RETAIL STORE MANAGEMENT SYSTEM

Project Report

Table of Contents

Conceptual Design of the Retail Store Database 2

Logical Database Design..... 6

ER Diagram..... 13

Retail Store Overview 14

Triggers-Validations,Business & Logic..... 15

Conceptual Design of the Retail Store database

In this schema the entities and its relationship are as follows:

EMPLOYEE

- ssn (ssn number Uniquely identifies the employee)
- fname (Identifies the First Name for employee)
- lname(Identifies the Last Name for employee)
- bdate (Determines the date of birth of the employee)
- sex(Identifies the sex for the employee)
- salary (Amount of salary given to the employee)
- deptnumber(Determines the department in which the employee is working)
- address1(Determines the address of the employee)
- address2(Determines the address of the employee)
- city(Determines the city of resident of the employee)
- state(Determines the state of resident of the employee)
- zip(Determines the zipcode of the employee)

DEPARTMENT

- deptnumber (Department number Identifies the Department)
- deptname (Descriptive name for the Department)
- mgrssn(Identifies the ssn number of the manager of the department)
- locationnumber(Identifies the location in which the Department is situated)

LOCATION

- locationnumber(Identifies the location in which the Department is situated)
- locationname (Location name for the Department)
- lmgrssn(Identifies the ssn number of the manager of the location in which department is situated)

WORKS

- workid(Id number Uniquely identifies the department the employee works in)
- essn (ssn number of the employee)
- deptnumber (Department number Identifies the Department)
- hours(determines the number of hours the employee works)

INVENTORY

- itemnumber (Identifies item in the Inventory)
- itemname (Item name for the item in the Inventory)
- suppliernumber(Supplier number Uniquely identifies the Supplier of the item)
- unitprice(Determines the item price)
- quantityinhand(Determines the quantity of item in the Inventory)
- deptnumber (Department number Identifies the Department)

SUPPLIER

- suppliernumber(Supplier number Uniquely identifies the Supplier of the item)
- suppliername (Name of the supplier)
- deptnumber (Department number Identifies the Department)
- address1(Determines the address of the Supplier)
- address2(Determines the address of the Supplier)
- city(Determines the city of resident of the Supplier)
- state(Determines the state of resident of the Supplier)
- zip(Determines the zipcode of the Supplier)
-

ORDERS

- ordernumber (Uniquely Identifies the order placed)
- suppliernumber(Supplier number Uniquely identifies the Supplier of the item)
- orderdate (Identifies the Order date)

SUPPLIERORDER

- supplierordernumber (Identifies supplier order number in the SupplierOrder)
- ordernumber (Identifies the order placed)
- itemnumber (Identifies item in the SupplierOrder)
- qty (Determines the quantity of item in the SupplierOrder)
- unitprice(Determines the item price)
- check(Checking if the price is greater than zero)

TRANSACTIONS

- transactionid (Uniquely Identifies transaction id in the Transactions)
- storecard (Identifies the card that is issued by the store)
- paymenttype (Identifies the type of payment made by the customer)
- checknumber (If the payment is made by check then it checks for the checknumber)
- creditcardtype(If the payment is made by credit card then it checks for the type of credit card used)

- creditexpdate(Checking for the expiry date of the credit card used)
- creditnumber(Identifies the credit card number used)
- dateoftransaction(Identifies the date of transaction made)
- amount(Identifies the amount to be paid)

SALES

- salesid (Uniquely Identifies id of the Sales)
- transactionid (Uniquely Identifies transaction id in the Transactions)
- itemnumber (Identifies item in the Sales)
- qty (Determines the quantity of item in the Sales)

ERRORLOG

- errorlogid(Uniquely Identifies the id in the error table)
- errordescription(Description for the type of error made)
- creationdate (Date when the error was created)

PRICE LOOK UP (PLU)

- pricelookupid(Uniquely identifies the PLU id for a particular item)
- itemnumber (Identifies item in the PLU)
- description (Description for the PLU Item)
- pricenumber (Defines the price of the PLU)
- activeorpassive (Identifies if the PLU is Active or passive)

