week 9 pp answers

From now on, before you start working through each chapter's examples, you need to run a script to rebuild the database. This step ensures that the database is in the correct state to achieve the coding results shown in the chapter and includes the new objects required to for chapter examples. Use the script file provided above to rebuild the database.

Complete Hands-On Assignments 3.1-3.5 and 3-8 (Part I) for Chapter 3 given in the textbook. Use the script files provided above when appropriate

3.1 Querying Data in a Block

- A Brewbean's application page is being developed for employees to enter a basket number and view shipping information for the order including:
    - date
    - shipper
    - shipping number
- An IDSTAGE of 5 means the order has shipped
- There are steps we have to follow here. Code is below

```
DECLARE
    lv_ship_date bb_basketstatus.dtstage%TYPE;
    lv_shipper_txt bb_basketstatus.shipper%TYPE;
    lv_ship_num bb_basketstatus.shippingnum%TYPE;
    lv_back_num bb_basketstatus.idbasket%TYPE := 7;
begin
    SELECT dtstage, shipper, shippingnum
        INTO lv_ship_date, lv_shipper_txt, lv_ship_num
        FROM bb_basketstatus
        WHERE idbasket = lv_back_num
            AND idstage = 5;
    dbms_output.put_line( 'Date Shipped: ' || lv_ship_date );
    dbms_output.put_line( 'Shipper: ' || lv_shipper_txt );
    dbms_output.put_line( 'Shipping #: ' || lv_ship_num );
end;
/
```

```
SQL> DECLARE
    lv_ship_date bb_basketstatus.dtstage%TYPE;
    lv_shipper_txt bb_basketstatus.shipper%TYPE;
    lv_ship_num bb_basketstatus.shippingnum%TYPE;
    lv_back_num bb_basketstatus.idbasket%TYPE := 3;
begin
    SELECT dtstage, shipper, shippingnum
        INTO lv_ship_date, lv_shipper_txt, lv_ship_num
        FROM bb_basketstatus
        WHERE idbasket = lv_back_num
            AND idstage = 5;
    dbms_output.put_line( 'Date Shipped: ' || lv_ship_date );
    dbms_output.put_line( 'Shipper: ' || lv_shipper_txt );
    dbms_output.put_line( 'Shipping #: ' || lv_ship_num );
end;
/  2    3    4    5    6    7    8    9   10   11   12   13   14
Date Shipped: 25-JAN-12
Shipper: UPS
Shipping #: ZW845584GD89H569

PL/SQL procedure successfully completed.

SQL> michael lundquist 00737340
```

```
SQL> SQL> DECLARE
    lv_ship_date bb_basketstatus.dtstage%TYPE;
    lv_shipper_txt bb_basketstatus.shipper%TYPE;
    lv_ship_num bb_basketstatus.shippingnum%TYPE;
    lv_back_num bb_basketstatus.idbasket%TYPE := 7;
begin
    SELECT dtstage, shipper, shippingnum
        INTO lv_ship_date, lv_shipper_txt, lv_ship_num
        FROM bb_basketstatus
        WHERE idbasket = lv_back_num
            AND idstage = 5;
    dbms_output.put_line( 'Date Shipped: ' || lv_ship_date );
    dbms_output.put_line( 'Shipper: ' || lv_shipper_txt );
    dbms_output.put_line( 'Shipping #: ' || lv_ship_num );
end;
/  2    3    4    5    6    7    8    9   10   11   12   13   1
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 7


SQL> michael lundquist 00737340
```

