAME VARCHAR2(30) NOT NULL,

EMAIL VARCHAR2(50) NOT NULL,

PHONE\_NUMBER VARCHAR2(15) NOT NULL,

REGISTRATION\_DATE DATE NOT NULL,

EXPIRATION\_DATE DATE NOT NULL,

LAST\_UPDATE\_DATE DATE NOT NULL);

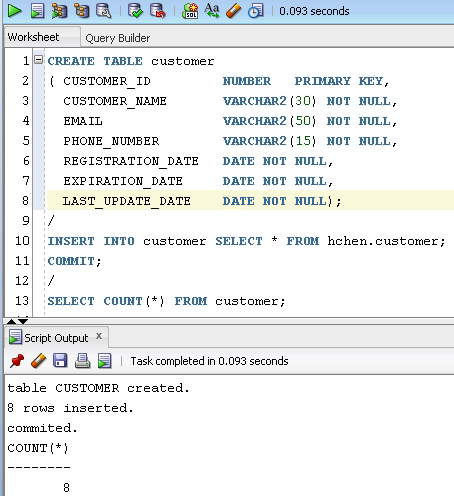
/

INSERT INTO customer SELECT \* FROM **hchen**.customer;

COMMIT;

/

SELECT COUNT(\*) FROM customer;



Please make sure that there are 8 rows in your customer table.

CREATE TABLE video

( VIDEO\_ID NUMBER PRIMARY KEY,

VIDEO\_NAME VARCHAR2(100) NOT NULL,

FORMAT VARCHAR2(100) NOT NULL,

PUBLISH\_DATE DATE NOT NULL,

MAXIMUM\_RENTAL\_DAYS NUMBER(3) NOT NULL

);

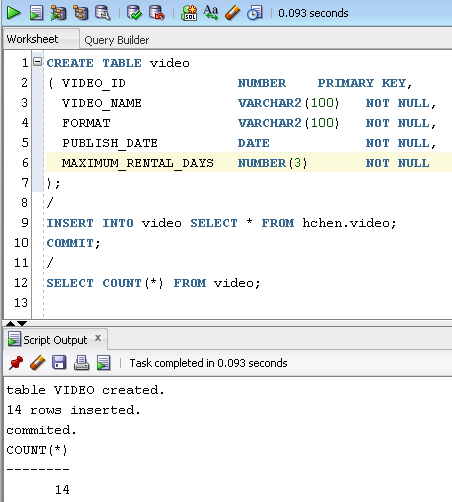
/

INSERT INTO video SELECT \* FROM **hchen**.video;

COMMIT;

/

SELECT COUNT(\*) FROM video;



Please make sure that there are 14 rows in your video table.

CREATE TABLE video\_copy

( COPY\_ID NUMBER PRIMARY KEY,

VIDEO\_ID NUMBER NOT NULL REFERENCES VIDEO (VIDEO\_ID),

COPY\_STATUS CHAR NOT NULL CONSTRAINT ck\_item

CHECK (COPY\_STATUS in ('A', 'R', 'D')),

NOTE VARCHAR2(200));

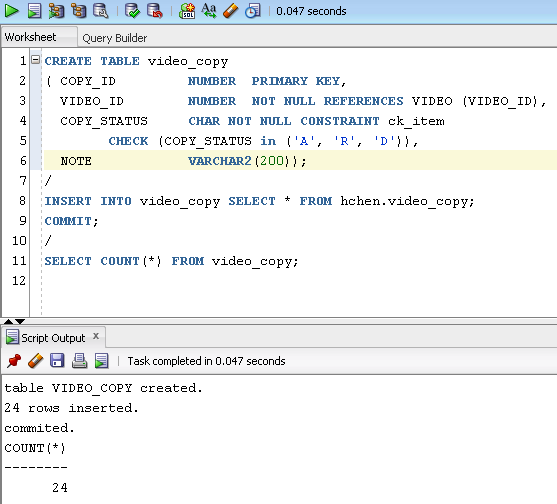
/

INSERT INTO video\_copy SELECT \* FROM **hchen**.video\_copy;

COMMIT;

/

SELECT COUNT(\*) FROM video\_copy;



Please make sure that there are 24 rows in your video\_copy table.