TOTALSALE

- totalsaleid(Uniquely Identifies the id in the Total Sale table)
- totalsaleamount(Amount of the Total Sale made)
- creationdate (Date when the Total Sale was created)

RETURN

- returnid(Uniquely Identifies the return of the item in the Return Table)
- transactionid (Uniquely Identifies transaction id in the Transactions)
- itemnumber (Identifies item in the PLU)

TEMPMINING (Like a warehouse data. Not normalized. Useful for decision taking process)

- tempminingid (Uniquely Identifies tempmining id in the TempMining)
- transactionid (Identifies transaction id in the TempMining)
- storecard (Identifies the card that is issued by the store)
- paymenttype (Identifies the type of payment made by the customer)
- checknumber (If the payment is made by check then it checks for the checknumber)
- creditcardtype(If the payment is made by credit card then it checks for the type of credit card used)
- creditexpdate(Checking for the expiry date of the credit card used)
- creditnumber(Identifies the credit card number used)
- dateoftransaction(Identifies the date of transaction made)
- amount(Identifies the amount to be paid)
- salesid (Uniquely Identifies id of the TempMining)
- itemnumber (Identifies item in the TempMining)
- qty (Determines the quantity of item in the TempMining)
- itemname (Item name for the item in the TempMining)
- suppliernumber(Supplier number identifies the Supplier of the item in TempMining)
- unitprice(Determines the item price)
- quantityinhand(Determines the quantity of item in the TempMining)
- deptnumber (Department number Identifies the TempMining)

SHAREDDATA

- sharedataid(Uniquely Identifies the sharedata of the item in the ShareData Table)
- i (An integer defined)

SEQ_USER_ID

- SEQ_USER_ID(Uniquely Identifies the id of the item)
- MINVALUE 1(Assigns the minimum value to one)
- MAXVALUE 9999999999999999(Assigns the maximum value to declared one)
- START WITH 1(Start assigning values with 1)
- INCREMENT BY 1(Increment each time the value by 1)
- NOCACHE

Logical Database Design

The Logical model of POS database consists of the following tables.

TABLE NAME	DESCRIPTION	TABLE ATTRIBUTES
EMPLOYEE	Determines the details of the employee in the retail store	ssn (ssn number Uniquely identifies the employee) fname (Identifies the First Name for employee) lname(Identifies the Last Name for employee) bdate (Determines the date of birth of the employee) sex(Identifies the sex for the employee) salary (Amount of salary given to the employee) deptnumber(Determines the department in which the employee is working) address1(Determines the address of the employee) address2(Determines the address of the employee) city(Determines the city of resident of the employee) state(Determines the state

		of resident of the employee) zip(Determines the zipcode of the employee)
DEPARTMENT	Determines the details of the department in the retail store	deptnumber (Department number Identifies the Department) deptname (Descriptive name for the Department) mgrssn(Identifies the ssn number of the manager of the department) locationnumber(Identifies the location in which the Department is situated)
LOCATION	Determines the location of the department, its name and the ssn of the manager.	locationnumber(Identifies the location in which the Department is situated) locationname (Location name for the Department) lmgrssn(Identifies the ssn number of the manager of the location in which department is situated)
WORKS	Determines the employee working in a particular department.	workid(Id number Uniquely identifies the department the employee works in) essn (ssn number of the employee) deptnumber (Department number Identifies the Department) hours(determines the number of hours the employee works)
INVENTORY	Determines the inventory of the retail store which has the stock of the items in the inventory.	itemnumber (Identifies item in the Inventory) itemname (Item name for the item in the Inventory) suppliernumber(Supplier number Uniquely identifies the

