Sheikh Nooruddin

1507062

Mail Order Database System

Project Overview :

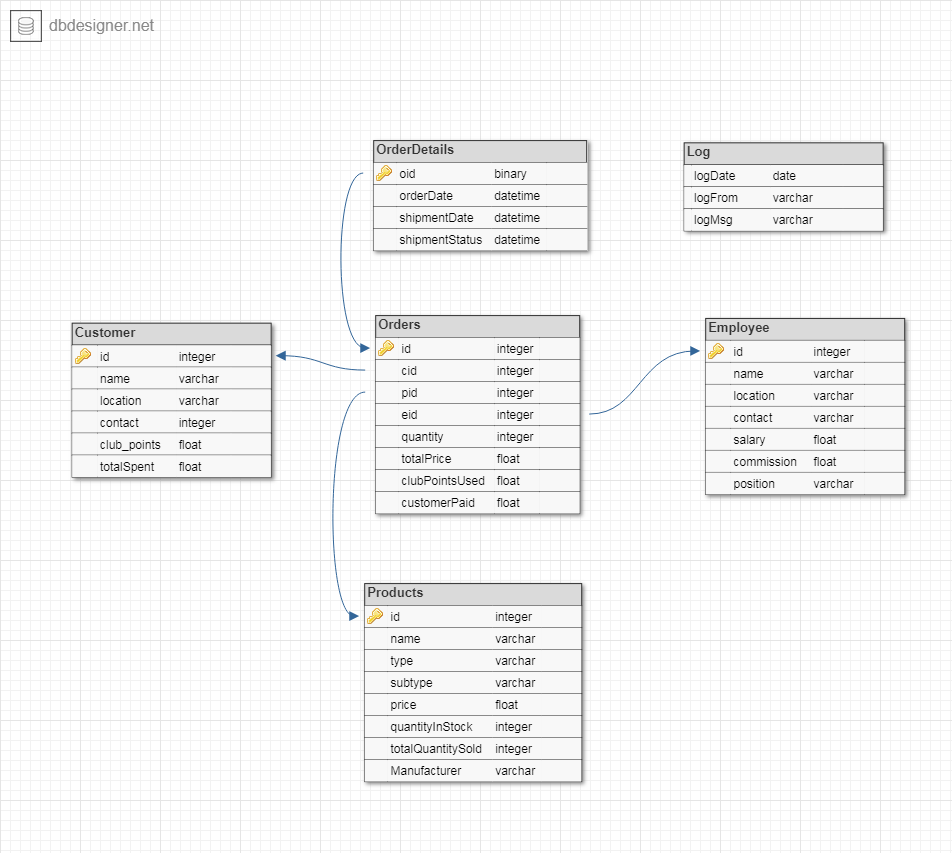
This Database system is for a online shop who has various products in its warehouse. This online shop has multiple customer care operators who permit the orders placed by registered customers and ship them to the customer addresses.

Database Overview :

The database will contain 6 tables. The tables are:

* Customers : contains id, name, location, contact, club points and total spent. Id is the primary key. Contact must be unique. Club points and total spent will be default to 0.
* Employees : contains id, name, location, contact, position, salary and commission. Id is the primary key. Contact must be unique. Salary and commission will be default to 0.
* Products : contains id, name, manufacturer, type, subtype, price, quantity in stock and total quantity sold so far(from the beginning of company). Id is the primary key. Price and quantityInStock should be greater than 0.
* Orders : contains id, customer id, employee id, product id, quantity, total price, clubPointUsed and customerPaid. Id is the primary key. Quantity and total price should be greater than 0.
* OrderDetails : contains order id, order date, possible shipping date(automatically three days from order date), shipment status. Order id is both the primary key and foreign key. Order id references from id of Orders table.
* Log : contains logDate, logFrom(whether logging from trigger, procedure, function etc) and logMsg. This table is completely disassociated from other tables as this table is created for debugging and logging purposes.

Schema Diagram :



Functionality and Design :

1. Admin can set discount on all or specific type of product.  
   **Implementation type : Procedure**

*Procedure Name : setDiscountOnAllProducts*

*Parameters : Amnt [IN] NUMBER*

*Description : This procedure updates the price of all the products to their discounted price.The percetage of discount is passed through the 'Amnt' parameter*

CREATE OR REPLACE PROCEDURE SETDISCOUNTONALLPRODUCTS(AMNT IN NUMBER)

AS

CURSOR PROD\_INFO

IS

SELECT

ID

, PRICE

FROM PRODUCTS;

PRODID PRODUCTS.ID%TYPE;

CRNTPRICE PRODUCTS.PRICE%TYPE;