CREATE TABLE rental\_history

( CUSTOMER\_ID NUMBER REFERENCES CUSTOMER (CUSTOMER\_ID),

COPY\_ID NUMBER REFERENCES VIDEO\_COPY (COPY\_ID),

CHECKOUT\_DATE DATE NOT NULL,

DUE\_DATE DATE NOT NULL,

RETURN\_DATE DATE,

NOTE VARCHAR2(200),

CONSTRAINT pk\_rental PRIMARY KEY

(CUSTOMER\_ID, COPY\_ID, CHECKOUT\_DATE));

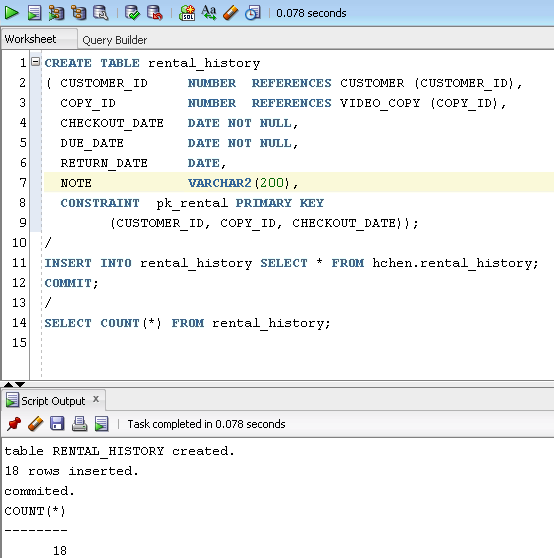
/

INSERT INTO rental\_history SELECT \* FROM **hchen**.rental\_history;

COMMIT;

/

SELECT COUNT(\*) FROM rental\_history;



Please make sure that there are 18 rows in your video\_copy table.

**Part 2) [260 Points]**

* You are not allowed to create temporary tables/views/triggers.
* You can only use the CUSTOMER, VIDEO, VIDEO\_COPY, and RENTAL\_HISTORY tables. You will get a zero point if you use a different table (e.g., different table names, column names, or data types).
* You cannot change the procedure headers. You will get a zero point if you use a different procedure header (e.g., different procedure names, parameter names, data types, or default values).

1) [0 Point] **– new\_customer\_registration()**

Create a procedure called new\_customer\_registrationto add a new customer to the CUSTOMER table.

**(You may use my example in your project.)**

The procedure header is

CREATE OR REPLACE PROCEDURE new\_customer\_registration

(

in\_customer\_id NUMBER,

in\_customer\_name VARCHAR2,

in\_email VARCHAR2,

in\_phone\_number VARCHAR2,

in\_registration\_date DATE,

in\_expiration\_date DATE

)

Consider the following special cases:

* The in\_customer\_id is invalid.
* The in\_customer\_name is empty.
* The in\_email is empty.
* The value of in\_registration\_date is greater than the current date.
* The value of in\_registration\_date is greater than the value of in\_expiration\_date.

