

**DIGITAL MEDIA & IT**

**Final Exam**

**December 16th, 2021**

**Intermediate Database Programming**

**DMIT2019**

Student Name: Guillermo Moronta Fernandez Section: A01

Instructor: J. Henkelman

Time Allowed: 110 minutes

|  |  |  |
| --- | --- | --- |
| Question | Possible Marks | Marks Earned |
| 1 | 8 |  |
| 2 | 5 |  |
| 3 | 4 |  |
| 4 | 6 |  |
| 5 | 4 |  |
| 6 | 7 |  |
| 7 | 5 |  |
| 8 | 5 |  |
| Total | 44 |  |

Course Weight: 30 % of Final Mark

Number of Pages: 9

Questions are based on the database used in Labs 1 and 2

Answer all questions on this exam document

**Do not leave any property blank – use <EMPTY> instead!**

1. Answer the following True/False questions by typing T or F in the box to the left of the question **½ mark ea.**

|  |  |
| --- | --- |
| T | A data block cannot exist without being displayed on the form |
| T | The Relationships page of the Data Block Wizard is used to define a master detail relationship between two data blocks |
| T | When a form is in Normal mode, you can create new records |
| T | The Update Layout property of a frame can be set to automatically move and reposition the objects within the frame when you move or resize the frame |
| F | The default form item is a display item |
| F | When you create a List Of Values, a button to open it is automatically created and placed on the form |
| F | Every form object has the same properties in their Property Palette |
| T | A List Item contains hard-coded values |
| T | A Check Box can be used when the valid values for the item are "True" and "False" |
| T | The default radio button is set using the “Initial Value” property of the corresponding Radio Group |
| T | A user cannot alter the data in a display item |
| F | The SQL select statement in a record group may access multiple tables but not contain order by and where clauses |
| F | The Case Restriction property specifies whether the text entered is displayed or hidden |
| F | It is not necessary to recompile the menu module in order to show the most recent menu changes |
| T | A menu item may be hidden |
| F | Cut, Copy and Clear items are examples of a plain menu type |

The following questions relate to a form used to maintain items sold through room service. Snapshots of the layout and object navigator for this form are attached in the Appendices (layout editor – Appendix B, object navigator – Appendix C).

1. We want to automate the generation of item numbers, using the sequence Seq\_Item, when a new item is **saved**. The user **must** be allowed to navigate to the item number field when in data entry mode.
   1. Fill in the following partial property sheet with the correct values for the Item\_Numberfield. **½ mark each**

|  |  |  |
| --- | --- | --- |
| SECTION | PROPERTY | VALUE |
| General | Item Type | **<Text>** |
| Functional | Enabled | **<Yes>** |
| Data | Data Type | **<Number>** |
|  | Required | **<Yes>** |
| Database | Database Item | **<Yes>** |

* 1. Put an X beside the trigger that is **BEST** suited to generating the item number when a new record is **saved**. **½ mark**

|  |  |
| --- | --- |
| X | Pre-Insert on the **Item** Block |
|  | When-Validate-Item on the **Item\_Number** text item |
|  | When-Create-Record on the **Item** Block |

* 1. Write the code for the trigger you selected in the previous question. Ensure that if the field is already filled with a value, you do **NOT** replace the existing value **2 marks**

Select Seq\_Item.nextval

Into :Item.Item\_Number

From Dual;

1. We are finding that the discount field was not being used consistently. To mitigate this problem, we decided to simplify the data and only use four distinct values (0, 10, 25 and 100). A list item is now used to display the discount field. Fill in the following property palettes to ensure a default value of “Ten” and that any non-standard data will appear as “Zero”.
   1. Discount **2 marks**

|  |  |  |
| --- | --- | --- |
| **SECTION** | **PROPERTY** | **VALUE** |
| Functional | Mapping of other values | **0** |
| Data | Data Type | **Number** |
|  | Initial Value | **10** |
| Database | Database Item | **Yes** |

* 1. Elements in List (assume the focus is on the first List Element) **2 marks**

|  |
| --- |
| **List Elements** |
| **10** |
| **25** |
| **100** |
| **0** |
| **<empty>** |
| **<empty>** |
| **List Item Value** |
| **10** |

1. The users have decided that they want to know if an item has been used on any invoice(s). We will add a radio group (**Used\_On\_Invoices**) to the form to indicate this. Fill in the following property palettes to show if there are any invoices using this item.
   1. Used\_On\_Invoices **2 marks**

|  |  |  |
| --- | --- | --- |
| **SECTION** | **PROPERTY** | **VALUE** |
| General | Item Type | **Radio** |
| Data | Data Type | **Char** |
|  | Initial Value | **N** |
| Database | Database Item | **No** |

* 1. Radio\_Button\_Yes **1 mark**

|  |  |  |
| --- | --- | --- |
| **SECTION** | **PROPERTY** | **VALUE** |
| Functional | Label | **Item Used** |
|  | Radio Button Value | **Y** |