		Supplier of the item) unitprice(Determines the item price) quantityinhand(Determines the quantity of item in the Inventory) deptnumber (Department number Identifies the Department)
SUPPLIER	Determines the details of the Supplier who supplies the items to the Retail store.	suppliernumber(Supplier number Uniquely identifies the Supplier of the item) suppliername (Name of the supplier) deptnumber (Department number Identifies the Department) address1(Determines the address of the Supplier) address2(Determines the address of the Supplier) city(Determines the city of resident of the Supplier) state(Determines the state of resident of the Supplier) zip(Determines the zipcode of the Supplier)
ORDERS	Determines the orders placed by the retail store	ordernumber (Uniquely Identifies the order placed) suppliernumber(Supplier number Uniquely identifies the Supplier of the item) orderdate (Identifies the Order date)
SUPPLIER ORDER	Determines the orders placed by the retail store to the supplier based on the quantity placed in the store.	supplierordernumber (Identifies supplier order number in the SupplierOrder) ordernumber (Identifies the order placed) itemnumber (Identifies item in the SupplierOrder) qty (Determines the quantity of

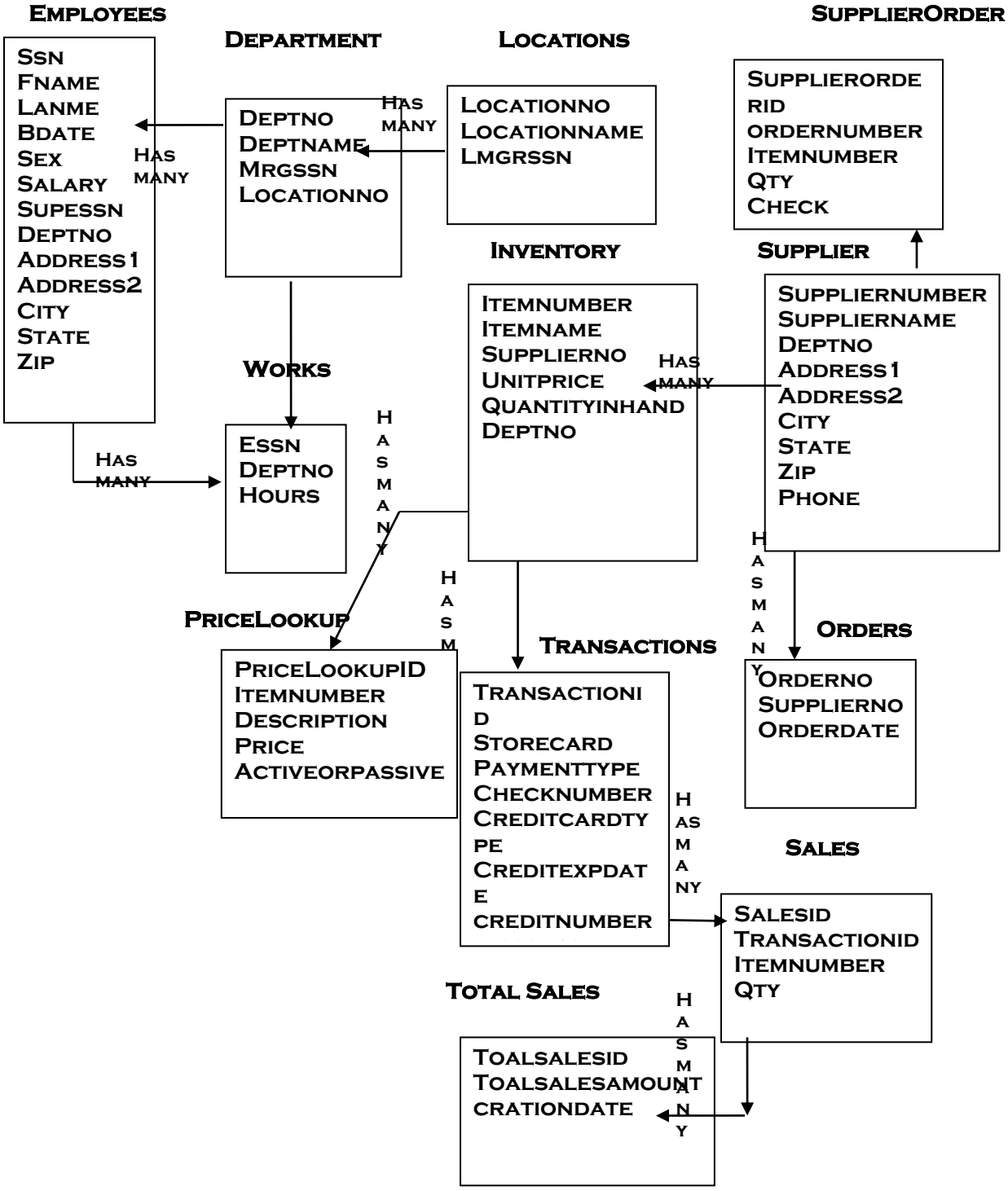
		item in the SupplierOrder) unitprice(Determines the item price) check(Checking if the price is greater than zero)
TRANSACTION	Determines the transaction the customer undergoes when he buys the items from the retail store.	transactionid (Uniquely Identifies transaction id in the Transactions) storecard (Identifies the card that is issued by the store) paymenttype (Identifies the type of payment made by the customer) checknumber (If the payment is made by check then it checks for the checknumber) creditcardtype(If the payment is made by credit card then it checks for the type of credit card used) creditexpdate(Checking for the expiry date of the credit card used) creditnumber(Identifies the credit card number used) dateoftransaction(Identifies the date of transaction made) amount(Identifies the amount to be paid)
SALES	Determines the sales made by the retail store of a particular item.	salesid (Uniquely Identifies id of the Sales) transactionid (Uniquely Identifies transaction id in the Transactions) itemnumber (Identifies item in the Sales) qty (Determines the quantity of item in the Sales)
ERROR LOG	Determines the error that occurs when the user of the system tries to put some item in the database	errorlogid(Uniquely Identifies the id in the error table) errordescription(Description for the type of error made)