**Example**

***CREATE OR REPLACE PROCEDURE new\_customer\_registration***

***(***

***in\_customer\_id NUMBER,***

***in\_customer\_name VARCHAR2,***

***in\_email VARCHAR2,***

***in\_phone\_number VARCHAR2,***

***in\_registration\_date DATE,***

***in\_expiration\_date DATE)***

***IS***

***v\_count NUMBER;***

***BEGIN***

***IF in\_customer\_id <= 0 THEN***

***DBMS\_OUTPUT.PUT\_LINE('Invalid customer ID!');***

***RETURN;***

***END IF;***

***SELECT COUNT(\*)***

***INTO v\_count***

***FROM customer***

***WHERE customer\_id = in\_customer\_id;***

***IF v\_count != 0 THEN***

***DBMS\_OUTPUT.PUT\_LINE('Invalid customer ID!');***

***RETURN;***

***END IF;***

***IF in\_customer\_name is NULL THEN***

***DBMS\_OUTPUT.PUT\_LINE('Invalid customer name!');***

***RETURN;***

***END IF;***

***IF in\_email is NULL THEN***

***DBMS\_OUTPUT.PUT\_LINE('Invalid email address!');***

***RETURN;***

***END IF;***

***IF in\_registration\_date IS NULL OR***

***TO\_CHAR(in\_registration\_date, 'yyyymmdd') >***

***TO\_CHAR(sysdate, 'yyyymmdd') THEN***

***DBMS\_OUTPUT.PUT\_LINE('Invalid registration date!');***

***RETURN;***

***END IF;***

***IF in\_expiration\_date IS NULL OR***

***TO\_CHAR(in\_expiration\_date, 'yyyymmdd') <***

***TO\_CHAR(in\_registration\_date, 'yyyymmdd') THEN***

***DBMS\_OUTPUT.PUT\_LINE('Invalid expiration date!');***

***RETURN;***

***END IF;***

***INSERT INTO customer***

***VALUES(in\_customer\_id, UPPER(in\_customer\_name), in\_email,***

***in\_phone\_number, in\_registration\_date, in\_expiration\_date,***

***sysdate);***

***COMMIT;***

***DBMS\_OUTPUT.PUT\_LINE***

***(INITCAP(in\_customer\_name) ||***

***' has been added into the customer table.');***

***EXCEPTION***

***WHEN OTHERS THEN***

***DBMS\_OUTPUT.PUT\_LINE('My exception: ' ||***

***TO\_CHAR(SQLCODE) || ' ' || SQLERRM);***

***END;***

**Testing the procedure**

* EXEC new\_customer\_registration(2009, 'Adams', 'adams\_1@yahoo.com', '3123621111', '02-OCT-2016', '01-OCT-2018')

Dbms Output: Adams has been added into the customer table.

* EXEC new\_customer\_registration(2010, '', 'ford1@yahoo.com',

'3123622222', '02-OCT-2016', '01-OCT-2018')

Dbms Output: Invalid customer name!

* ……

2) [CSC 352 - 25 points | CSC 452 – 15 points] **– reset\_expiration\_date()**

Create a procedure called reset\_expiration\_date to reset an existing customer’s expiration date.

The procedure header is

CREATE OR REPLACE PROCEDURE reset\_expiration\_date

(

in\_customer\_id NUMBER,

in\_new\_expiration\_date DATE

)

You need to consider the following cases:

* The value of in\_customer\_id is not in the CUSTOMER\_ID column of the CUSTOMER table.

Dbms Output: Invalid customer ID!

* The value of in\_customer\_id is in the CUSTOMER\_ID column of the CUSTOMER table.
* Update the CUSTOMER table (the EXPIRATION\_DATE and LAST\_UPDATE\_DATE columns).
* Dbms Output: The expiration date has been updated.

3) [CSC 352 – 90 points | CSC 452 - 70 points] **– video\_search()**

Create a procedure called video\_search to search a video and display the name, copy ID, format, and status of the video’s copies. In addition, the checkout dates and due dates are also displayed for unreturned copies. The damaged copies (COPY\_STATUS = 'D') are excluded in your output. Sort your output by the video name (NAME) and then the copy ID (COPY\_ID).

The video name is not case-sensitive (e.g., Ocean = ocean). The format of a video is not case-sensitive (e.g., DVD = dvd).

The procedure header is

CREATE OR REPLACE PROCEDURE video\_search

(

in\_video\_name VARCHAR2,

in\_format VARCHAR2 DEFAULT NULL

)

Hint: WHERE UPPER(video\_name) like '%' || UPPER(in\_video\_name) || '%';

**Testing the procedure**

(If your output does not match mine EXACTLY, you will lose some points.)