UPDPRICE PRODUCTS.PRICE%TYPE;

F ***UTL\_FILE.FILE\_TYPE***;

BEGIN

OPEN PROD\_INFO;

F := ***UTL\_FILE.FOPEN***('DNAME','ActualPrice.csv','W');

IF ***UTL\_FILE.IS\_OPEN***(F) THEN

***DBMS\_OUTPUT.PUT\_LINE***('ActualPrice.csv OPENED FOR WRITING');

***UTL\_FILE.PUT***(F, 'id' || ',' || 'price' );

***UTL\_FILE.NEW\_LINE***(F);

LOOP

FETCH PROD\_INFO INTO PRODID, CRNTPRICE;

EXIT WHEN PROD\_INFO%NOTFOUND;

***UTL\_FILE.PUT***(F, PRODID || ',' || CRNTPRICE);

***UTL\_FILE.NEW\_LINE***(F);

UPDPRICE := (CRNTPRICE \* 11) / 100;

UPDATE PRODUCTS

SET PRICE = UPDPRICE

WHERE ID = PRODID;

END LOOP;

END IF;

COMMIT;

***UTL\_FILE.FCLOSE***(F);

CLOSE PROD\_INFO;

END;

SHOW ERRORS;

1. Admin can remove discount on all or specific type of product.  
   **Implementation type : Function**

*Function Name : resetDiscountsOnAllProducts*

*Return Type : NUMBER*

*Description : Resets discount on all the products and return their price to original value before discount. Return the number of products updated.*

CREATE OR REPLACE FUNCTION RESETDISCOUNTSONALLPRODUCTS

RETURN NUMBER

AS

F ***UTL\_FILE.FILE\_TYPE***;

LINE VARCHAR(10000);

PRODID PRODUCTS.ID%TYPE;

ACTUALPRICE PRODUCTS.PRICE%TYPE;

UPDATED NUMBER;

BEGIN

UPDATED := 0;

F := ***UTL\_FILE.FOPEN***('DNAME','ActualPrice.csv', 'R');

IF ***UTL\_FILE.IS\_OPEN***(F) THEN

***DBMS\_OUTPUT.PUT\_LINE***('ActualPrice.csv OPENED FOR READING');

***UTL\_FILE.GET\_LINE***(F, LINE, 10000);

LOOP

***UTL\_FILE.GET\_LINE***(F, LINE, 10000);

PRODID := REGEXP\_SUBSTR(LINE, '[^,]+', 1, 1);

ACTUALPRICE := REGEXP\_SUBSTR(LINE, '[^,]+', 1, 2);

UPDATE PRODUCTS

SET PRICE = ACTUALPRICE

WHERE ID = PRODID;

UPDATED := UPDATED + 1;

END LOOP;

END IF;

***UTL\_FILE.FCLOSE***(F);

COMMIT;

RETURN UPDATED;

END;

SHOW ERRORS;

1. Admin can calculate gross sale for a month.  
   **Implementation type : Function**.

*Function Name : grossSale*

*Return Type : NUMBER*

*Description : Return total sale of all products in record*

1. Admin can calculate total salary + commission to be paid to employees for a month.  
   **Implementation type : Function**

*Function Name : totalPaidToEmployees*

*Return Type : NUMBER*

*Description : Return total amount of commission and salary to be paid to employees*

1. Admin can pay the commission to all or specific employee any time.  
   **Implementation type : Procedure**

*Procedure Name : payAllCommission*

*Description : This procedure sets the commission of all the employees to 0*

1. Admin can get single customer or employee info.  
   **Implementation type : Procedure**

*Procedure Name : getSingleEmployeeInfo*

*Parameters : EMPID [IN] INTEGER*

*Description : Prints All Available Information of Employee given the employee id is passed in the parameter.*

1. Admin can get info of most sold products.  
   **Implementation type : Procedure**

*Procedure Name : getMostSoldProducts*

*Description : Return list of 10 most sold products in inventory*

1. Admin can get the info of top 10 most spending customers.  
   **Implementation type : Procedure**

*Procedure Name : mostPayingCustomers*

*Description : Prints all available information of the 10 most paying customers*

CREATE OR REPLACE PROCEDURE MOSTPAYINGCUSTOMERS

AS

CURSOR CUST\_INFO

IS

SELECT

NAME

, LOCATION

, CONTACT

, TOTALSPENT

FROM CUSTOMERS

ORDER BY TOTALSPENT;

CUSTID CUSTOMERS.ID%TYPE;

CUSTNAME CUSTOMERS.NAME%TYPE;

CUSTLOC CUSTOMERS.LOCATION%TYPE;

CUSTCONT CUSTOMERS.CONTACT%TYPE;

