CIS 229

Professor: Bob Desilets

Date 10/17/2018

Cedric L Mulumba

**Hands-On Assignments Part I**

|  |
| --- |
| **Assignment 4-1: Using an Explicit Cursor** |
| DECLARE  CURSOR cur\_basket IS  SELECT bi.idBasket, bi.quantity, p.stock  FROM bb\_basketitem bi INNER JOIN bb\_product p  USING (idProduct)  WHERE bi.idBasket = 6;  TYPE type\_basket IS RECORD (  basket bb\_basketitem.idBasket%TYPE,  qty bb\_basketitem.quantity%TYPE,  stock bb\_product.stock%TYPE);  rec\_basket type\_basket;  lv\_flag\_txt CHAR(1) := 'Y';  BEGIN  OPEN cur\_basket;  LOOP  FETCH cur\_basket INTO rec\_basket;  EXIT WHEN cur\_basket%NOTFOUND;  IF rec\_basket.stock < rec\_basket.qty THEN lv\_flag\_txt := 'N'; END IF;  END LOOP;  CLOSE cur\_basket;  IF lv\_flag\_txt = 'Y' THEN DBMS\_OUTPUT.PUT\_LINE('All items in stock!'); END IF;  IF lv\_flag\_txt = 'N' THEN DBMS\_OUTPUT.PUT\_LINE('All items NOT in stock!'); END IF;  END; |
|  |
| **Assignment 4-2: Using a CURSOR FOR Loop** |
| DECLARE  CURSOR cur\_shopper IS  SELECT a.idShopper, a.promo, b.total  FROM bb\_shopper a, (SELECT b.idShopper, SUM(bi.quantity\*bi.price) total  FROM bb\_basketitem bi, bb\_basket b  WHERE bi.idBasket = b.idBasket  GROUP BY idShopper) b  WHERE a.idShopper = b.idShopper  FOR UPDATE OF a.idShopper NOWAIT;  lv\_promo\_txt CHAR(1);  BEGIN  FOR rec\_shopper IN cur\_shopper LOOP  lv\_promo\_txt := 'X';  IF rec\_shopper.total > 100 THEN  lv\_promo\_txt := 'A';  END IF;  IF rec\_shopper.total BETWEEN 50 AND 99 THEN  lv\_promo\_txt := 'B';  END IF;  IF lv\_promo\_txt <> 'X' THEN  UPDATE bb\_shopper  SET promo = lv\_promo\_txt  WHERE CURRENT OF cur\_shopper;  END IF;  END LOOP;  COMMIT;  END; |
|  |
| **Assignment 4-3: Using Implicit Cursors** |
| UPDATE bb\_shopper  SET promo = NULL;  UPDATE bb\_shopper  SET promo = 'B'  WHERE idShopper IN (21,23,25);  UPDATE bb\_shopper  SET promo = 'A'  WHERE idShopper = 22;  COMMIT;  BEGIN  UPDATE bb\_shopper  SET promo = NULL  WHERE promo IS NOT NULL;  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE(SQL%ROWCOUNT||' rows updated');  ELSE  DBMS\_OUTPUT.PUT\_LINE('No rows changed');  END IF;  END; |
|  |
| **Assignment 4-4: Using Exception Handling** |
| DECLARE  lv\_tax\_num NUMBER(2,2);  BEGIN  CASE 'NJ'  WHEN 'VA' THEN lv\_tax\_num := .04;  WHEN 'NC' THEN lv\_tax\_num := .02;  WHEN 'NY' THEN lv\_tax\_num := .06;  END CASE;  DBMS\_OUTPUT.PUT\_LINE('tax rate = '||lv\_tax\_num);  END; |
|  |
| DECLARE  lv\_tax\_num NUMBER(2,2);  BEGIN  CASE 'NJ'  WHEN 'VA' THEN lv\_tax\_num := .04;  WHEN 'NC' THEN lv\_tax\_num := .02;  WHEN 'NY' THEN lv\_tax\_num := .06;  END CASE;  DBMS\_OUTPUT.PUT\_LINE('tax rate = '||lv\_tax\_num);  EXCEPTION  WHEN CASE\_NOT\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('No tax');  END; |
|  |
| **Assignment 4-5: Handling Predefined Exceptions** |
| DECLARE  rec\_shopper bb\_shopper%ROWTYPE;  lv\_shopper\_num NUMBER (3) := 99;  BEGIN  SELECT \*  INTO rec\_shopper  FROM bb\_shopper  WHERE idShopper = lv\_shopper\_num;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('Invalid shopper ID');  END; |
|  |
| **Assignment 4-6: Handling Exceptions with Undefined Errors** |
| DECLARE  ex\_Ck\_Qty EXCEPTION;  PRAGMA EXCEPTION\_INIT(ex\_Ck\_Qty, -02290);  lv\_quatity bb\_basketitem.quantity%TYPE;  BEGIN  INSERT INTO bb\_basketitem  VALUES (88,8,10.8,21,16,2,3);  IF lv\_quatity < 20 THEN  RAISE ex\_Ck\_Qty;  END IF;  EXCEPTION  WHEN ex\_Ck\_Qty THEN  DBMS\_OUTPUT.PUT\_LINE('Check Quantity');  END; |
|  |
| **Assignment 4-7: Handling Exceptions with User-Defined Errors** |
| DECLARE  ex\_basket\_update EXCEPTION;  lv\_old\_num NUMBER(10,2) := 30;  lv\_new\_num NUMBER(10,2) := 4;  BEGIN  UPDATE bb\_basketitem  SET idBasket = lv\_new\_num  WHERE idBasket = lv\_old\_num;  IF SQL%NOTFOUND THEN  RAISE ex\_basket\_update;  END IF;  EXCEPTION  WHEN ex\_basket\_update THEN  DBMS\_OUTPUT.PUT\_LINE('Invalid original basket ID');  END; |
|  |
| **Assignment 4-8: Processing and Updating a Group of Rows** |
|  |
|  |