* EXEC video\_search('ocean')

Dbms Output:

\*\*\*\*\* 0 results found for ocean. \*\*\*\*\*

* EXEC video\_search('PRETTY WOMAN', 'Blu-Ray')

Dbms Output:

\*\*\*\*\* 0 results found for PRETTY WOMAN (Blu-Ray). \*\*\*\*\*

* EXEC video\_search('Pretty Woman')

Dbms Output:

\*\*\*\*\* 3 results found for Pretty Woman. (Available copies: 3) \*\*\*\*\*

VIDEO NAME COPY ID FORMAT COPY STATUS CHECKOUT DATE DUE DATE

---------------------------------------------------------------------------------------------

PRETTY WOMAN 6000 VHS TAPE Available

PRETTY WOMAN 6001 VHS TAPE Available

PRETTY WOMAN 6015 DVD Available

* EXEC video\_search('Another')

Dbms Output:

\*\*\*\*\* 4 results found for Another. (Available copies: 2) \*\*\*\*\*

VIDEO NAME COPY ID FORMAT COPY STATUS CHECKOUT DATE DUE DATE

---------------------------------------------------------------------------------------------

DIE ANOTHER DAY 6010 VHS TAPE Available

DIE ANOTHER DAY 6011 VHS TAPE Rented 20-APR-2016 04-MAY-2016

DIE ANOTHER DAY 6014 DVD Available

DIE ANOTHER DAY 6016 BLU-RAY Rented 01-OCT-2016 04-OCT-2016

* EXEC video\_search('ANOTHER', 'Dvd')

Dbms Output:

\*\*\*\*\* 1 result found for ANOTHER (Dvd). (Available copies: 1) \*\*\*\*\*

VIDEO NAME COPY ID FORMAT COPY STATUS CHECKOUT DATE DUE DATE

---------------------------------------------------------------------------------------------

DIE ANOTHER DAY 6014 DVD Available

* EXEC video\_search('Story')

Dbms Output:

\*\*\*\*\* 7 results found for Story. (Available copies: 4) \*\*\*\*\*

VIDEO NAME COPY ID FORMAT COPY STATUS CHECKOUT DATE DUE DATE

---------------------------------------------------------------------------------------------

TOY STORY 6002 VHS TAPE Rented 09-APR-2016 23-APR-2016

TOY STORY 6003 VHS TAPE Available

TOY STORY 6017 DVD Rented 01-OCT-2016 08-OCT-2016

TOY STORY 2 6009 VHS TAPE Available

TOY STORY 2 6018 DVD Rented 28-APR-2016 05-MAY-2016

TOY STORY 2 6019 DVD Available

TOY STORY 2 6020 BLU-RAY Available

4) [CSC 352 – 75 points | CSC 452 - 65 Points] **–** **video\_checkout()**

Create a procedure called video\_checkout to record a new rental. When the video is successfully checked out, you need to insert a new row (record) into the RENTAL\_HISTORY table and update the corresponding row (record) in the VIDEO\_COPY table. Otherwise, the action is denied.

The procedure header is

CREATE OR REPLACE PROCEDURE video\_checkout

(

in\_customer\_id NUMBER,

in\_copy\_id NUMBER,

in\_video\_checkout\_date DATE

)

A customer whose expiration date is less than the current date (sysdate) is not able to make a rental.

Consider the following special cases:

* The value of in\_customer\_id is not in the CUSTOMER\_ID column of the CUSTOMER table.
* The customer’s expiration date is less than the current date.
* The copy is not available (COPY\_STATUS = 'R' or 'D').
* The value of in\_video\_checkout\_date is greater than the current date.
* Checkout periods are determined by the values in the MAXIMUM\_RENTAL\_DAYS column. The due date is in\_video\_checkout\_date plus the corresponding MAXIMUM\_RENTAL\_DAYS. Hard coding (e.g., 3, 7, or 14) is not allowed.
* A customer may have up to **eight (8)** copies checked out at any one time.

(For example, Tom has eight copies checked out; he cannot rent a copy before he returns one of the eight copies he checked out.)

* (**CSC 452 only**) Before a customer returns a copy, he/she cannot rent a second copy of the same video (**VIDEO\_ID**).