CUSTSPENT CUSTOMERS.CONTACT%TYPE;

COUNTER INTEGER;

BEGIN

OPEN CUST\_INFO;

FOR COUNTER IN 0..9

LOOP

FETCH CUST\_INFO INTO CUSTNAME, CUSTLOC, CUSTCONT, CUSTSPENT;

***DBMS\_OUTPUT.PUT\_LINE***('CUSTOMER NAME: ' || CUSTNAME || ' LOCATION: ' || CUSTLOC || ' CONTACT: ' || CUSTCONT || ' TOTAL SPENT: ' || CUSTSPENT );

END LOOP;

CLOSE CUST\_INFO;

END;

SHOW ERRORS;

1. Admin can get a list of products that are less than 10 available in stock.  
   **Implementation type : Function**

*Function Name : getUnderstockedProducts*

*Return Type : NUMBER*

*Description : Prints the information of products that are less than 10 available in stock. Return the number of such products.*

CREATE OR REPLACE FUNCTION GETUNDERSTOCKEDPRODUCTS

RETURN NUMBER

AS

CURSOR PROD\_INFO

IS

SELECT

ID

, NAME

, MANUFACTURER

, QUANTITYINSTOCK

, TOTALQUANTITYSOLD

FROM PRODUCTS

WHERE QUANTITYINSTOCK < 10;

PRODID PRODUCTS.ID%TYPE;

PRODNAME PRODUCTS.NAME%TYPE;

PRODMAN PRODUCTS.MANUFACTURER%TYPE;

PRODQUANT PRODUCTS.QUANTITYINSTOCK%TYPE;

PRODTOTSOLD PRODUCTS.TOTALQUANTITYSOLD%TYPE;

NUMOFPROD NUMBER(9,2) := 0;

BEGIN

OPEN PROD\_INFO;

LOOP

FETCH PROD\_INFO INTO PRODID, PRODNAME, PRODMAN, PRODQUANT, PRODTOTSOLD;

EXIT WHEN PROD\_INFO%NOTFOUND;

***DBMS\_OUTPUT.PUT\_LINE***('PRODUCT ID: ' || PRODID || 'NAME: ' || PRODNAME || 'MANUFACTURER' || PRODMAN || 'QUANTITY IN STOCK: ' || PRODQUANT || 'TOTAL SOLD: ' || PRODTOTSOLD );

NUMOFPROD := NUMOFPROD + 1;

END LOOP;

CLOSE PROD\_INFO;

RETURN NUMOFPROD;

END;

SHOW ERRORS;

1. Admin can award club points to customers.  
   **Implementation type : Procedure**

*Procedure Name : awardClubPointsToAllCustomers*

*Parameters : Award [IN] NUMBER*

*TotalAwardedTo [OUT] INTEGER*

*Description : Adds club points to all customers, club points are supplied through 'Award', The number of customers that are awarded the club points are stored in 'TotalAwardedTo'.*

1. Admin can update current shipment status of all the orders.  
   **Implementation type : Procedure**

*Procedure Name : updateShipmenStatus*

*Description : Updates shipment status of orders in the order details table.By default, the possible shipment date is set to three days from the order date and shipment status is set to 'processing'. This function will compare the shipment date with SYSDATE and if it has already been three days from order date, will update the shipment status to 'shipped'.*

CREATE OR REPLACE PROCEDURE UPDATESHIPMENTSTATUS

AS

CURSOR ORDERDET IS

SELECT ID

, SHIPMENTDATE

FROM ORDERDETAILS;

CURRDATE ORDERDETAILS.SHIPMENTDATE%TYPE;

OID ORDERDETAILS.ID%TYPE;

SHIPDATE ORDERDETAILS.SHIPMENTDATE%TYPE;

BEGIN

CURRDATE := TO\_DATE(TO\_CHAR(SYSDATE, 'dd/mm/yyyy'),'dd/mm/yyyy');

OPEN ORDERDET;

LOOP

FETCH ORDERDET INTO OID, SHIPDATE;

EXIT WHEN ORDERDET%NOTFOUND;

IF CURRDATE = SHIPDATE THEN

UPDATE ORDERDETAILS

SET SHIPMENTSTATUS = 'SHIPPED'

WHERE ID = OID;

END IF;

END LOOP;

COMMIT;

END;

SHOW ERRORS;

1. Admin can restock all or specific products.  
   **Implementation type : Procedure**

*Procedure Name : restockAllProducts*

*Parameters : Amnt [IN] INTEGER*

*Updtd [OUT] INTEGER*

*Description : Adds the amount supplied via 'Amnt' to QunatityInStock of all products. Stores the number of products updated via this procedure to 'Updtd'.*