		creationdate (Date when the error was created)
PRICE LOOK UP	Determines the price for every item that is bought by the customer.	pricelookupid(Uniquely identifies the PLU id for a particular item) itemnumber (Identifies item in the PLU) description (Description for the PLU Item) pricenumber (Defines the price of the PLU) activeorpassive (Identifies if the PLU is Active or passive)
TOTAL SALE ID	Determines the total sale made by the retail store at a particular location.	totalsaleid(Uniquely Identifies the id in the Total Sale table) totalsaleamount(Amount of the Total Sale made) creationdate (Date when the Total Sale was created)
RETURN	Determines the item that is returned by the customer.	returnid(Uniquely Identifies the return of the item in the Return Table) transactionid (Uniquely Identifies transaction id in the Transactions) itemnumber (Identifies item in the PLU)
TEMP MINING	This takes the records from the Transaction table , Sales table and the Inventory table and applies data mining techniques on the data starting from a specific date and gives the manager of the retail store some rules which the manager can apply in order to improve the sales of the store.	tempminingid (Uniquely Identifies id in the TempMining) transactionid (Identifies transaction id in the TempMining) storecard (Identifies the card that is issued by the store) paymenttype (Identifies the type of payment made by the customer) checknumber (If the payment is made by check then it checks for the checknumber)

		creditcardtype(If the payment is made by credit card then it checks for the type of credit card used) creditexpdate(Checking for the expiry date of the credit card used) creditnumber(Identifies the credit card number used) dateoftransaction(Identifies the date of transaction made) amount(Identifies the amount to be paid) salesid (Uniquely Identifies id of the TempMining) itemnumber (Identifies item in the TempMining) qty (Determines the quantity of item in the TempMining) itemname (Item name for the item in the TempMining) suppliernumber(Supplier number identifies the Supplier of the item in TempMining) unitprice(Determines the item price) quantityinhand(Determines the quantity of item in the TempMining) deptnumber (Department number Identifies the TempMining)
SHARE DATA	Determines the data that is shared among the tables	sharedataid(Uniquely Identifies the sharedata of the item in the ShareData Table) i (An integer defined)
SEQ_USER_ID	Gives a unique id for the records entered in the table.	SEQ_USER_ID(Uniquely Identifies the id of the item) MINVALUE 1(Assigns the minimum value to one) MAXVALUE 9999999999999999(Assigns the maximum value to declared

		one) START WITH 1(Start assigning values with 1) INCREMENT BY 1(Increment each time the value by 1) NOCACHE