* 1. Put an X beside the trigger that is **BEST** suited to appropriately populate the radio buttons when a record is **retrieved**? **½ mark**

|  |  |
| --- | --- |
|  | When-Validate-Item on **Item\_Number** field |
| X | Post-Query on the **Item** Block |
|  | Post-Text-Item on **Item\_Number** field |

* 1. Write the code for the trigger selected in the previous question. **2 ½ marks**

Select Item\_Number as ‘Y’

Into :Item.Used\_On\_Invoices

From Invoice\_Item

Where :Item.Item\_Number = Invoice\_Item.Item\_Number;

1. We want to display the discounted price of each item. To facilitate this, we will create a display field (Current\_Profit) that will display this value (current price minus current cost) for each item.

Fill in the following property sheet for the new display item. **½ mark each**

|  |  |  |
| --- | --- | --- |
| SECTION | PROPERTY | VALUE |
| General | Name | **Current\_Profit** |
| Calculation | Calculation Mode | **Formula** |
|  | Formula | **:Item.Current\_Price - :Item.Current\_Cost** |
|  | Summary Function | **None** |
|  | Summarized Block | **<null>** |
|  | Summarized Item | **<null>** |
| Data | Data Type | **Number** |
| Database | Database Item | **No** |

1. We are using an **informational** alert to notify the user when an item’s current price is less than current cost. The **default** response will be to set the current price to current cost plus 25%. The alternate response will be to display an error message (including the value of the current price) and not continue processing. See Appendix D for a sample output.
   1. Fill in the Property Palette **½ mark each**

|  |  |  |
| --- | --- | --- |
| **SECTION** | **PROPERTY** | **VALUE** |
| Functional | Message | **Your price is less than your cost** |
|  | Alert Style | **Note** |
|  | Button 1 Label | **Do Not Continue** |
|  | Button 2 Label | **Change the price to cost plus 25%** |
|  | Button 3 Label | **<empty>** |
|  | Default Alert Button | **Button 2** |

* 1. Put an X beside the trigger that is **BEST** suited to show the alert when an existing value is **updated?** **½ mark**

|  |  |
| --- | --- |
|  | Post-Text-Item on the **Current\_Cost** item |
| X | When-Validate-Record on the **Item** block |
|  | Post-Text-Item on the **Current\_Price** item |

* 1. Write the code for your trigger **3 ½ marks**

Declare

alert number;

Begin

If :Item.Current\_Price < :Item.Current\_Cost Then

alert := Show\_Alert(‘Item\_Price\_Alert’);

If alert = Alert\_Button1 Then

Message(‘You have chosen to not continue’);

Else

:Item.Current\_Price := :Item.Current\_Cost + (:Item.Current\_Cost \* 0.25);

Message(‘The Current Price has been adjusted to a 25% increase of the Current Cost.’);

End If;

End If;

End;

1. User Interface
   1. The invoice item information will be added in a block called INV\_ITEM on a stacked canvas called INVITEM\_CANVAS. Write the code that will: i) show the canvas to the user and ii) hide the canvas from the user.
      1. When-Button-Pressed trigger code for Hide Button **1 mark**

Go\_Block(‘Inv\_Item’);

Hide\_View(‘InvItem\_Canvas’);

* + 1. When-Button-Pressed trigger code for Show Button **1 mark**

Go\_Block(‘Inv\_Item’);

Show\_View(‘InvItem\_Canvas’);

* 1. Ensure that the close button on the title bar will close the form regardless of what mode the user is in.
     1. Which trigger would **BEST** be used for this **½ mark**

When-Window-Closed-Trigger

* + 1. Write the code for your trigger **1 ½ marks**

If :System.Mode = ‘Enter-Query’ Then

DO\_KEY(‘Exit\_Form’);

End If;

Do\_Key(‘Exit\_Form’);

* 1. If you perform a query and receive an error message and want more information, how do you obtain it within the runtime environment? **1 mark**

Check bottom left of the window for error messages or displayed hints if defined

For greater detail of the error message, do as follows.

Oracle Menu > Help > Display Error

1. Application Integration
   1. Multi-Form Applications (type T or F in the boxes) **½ mark each**

When an Oracle form (form2) is opened from another Oracle form (form1) with the following **Oracle built-in procedures**, form1…

|  |  |  |  |
| --- | --- | --- | --- |
|  | **OPEN\_FORM** | **NEW\_FORM** | **CALL\_FORM** |
| …remains operable | **T** | **F** | **F** |
| …remains open | **T** | **F** | **T** |

* 1. File Types – Fill in the following table: **½ mark each**

|  |  |  |
| --- | --- | --- |
|  | Executable File Extension | Source Code File Extension |
| Menus | **mmx** | **mmb** |
| Forms | **fmx** | **fmb** |