1. All commission of an employee must be cleared before admin can delete an employee.   
   **Implementation type : Trigger**
2. Customer can't order products that are unavailable in stock.  
   **Implementation type : Trigger**

IF PRODQUANT = 0 OR :NEW.QUANTITY > PRODQUANT THEN

RAISE\_APPLICATION\_ERROR(-20001, 'QUANTITY NOT AVAILABLE');

END IF;

1. Customer can delete an order.   
   **Implementation type : Trigger**
2. Customer gets 5% of total order price(not including club point discount price) as club points.  
   **Implementation type : Trigger**

CURRCLUB := CURRCLUB + ACTUALPAYMENT \* .05;

UPDATE CUSTOMERS SET CLUBPOINTS = CURRCLUB WHERE ID = :NEW.CID;

1. Customer can use club points to buy products.   
   **Implementation type : Trigger**

SELECT CLUBPOINTS INTO CURRCLUB FROM CUSTOMERS WHERE ID = :NEW.CID;

IF :NEW.CLUBPOINTSUSED > CURRCLUB THEN

RAISE\_APPLICATION\_ERROR(-20003, 'NOT ENOUGH CLUB POINTS');

END IF;

1. Every operation (including procedures, function calls, triggers, updates, deletions) are tracked in a table with date.   
   **Implementation type : Functions, Procedures, Triggers**
2. Operators get 2% commission on the total price of the order.   
   **Implementation type : Trigger**

SELECT COMMISSION INTO CURRCOMM FROM EMPLOYEES WHERE ID = :NEW.EID;

CURRCOMM := CURRCOMM + TOTALORDERPRICE \* .02;

UPDATE EMPLOYEES SET COMMISSION = CURRCOMM WHERE ID = :NEW.EID;

1. Total price of a order is automatically calculated.  
   **Implementation type : Trigger**

SELECT price INTO pricePerPiece FROM products WHERE id = :NEW.PID;

totalOrderPrice := :NEW.quantity \* pricePerPiece;

:NEW.totalPrice := totalOrderPrice;

1. Only operators can permit a product.   
   **Implementation type : Trigger**

SELECT POSITION INTO EMPTYPE FROM EMPLOYEES WHERE ID = :NEW.EID

IF EMPTYPE != 'operator' THEN

RAISE\_APPLICATION\_ERROR(-20002, 'EMPLOYEE NOT AN OPERATOR');

END IF

1. Customers actual payment is automatically calculated.   
   **Implementation type : Trigger**

ACTUALPAYMENT := TOTALORDERPRICE - :NEW.CLUBPOINTSUSED;

:NEW.CUSTOMERPAID := ACTUALPAYMENT;

1. Quantity of products in stock after each order automatically decreases.  
   **Implementation type : Trigger**

PRODQUANT := PRODQUANT - :NEW.QUANTITY;

UPDATE PRODUCTS SET QUANTITYINSTOCK = PRODQUANT WHERE ID = :NEW.PID;

1. Total quantity sold of a product increases after every order.  
   **Implementation type : Trigger**

SELECT TOTALQUANTITYSOLD INTO TOTSOLD FROM PRODUCTS WHERE ID = :NEW.PID;

TOTSOLD := TOTSOLD + :NEW.QUANTITY;

UPDATE PRODUCTS SET TOTALQUANTITYSOLD = TOTSOLD WHERE ID = :NEW.PID;

1. Customer's total spent money increases after every order.  
   **Implementation type : Trigger**

SELECT TOTALSPENT INTO CURRSPENT FROM CUSTOMERS WHERE ID = :NEW.CID;

CURRSPENT := CURRSPENT + ACTUALPAYMENT;

UPDATE CUSTOMERS SET TOTALSPENT = CURRSPENT WHERE ID = :NEW.CID;

1. Order details will be automatically filled out.   
   **Implementation type : Trigger**

INSERT INTO ORDERDETAILS(ID, ORDERDATE, SHIPMENTDATE, SHIPMENTSTATUS)

VALUES(:NEW.ID, TO\_DATE(TO\_CHAR(SYSDATE, 'dd/mm/yyyy'),'dd/mm/yyyy'), TO\_DATE(TO\_CHAR(SYSDATE + 3, 'dd/mm/yyyy'),'dd/mm/yyyy'), 'Processing');

Summary :

This database system emulates a mail ordering company that has some products in their warehouse and serves those products to customers via mail system. The employees of the company receive the customer calls and ships the products. This database system development was a huge learning opportunity. By developing this project, I have learnt the necessary skills to develop database system for any real-life project. I have learnt to plan and design a database system from scratch and implement the necessary functionalities