3.2

A Brewbean's application page is being developed for employees to enter a basket number and view shipping information for the order. The page needs to display all column values from the BB_BASKETSTATUS table. An INSTAGE value of 5 in the BB_BASKETSTATUS table indicates that the order has been shipped. They tell you to run the following code

```
DECLARE
    rec_ship bb_basketstatus%ROWTYPE;
```

```
    lv_back_num bb_basketstatus.idbasket%TYPE := 3;
begin
  select *
    into rec_ship
    FROM bb_basketstatus
        WHERE idbasket = lv_back_num
            AND idstage = 5;

    dbms_output.put_line( 'Date Shipped: ' || rec_ship.dtstage );
    dbms_output.put_line( 'Shipper: ' || rec_ship.shipper );
    dbms_output.put_line( 'Shipping #: ' || rec_ship.shippingnum );
    dbms_output.put_line( 'Notes: ' || rec_ship.notes );

end;
/
```

```
SQL> DECLARE
    rec_ship bb_basketstatus%ROWTYPE;
    lv_back_num bb_basketstatus.idbasket%TYPE := 3;
begin
  select *
    into rec_ship
    FROM bb_basketstatus
        WHERE idbasket = lv_back_num
            AND idstage = 5;

    dbms_output.put_line( 'Date Shipped: ' || rec_ship.dtstage );
    dbms_output.put_line( 'Shipper: ' || rec_ship.shipper );
    dbms_output.put_line( 'Shipping #: ' || rec_ship.shippingnum );
    dbms_output.put_line( 'Notes: ' || rec_ship.notes );

end;
/  2    3    4    5    6    7    8    9   10   11   12   13   14   15
Date Shipped: 25-JAN-12
Shipper: UPS
Shipping #: ZW845584GD89H569
Notes: Customer called to confirm shipment

PL/SQL procedure successfully completed.

SQL> michael lundquist 00737340
```

3.3 Processing Database Data with IF Statements

The Brewbean's application needs a block to determine whether a customer is HIGH, MID or LOW based on his or her total purchases. The block needs to select the total amount of orders for a specified customer, determine the rating, and then display the results onscreen. The code rates the customer HIGH if total purchases are greater than $200, MID if greater than $100, and LOW if lower than $100. Use an initialized variable to provide the shopper ID.

```
DECLARE
 lv_total_num NUMBER(6,2);
 lv_rating_txt VARCHAR2(4) := 'LOW';
 lv_shop_num bb_basket.idshopper%TYPE := 22;
BEGIN
 SELECT SUM(total), idshopper
```

```
 into lv_total_num, lv_shop_num
  FROM bb_basket
  WHERE idShopper = 22
    AND orderplaced = 1
  GROUP BY idshopper;

  IF lv_total_num > 200
        THEN lv_rating_txt := 'HIGH';
    elsif lv_total_num > 100
        THEN lv_rating_txt := 'MID';
  END IF;
   DBMS_OUTPUT.PUT_LINE('Shopper '||lv_shop_num||' is rated '||lv_rating_txt);
END;
/
```

```
SQL> DECLARE
 lv_total_num NUMBER(6,2);
 lv_rating_txt VARCHAR2(4) := 'LOW';
 lv_shop_num bb_basket.idshopper%TYPE := 22;
BEGIN
 SELECT SUM(total), idshopper
 into lv_total_num, lv_shop_num
  FROM bb_basket
  WHERE idShopper = 22
    AND orderplaced = 1
  GROUP BY idshopper;

  IF lv_total_num > 200
        THEN lv_rating_txt := 'HIGH';
    elsif lv_total_num > 100
        THEN lv_rating_txt := 'MID';
  END IF;
   DBMS_OUTPUT.PUT_LINE('Shopper '||lv_shop_num||' is r
END;
/  2    3    4    5    6    7    8    9   10   11   12
  19   20
Shopper 22 is rated HIGH

PL/SQL procedure successfully completed.

SQL> michael lundquist
```

3.4

same as 3.3, but use a searched case statement.

```
DECLARE
 lv_total_num NUMBER(6,2);
 lv_rating_txt VARCHAR2(4) := 'LOW';
 lv_shop_num bb_basket.idshopper%TYPE := 22;
BEGIN
 SELECT SUM(total), idshopper
 into lv_total_num, lv_shop_num
  FROM bb_basket
```

```
    WHERE idShopper = 22
      AND orderplaced = 1
    GROUP BY idshopper;

    CASE
      WHEN lv_total_num > 200
          THEN lv_rating_txt := 'HIGH';
      WHEN lv_total_num > 100
          THEN lv_rating_txt := 'MID';
    END CASE;
     DBMS_OUTPUT.PUT_LINE('Shopper '||lv_shop_num||' is rated '||lv_rating_txt);
  END;
  /
```

```
SQL> DECLARE
 lv_total_num NUMBER(6,2);
 lv_rating_txt VARCHAR2(4) := 'LOW';
 lv_shop_num bb_basket.idshopper%TYPE := 22;
BEGIN
 SELECT SUM(total), idshopper
 into lv_total_num, lv_shop_num
  FROM bb_basket
  WHERE idShopper = 22
    AND orderplaced = 1
  GROUP BY idshopper;

  CASE
    WHEN lv_total_num > 200
        THEN lv_rating_txt := 'HIGH';
    WHEN lv_total_num > 100
        THEN lv_rating_txt := 'MID';
  END CASE;
   DBMS_OUTPUT.PUT_LINE('Shopper '||lv_shop_num||' is rated '||lv_rating_txt);
END;
/  2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17
  19   20   21
Shopper 22 is rated HIGH

PL/SQL procedure successfully completed.

SQL> MICHAEL LUNDQUSIT 00737340
```

