1Z0-151 Q&As

Oracle Fusion Middleware 11g: Build Applications with Oracle Forms

Pass Oracle 1Z0-151 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.geekcert.com/1z0-151.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Oracle Official Exam Center

- **Instant Download** After Purchase
- **100% Money Back** Guarantee
- **365 Days** Free Update
- **800,000+** Satisfied Customers

QUESTION 1

To avoid overloading the database during busy times, you decide to restrict the queries that are executed on the Orders form so that users query by either Order ID or Customer ID during these times. Which trigger is most appropriate for the code to enforce this restriction?

A. When-New-Form-instance

B. When-New-Block-instance

C. On-Query

D. Pre-Query E. Post-Query

Correct Answer: D

About controlling queries with Pre-Query and Post-Query triggers

The Pre-Query and Post-Query triggers allow control over query processing. They can be defined at the form or block level. Most often, attach them to specific blocks to control the query functionality of those blocks.

The Pre-Query trigger fires just before Form Builder issues the SELECT statement to the database, after the operator has defined the example record by entering query criteria in Enter Query mode.

Inside a Pre-Query trigger, the example record defined by the query criteria is the current record. This means that trigger code can read and set the values of items in the example record using standard :block_name.item_name syntax.

A Pre-Query trigger can be used to disallow query conditions that might be invalid. When a form is in Enter Query mode, normal validation is suspended and no validation triggers fire as they do in Normal mode.

The Pre-Query trigger thus allows you to verify that any values entered by the operator are valid query conditions.

When invalid query conditions have been entered, you can abort the query by raising the FORM_TRIGGER_FAILURE built-in exception in the Pre-Query trigger.

You can also call SET_BLOCK_PROPERTY to modify the block\\'s WHERE and ORDER BY clauses from within the Pre-Query trigger, to further restrict or order the records the query will retrieve.

QUESTION 2

Immediately after creating a button in the Layout Editor, what is true about the button?

A. It is an iconic button.

B. It has no functionality.

C. It is not mouse navigable.

D. It is not keyboard navigable.

E. It is in the control block by default.

F. It is not enabled.

Correct Answer: B

QUESTION 3

The Orders form has four blocks. The Orders and Order_items block are on the CV_Order content canvas; the inventories block items are on the CV_inventories content canvas; and Control block buttons are on the CV_Buttons toolbar canvas. All buttons have mouse Navigate set to No.

The Order_Items block is a detail of Orders. The inventories block is a detail of Order_Items, showing the stock of the selected product.

There is a button in the Control block with a When-Button-Pressed trigger:

IF GET_CANVAS_PROPERTY(:SYSTEM.cursor_item, item_canvas) = \\'CV_ORDER\\' THEN GO_BLOCK (\\'inventories\\')

ELSE

GO_BLOCK(\\'orders\\');

END IF;

When you run the form and click the button, navigation does not occur, and the form displays the runtime error "FRM-41053: Cannot find Canvas: invalid ID." What should you do to correct this problem?

A. Change the sequence of blocks in the Object Navigator

B. Chang the Mouse Navigator property of the button to yes

C. in the first line of code, change the built-in to GET_ITEM_PROPERTY

D. in the first line of code, change the system variable to: SYSTEM.CURSOR_CANVAS.

E. in the first line of code, change the CV_ORDER to lowercase

F. Chang the argument to the GO_BLOCK built-ins to uppercase

Correct Answer: D

Note: GET_CANVAS_PROPERTY built-in Description Returns the given canvas property for the given canvas. . Syntax FUNCTION GET_CANVAS_PROPERTY(canvas_id Canvas ,property NUMBER);FUNCTION GET_CANVAS_PROPERTY(canvas_name VARCHAR2 ,property NUMBER)

QUESTION 4

Which statement is true about flexible code?

