

**Digital Media and IT**

## Midterm Exam

**October 28th, 2021**

**Intermediate Database Programming**

**DMIT2019**

**STUDENT NAME: Guillermo Moronta Section: A01**

**INSTRUCTOR: J. Henkelman**

**Time Allowed:** 110 minutes

|  |  |  |
| --- | --- | --- |
| Question | Possible Marks | Marks Earned |
| 1 | 8 |  |
| 2 | 10 |  |
| 3 | 10 |  |
| 4 | 4 |  |
| 5 | 4 |  |
| Total | 36 |  |

**Course Weight**: 30 % of Final Mark

**Additional materials permitted:**

All books, **your** notes, and **your** Lab 1

No database connectivity allowed

Answer all questions on the exam document

Questions are based on the database created in Lab 1A

When you are finished, please rename your exam to “Midterm\_LastName\_Firstname.docx” and submit the file to Moodle.

1. Answer the following True/False questions question by typing T or F in the box to the left of the question. **8 marks (½ each)**

|  |  |
| --- | --- |
| F | Precision is the number of digits to the right of the decimal point |
| T | When creating a table, a default may be defined on a column |
| T | The maximum length of a CHAR data type variable is 2000 |
| F | All DML commands are committed when they are executed |
| F | When creating a table, a multi-field Primary Key can be defined at the column level |
| T | A function must return a value |
| F | A procedure may be executed using a select statement |
| T | Every package body must have a matching package specification |
| F | A function within a package may be overloaded by specifying parameters with the same data types |
| T | The following is a valid cursor declaration:  CURSOR C\_Invoice\_Item IS  SELECT SUM (Quantity\_Sold \* Selling\_Price)  FROM Invoice\_Item  WHERE Selling\_Cost < Selling\_Price; |
| T | The %ROWCOUNT cursor attribute returns the number of rows that have been fetched from a cursor so far |
| F | The OTHERS exception must be the only exception in the exception section |
| F | A NO\_DATA\_FOUND exception is raised when a “select…into” statement returns more than one row |
| F | You can only have 12 triggers on a table |
| F | Statement triggers can access both the :OLD and :NEW virtual tables |
| F | Trigger names do not have to be unique |

1. Write a procedure called PR\_EXAM that will recalculate and update the Total\_Price column in the Registration table for all registrations with a registration status of 'O'. The formula used will be the Quantity\_Sold multiplied by the Current\_Price plus GST. The procedure will take an input parameter of the GST rate (inputted as a whole number, i.e. 5). You must use at least one cursor in your solution. **10 Marks**

Create or Replace Procedure PR\_Exam

(P\_GST Number)

As

Cursor C\_update\_total\_price Is Select Invoice.Registration\_Number,

Invoice\_Item.Quantity\_Sold,

Item.Current\_Price

From Invoice, Invoice\_Item, Item

Where Invoice.Invoice\_Number = Invoice\_Item.Invoice\_Number

AND Invoice\_Item.Item\_Number = Item.Item\_Number;

V\_RN Number(6,0); -- Registration Number

V\_QS Number(5,0); -- Quantity Sold

V\_CP Number(5,2); -- Current Price

Begin

Open C\_update\_total\_price;

Fetch C\_update\_total\_price into V\_RN, V\_QS, V\_CP;

While C\_update\_total\_price%Found Loop

Update Registration

Set Total\_Price = V\_QS \* (V\_CP + (V\_CP \* P\_GST))

Where V\_RN = Registration\_Number and

Registration\_Status\_Code = ‘O’;

Fetch C\_update\_total\_price into V\_RN, V\_QS, V\_CP;

End Loop;

Close C\_update\_total\_price;

End PR\_Exam;

/

Show Errors;

1. Write a function called FN\_EXAM that returns a formatted string containing the item description and number, the number of times the item has been sold and the profit when passed an item description. Calculate the total profit using Quantity\_Sold, Selling\_Cost and Selling\_Price and include only those invoices that have been paid. Return descriptive messages if the item description is not unique, the item does not exist, or if the item has never been sold. The case of the input parameter is not guaranteed, and you must use exception handling in your solution. **10 Marks**

Sample call: Select FN\_EXAM ('Hamburger') from dual;