3.5

Brewbean's wants to include a feature in its application that calculates the total amount (quantity) of a specified item that can be purchased with a given amount of money. Create a block with a WHILE loop to increment the item's cost 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 a product of your choice. Use initialized variables to provide the total spending amount and product ID.

```
DECLARE
 lv_balance_num NUMBER(6,2) := 100;

 lv_qty_txt NUMBER(4) := 0;
 lv_prodid_num bb_product.idproduct%TYPE := 4;
 lv_price_num bb_product.price%TYPE := 4;

BEGIN
 SELECT price
```

```
 into lv_price_num
  FROM bb_product
  WHERE idproduct = lv_prodid_num;

  WHILE lv_balance_num >= lv_price_num LOOP
   lv_balance_num := lv_balance_num - lv_price_num;
   lv_qty_txt := lv_qty_txt + 1;
  end loop;

  DBMS_OUTPUT.PUT_LINE('you can buy ' || lv_qty_txt || ' items at ' ||
lv_price_num || ' each.');
END;
/
```

```
SQL> DECLARE
 lv_balance_num NUMBER(6,2) := 100;

 lv_qty_txt NUMBER(4) := 0;
 lv_prodid_num bb_product.idproduct%TYPE := 4;
 lv_price_num bb_product.price%TYPE := 4;

BEGIN
 SELECT price
 into lv_price_num
  FROM bb_product
  WHERE idproduct = lv_prodid_num;

  WHILE lv_balance_num >= lv_price_num LOOP
   lv_balance_num := lv_balance_num - lv_price_num;
   lv_qty_txt := lv_qty_txt + 1;
  end loop;

  DBMS_OUTPUT.PUT_LINE('you can buy ' || lv_qty_txt || ' items at ' || lv_price_num |
| ' each.');
END;
/  2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18
  19   20   21
you can buy 3 items at 28.5 each.

PL/SQL procedure successfully completed.

SQL> michael lundquist 00737340
```

3.8

The Brewbean's application contains a page displaying order summary info, including IDBASKET, SUBTOTAL, SHIPPING, TAX, and TOTAL columns from the BB_BASKET table. Create a PL/SQL block with a record variable to retrieve this data and display it onscreen. An initialized variable should provide the IDBASKET value. Test the block usign the basket ID 12.*/

```
DECLARE
 lv_basket_var bb_basket%ROWTYPE;
BEGIN
 SELECT *
 into lv_basket_var
  FROM bb_basket
  WHERE idbasket = 12;

   DBMS_OUTPUT.PUT_LINE('Basket ID 12 has: ');
   DBMS_OUTPUT.PUT_LINE('subtotal: ' || lv_basket_var.subtotal);
   DBMS_OUTPUT.PUT_LINE('shipping: ' || lv_basket_var.shipping);
```

```
    DBMS_OUTPUT.PUT_LINE('tax: ' || lv_basket_var.tax);
    DBMS_OUTPUT.PUT_LINE('total: ' || lv_basket_var.total);
 END;
 /
```

```
SQL> DECLARE
 lv_basket_var bb_basket%ROWTYPE;
BEGIN
 SELECT *
 into lv_basket_var
  FROM bb_basket
  WHERE idbasket = 12;

   DBMS_OUTPUT.PUT_LINE('Basket ID 12 has: ');
   DBMS_OUTPUT.PUT_LINE('subtotal: ' || lv_basket_var.subtotal);
   DBMS_OUTPUT.PUT_LINE('shipping: ' || lv_basket_var.shipping);
   DBMS_OUTPUT.PUT_LINE('tax: ' || lv_basket_var.tax);
   DBMS_OUTPUT.PUT_LINE('total: ' || lv_basket_var.total);
END;
/  2    3    4    5    6    7    8    9   10   11   12   13   14   1
Basket ID 12 has:
subtotal: 72.4
shipping: 8
tax: 3.26
total: 83.66

PL/SQL procedure successfully completed.

SQL> michael lundquist 00737340
```