A. It is designed for reuse.

B. It typically includes hard-coded object names.

C. It is more difficult to maintain.

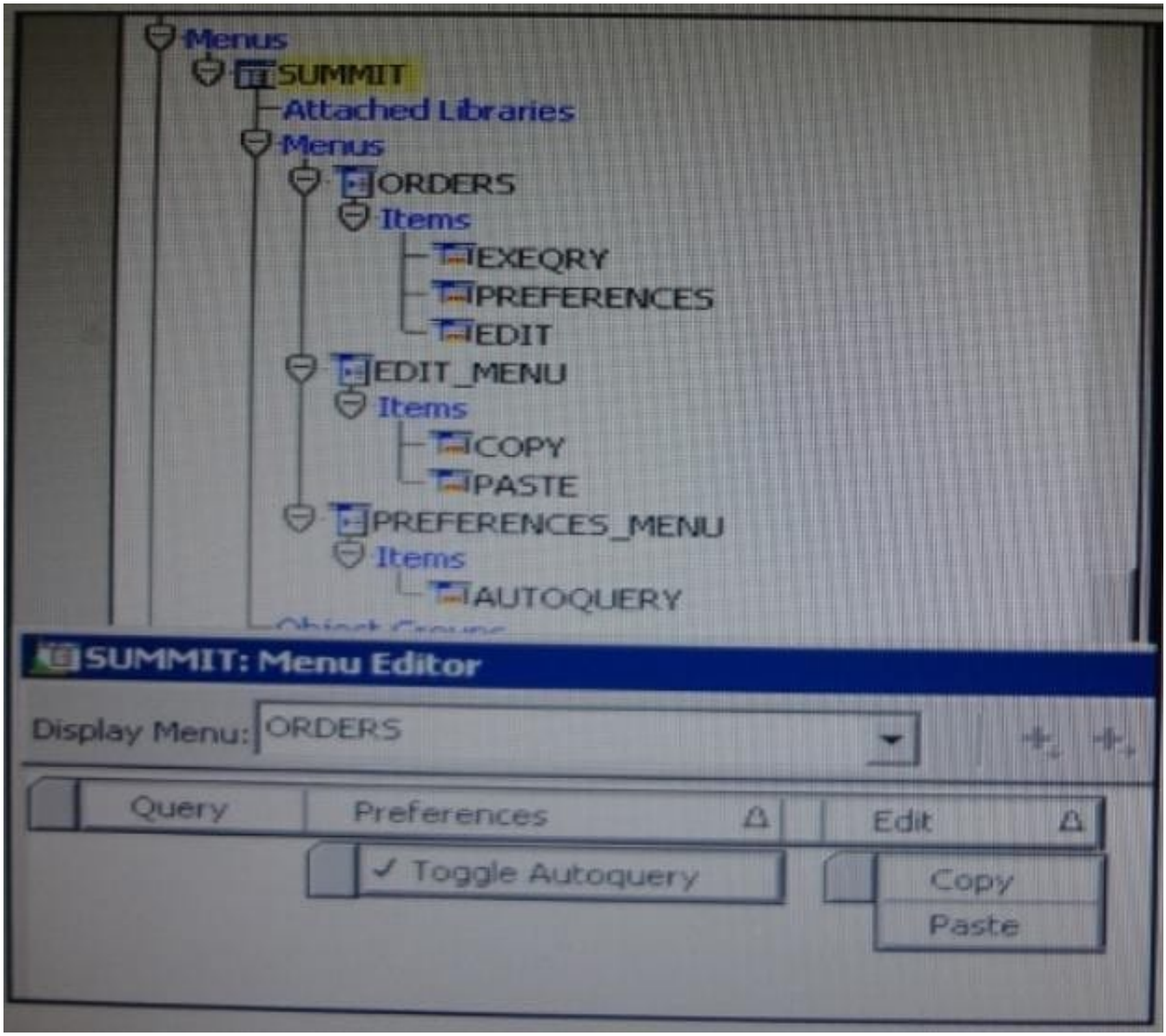
D. It is more difficult to write, so it decreases developer productivity.

E. It is specific to a particular module.

Correct Answer: A

QUESTION 5

View the Exhibit.



The Summit menu is attached to the Orders form. The Toggle Autoquery menu item is a check box that toggles whether a query is automatically performed when the Orders form is first invoked. If the check box is deselected, users must manually query.

In addition to using the menu, users want to be able to toggle the autoquery preference directly from the form. You add a button named Toggle Autoquery with the following When-Button- Pressed trigger:

DECLARE

mi_id MENUITEMS;

BEGIN

mi_id ;=FIND_ITEM (\\'Preferences.AutoQuery\\')

/* Determine the current checked static of the AutoCommit menu checkbox item And toggle the checked

state*/

IF GET_ITEM_PROPERTY (mi_id, CHECKED) = \\'TRUE\\' THEN

SET_ITEM_PROPERTY (mi_id, CHECKED, PROPERTY_FALSE);

ELSE

SET_ITEM_PROPERTY (mi_id, CHECKED, PROPERTY_TRUE);

END IF;

END;

However, the trigger does not compile. What three changes must you make so that the trigger compiles

successfully?

A. Change FIND_ITEM to FIND_MENU_ITEM.

B. Change \\'preferences.AutoQuery\\' to \\'orders.preferences.AutoQuery\\'.

C. Change \\'preferences.AutoQuery\\' to \\'AutoQuery\\'.

D. Change \\'preferences.AutoQuery\\' to \\'ORDERS.PREFERENCES>AUTOQUERY\\'.

E. Change \\'preferences.AutoQuery\\' to \\'AUTOQUERY\\'.

F. Change GET_ITEM_PROPERTY to GET_MENU_ITEM_PROPERTY

G. Change SET_ITEM_PROPERTY to SET_MENU_ITEM_PROPERTY

H. Change PROPERTY_FALSE to \\'FALSE\\'.

I. Change PROPERTY_TRUE to \\'TRUE\\'.

Correct Answer: AFG

A: Note: FIND_MENU_ITEM built-in Description Searches the list of menu items and returns a menu item ID when it finds a valid menu item with the given name. You must define an appropriately typed variable to accept the return value. Define the variable with a type of MenuItem.

Note 2:

FIND_ITEM built-in

Description

Searches the list of items in a given block and returns an item ID when it finds a valid item with the given

name. You must define an appropriately typed variable to accept the return value. Define the variable

with a type of Item.

Example (with FIND_MENU_ITEM, GET_MENU_ITEM_PROPERTY,

SET_MENU_ITEM_PROPERTY)

FIND_MENU_ITEM examples

/*

** Built-in: FIND_MENU_ITEM

** Example: Find the id of a menu item before setting

** multiple properties

*/

PROCEDURE Toggle_AutoCommit_Mode IS

mi_id MenuItem;

val VARCHAR2(10);

BEGIN

mi_id := Find_Menu_Item(\\'Preferences.AutoCommit\\');

/*

** Determine the current checked state of the AutoCommit ** menu checkbox item

*/

val := Get_Menu_Item_Property(mi_id,CHECKED);

/*

** Toggle the checked state

*/

IF val = \\'TRUE\\' THEN

Set_Menu_Item_Property(mi_id,CHECKED,PROPERTY_FALSE);

ELSE

Set_Menu_Item_Property(mi_id,CHECKED,PROPERTY_TRUE);

END IF;

END;