**BB BREWERYAPPLICATION**

**DATABASE BACKEND**

**SQL CODES**

|  |
| --- |
| **GROUP 10** |
|  |
|  |
|  |

Task1 : Creating a Procedure and GUI screen for Product Description change ability.

CREATE OR REPLACE PROCEDURE prod\_desc\_change(p\_id IN BB\_PRODUCT.IDPRODUCT%type,p\_new\_desc IN BB\_PRODUCT.DESCRIPTION%type )

IS

BEGIN

UPDATE bb\_product

SET description = p\_new\_desc

WHERE idproduct = p\_id;

Commit;

END prod\_desc\_change;

--testing—

EXECUTE(1,’CapressoBar Model #388’);

select \* from BB\_PRODUCT;

Graphical user interface, text, application, email

Description automatically generated

Task 2: Create(Enter) new product by using a Procedure with IN Parameters

CREATE OR REPLACE PROCEDURE PROD\_ADD\_SP (p\_name IN BB\_PRODUCT.PRODUCTNAME%type,

p\_desc IN BB\_PRODUCT.DESCRIPTION%type, p\_img IN BB\_PRODUCT.PRODUCTIMAGE%type, p\_price IN BB\_PRODUCT.PRICE%type, p\_active IN BB\_PRODUCT.ACTIVE%type )

IS

BEGIN

INSERT INTO BB\_PRODUCT(IDPRODUCT, PRODUCTNAME, DESCRIPTION, PRODUCTIMAGE, PRICE, ACTIVE)

VALUES (BB\_PRODID\_SEQ.NEXTVAL, p\_name, p\_desc, p\_img, p\_price, p\_active );

COMMIT;

END PROD\_ADD\_SP;

TESTING—

EXECUTE PROD\_ADD\_SP('Roasted Blend', 'Well-balanced mix of roasted beans, a medium body', 'roasted.jpg',9.50,1)

SELECT \* FROM BB\_PRODUCT;

Graphical user interface, text, application, email

Description automatically generated

Task 3: Calculating the Tax on an Order

CREATE OR REPLACE PROCEDURE TAX\_COST\_SP(p\_state IN BB\_TAX.STATE%type, p\_subtotal IN BB\_BASKET.SUBTOTAL%type, p\_tax\_amt OUT number )

IS

CURSOR cur\_bb\_tax\_cursor IS

SELECT state

FROM BB\_TAX

where state = p\_state;

BEGIN

FOR tax\_rec IN cur\_bb\_tax\_cursor

LOOP

IF (tax\_rec.state != p\_state)

THEN p\_tax\_amt := p\_subtotal;

ELSIF tax\_rec.state = 'VA'

THEN p\_tax\_amt := 0.045 \* p\_subtotal;

ELSIF tax\_rec.state = 'NC'

THEN p\_tax\_amt :=0.03 \* p\_subtotal;

ELSIF tax\_rec.state = 'SC'

THEN p\_tax\_amt :=0.06 \* p\_subtotal;

END IF;

END LOOP;

COMMIT;

END TAX\_COST\_SP;

--Testing—

DECLARE

v\_totalamt DECIMAL(4,2);

BEGIN

TAX\_COST\_SP('VA',100, v\_totalamt);

DBMS\_OUTPUT.PUT\_LINE('The tax amount is $' ||to\_char(v\_totalamt,'0.00'));

END;

Graphical user interface, application

Description automatically generated

Task 4: Updating Order Status

CREATE OR REPLACE PROCEDURE STATUS\_SHIP\_SP(p\_basketid IN bb\_basketstatus.idbasket%type,p\_date IN bb\_basketstatus.dtstage%type,p\_shipper IN bb\_basketstatus.shipper%type,p\_shipnum IN bb\_basketstatus.shippingnum%type)

IS

BEGIN

INSERT INTO bb\_basketstatus (idstatus, idbasket, idstage, dtstage,

shipper, shippingnum)

VALUES (bb\_status\_seq.NEXTVAL, p\_basketid, 3, p\_date, p\_shipper,

p\_shipnum);

COMMIT;

END;

--Testing—

Execute status\_ship\_sp(3,'20-FEB-12','UPS','ZW2384YXK4957');

Graphical user interface, text, application, email

Description automatically generated

Task 5: Adding Items to a Basket

CREATE OR REPLACE PROCEDURE BASKET\_ADD\_SP(

p\_productid IN bb\_basketitem.idproduct%type,

p\_price IN bb\_basketitem.price%type,

p\_quantity IN bb\_basketitem.quantity%type,

p\_basketid IN bb\_basketitem.idbasket%type,

p\_sizecode IN bb\_basketitem.option1%type,

p\_formcode IN bb\_basketitem.option2%type)

IS

BEGIN

INSERT INTO BB\_BASKETITEM (IDBASKETITEM, IDPRODUCT, PRICE, QUANTITY,

IDBASKET, OPTION1, OPTION2)

VALUES (BB\_IDBASKETITEM\_SEQ.NEXTVAL, p\_productid, p\_price, p\_quantity, p\_basketid, p\_sizecode, p\_formcode);

COMMIT;

END BASKET\_ADD\_SP;

--TESTING--

Execute BASKET\_ADD\_SP(8,10.80,1,14,2,4);

Graphical user interface, text, application

Description automatically generated

Task 6: Identifying Sale Products

CREATE OR REPLACE FUNCTION CK\_SALE\_SF(p\_productid bb\_product.idproduct%type, p\_date bb\_product.saleend%type)

RETURN VARCHAR2 IS

v\_sales VARCHAR2(20);

v\_salestart bb\_product.salestart%type;

v\_saleend bb\_product.saleend%type;

BEGIN

SELECT salestart, saleend

INTO v\_salestart, v\_saleend

FROM bb\_product

WHERE idproduct = p\_productid;

IF (to\_char(p\_date) >= to\_char(v\_salestart) and to\_char(p\_date) <= to\_char(v\_saleend))

THEN v\_sales:= 'ON SALE';

ELSE

v\_sales:= 'GREAT DEAL';

END IF;

RETURN v\_sales;

COMMIT;

END CK\_SALE\_SF ;

--Testing --

SET SERVEROUTPUT ON

DECLARE

sales varchar2(20);

BEGIN

sales := CK\_SALE\_SF(6,'12-Jun-06');

DBMS\_OUTPUT.PUT\_LINE(sales);

END;

Table

Description automatically generated with low confidence

Report 1: Report to show whether all items in her/his basket are in Stock or not? Using an  
Explicit Cursor

Report 2: Calculating a Shopper’s Total Spending