Assignment 3-5 Using a WHILE loop

Brewbean's wants to include a feature in their application that calculates the total number (quantity) of a specified item that can be purchased with a given amount of money. Create a block using a WHILE loop to increment the cost of the item until the dollar value is met. Test first with a total spending amount of $100 and product id 4. Then test with an amount and product of your choice. Use host variables to provide the total spending amount and product id to the block.

|  |
| --- |
| DECLARE  total\_spent NUMBER := 100; -- Host variable for total spending amount  product\_id NUMBER := 4; -- Host variable for the product ID  price NUMBER; -- Variable to store the price of the selected product  items\_purchased NUMBER := 0; -- Counter for the number of items that can be bought  BEGIN  -- Fetch the price of the product using the provided product\_id  SELECT price INTO price  FROM Products  WHERE product\_id = product\_id;    -- Ensure we found the product and have a valid price  IF price IS NULL THEN  DBMS\_OUTPUT.PUT\_LINE('Product not found');  RETURN;  END IF;    -- Use a WHILE loop to calculate how many items can be bought  WHILE total\_spent >= price LOOP  total\_spent := total\_spent - price; -- Deduct the price of one item from total\_spent  items\_purchased := items\_purchased + 1; -- Increment the item count  END LOOP;    -- Output the result  DBMS\_OUTPUT.PUT\_LINE('Items that can be purchased: ' || items\_purchased);  DBMS\_OUTPUT.PUT\_LINE('Remaining money: $' || total\_spent);    END; |

Assignment 3-6 Working with IF statements

Brewbean's calculates shipping cost based on the quantity of items in an order. Assume the quantity column in the bb\_basket table contains the total number of items in a basket. A block is needed to check the quantity provided by a host variable and determine the shipping cost. Display the determined shipping cost on the screen. Test using the basket ids of 5 and 12. Apply the shipping rates as listed in Table 3-3.

|  |  |
| --- | --- |
| Quantity of Items | Shipping Cost |
| Up to 3 | $5.00 |
| 4-6 | $7.50 |
| 7-10 | $10.00 |
| Over 10 | $12.00 |

|  |
| --- |
| DECLARE  basket\_id NUMBER := 5; -- Host variable for basket\_id, change for testing with other basket IDs (e.g., 12)  quantity NUMBER; -- Variable to store the quantity of items in the basket  shipping\_cost NUMBER; -- Variable to store the calculated shipping cost  BEGIN  -- Retrieve the quantity of items in the basket using the basket\_id  SELECT quantity INTO quantity  FROM bb\_basket  WHERE basket\_id = basket\_id;    -- Check if quantity is found  IF quantity IS NULL THEN  DBMS\_OUTPUT.PUT\_LINE('Basket not found');  RETURN;  END IF;    -- Determine the shipping cost based on the quantity using IF-ELSE or CASE statement  IF quantity <= 3 THEN  shipping\_cost := 5.00;  ELSIF quantity BETWEEN 4 AND 6 THEN  shipping\_cost := 7.50;  ELSIF quantity BETWEEN 7 AND 10 THEN  shipping\_cost := 10.00;  ELSE  shipping\_cost := 12.00;  END IF;    -- Display the shipping cost  DBMS\_OUTPUT.PUT\_LINE('Basket ID: ' || basket\_id);  DBMS\_OUTPUT.PUT\_LINE('Quantity of items: ' || quantity);  DBMS\_OUTPUT.PUT\_LINE('Calculated shipping cost: $' || shipping\_cost);    END; |

Assignment 3-7 Using scalar variables for data retrieval

The Brewbean's application contains a screen that displays order summary information, including IDBASKET, SUBTOTAL, SHIPPING, TAX, and TOTAL columns from the BB\_BASKET table. Create a PL/SQL block that will use scalar variables to retrieve this data and then display it on the screen. A host variable should provide the IDBASKET value. Test the block using IDBASKET 12.

|  |
| --- |
| DECLARE  -- Host variable for the basket ID (can be modified for testing)  basket\_id NUMBER := 12; -- Change this to test with other basket IDs    -- Scalar variables to store retrieved values  subtotal NUMBER;  shipping NUMBER;  tax NUMBER;  total NUMBER;  BEGIN  -- Retrieve the order summary data using the provided basket\_id  SELECT SUBTOTAL, SHIPPING, TAX, TOTAL  INTO subtotal, shipping, tax, total  FROM BB\_BASKET  WHERE IDBASKET = basket\_id;    -- Output the order summary information  DBMS\_OUTPUT.PUT\_LINE('Order Summary for Basket ID: ' || basket\_id);  DBMS\_OUTPUT.PUT\_LINE('-----------------------------------');  DBMS\_OUTPUT.PUT\_LINE('Subtotal: $' || TO\_CHAR(subtotal, '999.99'));  DBMS\_OUTPUT.PUT\_LINE('Shipping: $' || TO\_CHAR(shipping, '999.99'));  DBMS\_OUTPUT.PUT\_LINE('Tax: $' || TO\_CHAR(tax, '999.99'));  DBMS\_OUTPUT.PUT\_LINE('Total: $' || TO\_CHAR(total, '999.99'));    END; |

Assignment 3-8 Using a record variable for data retrieval

The Brewbean's application contains a screen that displays order summary information, including IDBASKET, SUBTOTAL, SHIPPING, TAX, and TOTAL columns from the BB\_BASKET table. Create a PL/SQL block that will use a record variable to retrieve this data and display it on the screen. A host variable should provide the IDBASKET value. Test the block using IDBASKET 12.

|  |
| --- |
| DECLARE  -- Host variable for the basket ID  basket\_id NUMBER := 12; -- Change this for testing with different basket IDs  -- Record type to store the basket data (IDBASKET, SUBTOTAL, SHIPPING, TAX, TOTAL)  TYPE basket\_record IS RECORD (  idbasket NUMBER,  subtotal NUMBER,  shipping NUMBER,  tax NUMBER,  total NUMBER  );  -- Record variable to hold the data for the basket  basket\_info basket\_record;  BEGIN  -- Retrieve the order summary data using the provided basket\_id  SELECT IDBASKET, SUBTOTAL, SHIPPING, TAX, TOTAL  INTO basket\_info.idbasket, basket\_info.subtotal, basket\_info.shipping, basket\_info.tax, basket\_info.total  FROM BB\_BASKET  WHERE IDBASKET = basket\_id;  -- Output the order summary information  DBMS\_OUTPUT.PUT\_LINE('Order Summary for Basket ID: ' || basket\_info.idbasket);  DBMS\_OUTPUT.PUT\_LINE('-----------------------------------');  DBMS\_OUTPUT.PUT\_LINE('Subtotal: $' || TO\_CHAR(basket\_info.subtotal, '999.99'));  DBMS\_OUTPUT.PUT\_LINE('Shipping: $' || TO\_CHAR(basket\_info.shipping, '999.99'));  DBMS\_OUTPUT.PUT\_LINE('Tax: $' || TO\_CHAR(basket\_info.tax, '999.99'));  DBMS\_OUTPUT.PUT\_LINE('Total: $' || TO\_CHAR(basket\_info.total, '999.99'));  END; |