Sample outputs: Hamburger (Item 563) was sold 45 times with a total profit of 250.33

Hamburger does not exist

Hamburger is not unique

Hamburger has never been sold

Create or Replace Function FN\_Exam

(P\_Item\_Description Varchar2)

Return Varchar2

As

V\_Output varchar2(100);

V\_Item\_Description varchar2(45);

V\_Item\_Number Number(4,0);

V\_Quantity\_Sold Number(5,0);

V\_Selling\_Price Number(5,2);

V\_Selling\_Cost Number(5,2);

V\_Profit Number(5,2);

E\_Item\_Never\_Sold Exception;

Begin

Select Item\_Description, Item\_Number

Into V\_Item\_Description, V\_Item\_Number

From Item

Where Item\_Description = P\_Item\_Description;

Select Quantity\_Sold, Selling\_Price, Selling\_Cost

Into V\_Quantity\_Sold, V\_Selling\_Price, V\_Selling\_Cost

From Invoice\_Item

Where Item\_Number = V\_Item\_Number;

If V\_Quantity\_Sold = 0

Raise E\_Item\_Never\_Sold;

Else

-- My hands and brain are too slow, how tragic.

End If;

Exception

When No\_Data\_Found Then

V\_Output := Raise\_Application\_Error(-20010, ‘’ || P\_Item\_Description || ’ does not exist’);

Return V\_Output;

When Too\_Many\_Rows Then

V\_Output := Raise\_Application\_Error(-20011, ‘’ || P\_Item\_Description || ’ is not unique’);

Return V\_Output;

When E\_Item\_Never\_Sold Then

V\_Output := Raise\_Application\_Error(-20012, ‘’ || P\_Item\_Description || ’ has never been sold’);

Return V\_Output;

When Others Then

V\_Output := Raise\_Application\_Error(-20013, ‘Unexpected error occurred when passing item ’ || P\_Item\_Description || );

Return V\_Output;

End FN\_Exam;

/

Show Errors;

1. Write a trigger called TR\_EXAM, which will restrict the user from increasing the current price in the item table by more than 25% at one time. If the increase is greater than 25%, discontinue processing and display an appropriate error message. Ensure the trigger does not fire more often than is necessary. **4 Marks**

Create or Replace Trigger TR\_Exam

Before Update of Current\_Price

On Item

For Each Row

Begin

If :NEW.Current\_Price > (Current\_Price + (:OLD.Current\_Price \* 0.25))

RAISE\_APPLICATION\_ERROR(-20010, ‘Cannot increase the current price by more than 25% at one time’);

End If;

End TR\_Exam;

/

Show Errors;

1. For the events shown below, write the number (in the box to the left) of the order in which the events would occur if the Update statement was executed. If a particular event would not occur, then show it as NE. If the order in which two events will occur is indeterminate, show them both with the same number. (Sample Answer: 2, 3, 4, 3, NE, 1, 2, 5) **4 Marks**

Update Registration

Set Departure\_Date = Arrival\_Date

Where Registration\_Status\_Code = ’C’;

|  |  |
| --- | --- |
| 3 | Execution of trigger created with the following:  CREATE TRIGGER TR\_1  AFTER UPDATE OR DELETE  ON Registration  FOR EACH ROW |
| NE | Execution of trigger created with the following:  CREATE TRIGGER TR\_2  AFTER INSERT  ON Registration |
| 4 | Execution of trigger created with the following:  CREATE TRIGGER TR\_3  AFTER UPDATE OF Arrival\_Date  ON Registration |
| 2 | Virtual tables are populated |
| 4 | Execution of trigger created with the following:  CREATE TRIGGER TR\_4  AFTER UPDATE  ON Registration |
| 1 | Execution of trigger created with the following:  CREATE TRIGGER TR\_5  BEFORE UPDATE  ON Registration |
| NE | Execution of trigger created with the following:  CREATE TRIGGER TR\_6  AFTER UPDATE  ON Registration\_Status  FOR EACH ROW |
| 1 | Execution of trigger created with the following:  CREATE TRIGGER TR\_7  BEFORE INSERT OR UPDATE OR DELETE  ON Registration |

**Appendix 1: Database Structure**