(For example, Mary has the copy with ID 6000 (COPY\_ID = **6000**) checked out. She cannot rent the copy with ID 6001 (COPY\_ID = **6001**) before she returns the copy with ID 6000. The reason is that both copies have the same VIDEO\_ID (VIDEO\_ID = 1000, 'PRETTY WOMAN', 'VHS TAPE')).

**You need to create/run some test cases.**

5) [CSC 352 – 45 points | CSC 452 - 40 points] **–** **video\_return()**

Create a procedure called video\_return to change the rental status for that returned copy. When the copy is successfully checked in, you need to update the corresponding rows (records) in the RENTAL\_HISTORY and VIDEO\_COPY tables. Otherwise, the action is denied.

The procedure header is

CREATE OR REPLACE PROCEDURE video\_return

(

in\_copy\_id NUMBER,

in\_video\_return\_date DATE

)

Consider the following special cases:

* The value of in\_copy\_id does not exist in the corresponding column of the VIDEO\_COPY table.
* The status of that copy is not “R” (COPY\_STATUS != 'R').
* The value of in\_video\_return\_date is greater than the current date.

**You need to create/run some test cases.**

6) [CSC 452 only - 45 points] - **print\_unreturned\_video()**

Create a procedure called print\_unreturned\_video to retrieve all the copies that a customer hasn't returned. The output should include the customer's ID, name, email, phone number, expiration date, first checkout date, last checkout date, the number of unreturned copies, video name, copy ID, format, checkout date, and due date of the rentals. Sort the data by due date and then the video name. The procedure header is

CREATE OR REPLACE PROCEDURE print\_unreturned\_video

(

in\_customer\_id NUMBER

)

**Testing the procedure**

(If your output does not match mine EXACTLY, you will lose some points.)

* EXEC print\_unreturned\_video(90)

Dbms Output:

The customer (id = 90) is not in the customer table.

* EXEC print\_unreturned\_video(2004)

Dbms Output:

----------------------------------------

Customer ID: 2004

Customer Name: JOHNSON

Email: jo12@gmail.com

Phone Number: 7732015678

Expiration Date: 21-APR-2017

First Checkout Date: 01-MAY-2015

Last Checkout Date: 01-MAY-2015

----------------------------------------

Number of Unreturned Videos: 0

----------------------------------------

* EXEC print\_unreturned\_video(2008)

Dbms Output:

----------------------------------------

Customer ID: 2008

Customer Name: SCOTT

Email: scott@hotmail.com

Phone Number: 7701239876

Expiration Date: 30-DEC-2011

First Checkout Date: N/A

Last Checkout Date: N/A

----------------------------------------

Number of Unreturned Videos: 0

----------------------------------------

* EXEC print\_unreturned\_video(2002)

Dbms Output:

----------------------------------------

Customer ID: 2002

Customer Name: JONES

Email: jones@hotmail.com

Phone Number: 6302077890

Expiration Date: 02-MAR-2017

First Checkout Date: 04-MAR-2015

Last Checkout Date: 01-OCT-2016

----------------------------------------

Number of Unreturned Videos: 3

----------------------------------------

Copy ID: 6007

Title: TARZAN

Format: VHS TAPE

Checkout Date: 22-APR-2016

Due Date: 06-MAY-2016

----------------------------------------

Copy ID: 6016

Title: DIE ANOTHER DAY

Format: BLU-RAY

Checkout Date: 01-OCT-2016

Due Date: 04-OCT-2016

----------------------------------------

Copy ID: 6017

Title: TOY STORY

Format: DVD

Checkout Date: 01-OCT-2016

Due Date: 08-OCT-2016

----------------------------------------

7) [CSC 352 - 25 points | CSC 452 – 25 points] **–** Package **video\_rental\_pkg**

Group all the above subprograms (new\_ustomer\_registration, reset\_expiration\_date, video\_search, video\_checkout …) together in a package (package specification and package body) called video\_rental\_pkg.

5 points - package specification

20 points - package body

You need to test your package.

Please submit a text file containing all the source codes to D2L before or on due date.