ER DIAGRAM


EMPLOYEES
SSN
FNAME
LANME
BDATE
SEX
SALARY
SUPESSN
DEPTNO
ADDRESS1
ADDRESS2
CITY
STATE
ZIP
HAS MANY
DEPARTMENT
DEPTNO
DEPTNAME
MRGSSN
LOCATIONNO
HAS MANY
LOCATIONS
LOCATIONNO
LOCATIONNAME
LMGRSSN
SUPPLIERORDER
SUPPLIERORDERID
ORDERNUMBER
ITEMNUMBER
QTY
CHECK
INVENTORY
ITEMNUMBER
ITEMNAME
SUPPLIERNO
UNITPRICE
QUANTITYINHAND
DEPTNO
HAS MANY
SUPPLIER
SUPPLIERNUMBER
SUPPLIERNAME
DEPTNO
ADDRESS1
ADDRESS2
CITY
STATE
ZIP
PHONE
WORKS
HAS MANY
ESSN
DEPTNO
HOURS
HAS MANY
HAS MANY
PRICELOOKUP
PRICELOOKUPID
ITEMNUMBER
DESCRIPTION
PRICE
ACTIVEORPASSIVE
HAS MANY
TRANSACTIONS
TRANSACTIONID
STORECARD
PAYMENTTYPE
CHECKNUMBER
CREDITCARDTYPE
CREDITEXPDATE
CREDITNUMBER
HAS MANY
ORDERS
ORDERNO
SUPPLIERNO
ORDERDATE
SALES
SALESID
TRANSACTIONID
ITEMNUMBER
QTY
TOTAL SALES
TOALSALESID
TOALSALESAMOUNT
CRATIONDATE
HAS MANY

RETAIL STORE SYSTEM

Objective: The purpose of retail store system is to record the sale of items sold at the retail store. The cashier scans every item at the store which the customer buys. After the item is sold it is updated in the inventory of the system. If the number of items of the same type goes below certain number in the inventory, the particular item is being ordered from the supplier. It also implements some data mining techniques where it generates rules which can help the manager of the retail store to improve up the current sale of items.

Description: The Retail Store contains items to be sold. It contains employees whose information is stored in the database.

TRIGGERS

VALIDATION RULE TRIGGERS:

1. **Checking the age of employee:**

```
Check the age of Employee
CREATE TRIGGER check_age BEFORE INSERT ON Employees
FOR EACH ROW EXECUTE PROCEDURE ageofemp();

CREATE OR REPLACE FUNCTION ageofemp() RETURNS TRIGGER AS $example_table$
    DECLARE years_old integer;
    BEGIN
            years_old=((CURRENT_DATE-NEW.bdate)/365);
            IF (years_old) < 18 THEN
                    RAISE EXCEPTION 'Underage man';
            END IF;
            return new;
    END;
```

2. **No Late Sale of items:**

```
CREATE TRIGGER no_late_sale BEFORE INSERT ON Transactions
FOR EACH ROW EXECUTE PROCEDURE nolatesaleitems();
CREATE OR REPLACE FUNCTION nolatesaleitems() RETURNS TRIGGER AS
$example_table1$
BEGIN
    if (CURRENT_DATE) > (to_date(to_char(CURRENT_DATE,'MM-DD-YYYY') || '
20:00:00', 'MM-DD-YYYY HH24:MI:SS')) then
            INSERT INTO Errorlog values (SEQ_USER_ID.nextval, 'NO late
sale',CURRENT_DATE);
    END IF;
    return new;
END;
```

BUSINESS RULE TRIGGERS:

1. Checking Inventory:

```
CREATE TRIGGER check_inventory BEFORE UPDATE ON Sales
FOR EACH ROW EXECUTE PROCEDURE checkinv();

CREATE OR REPLACE FUNCTION checkinv() RETURNS TRIGGER AS $example_table2$
DECLARE
        quant integer;
   dnumber integer;
   snumber integer;
   uprice integer;
   tempNum integer;
        i_cur  CURSOR  FOR  SELECT  quantityinhand,deptnumber,suppliernumber,unitprice
from Inventory
        where itemnumber=NEW.itemnumber;
BEGIN
        OPEN i_cur;
   FETCH i_cur into quant, dnumber, snumber, uprice;
        quant = quantityinhand-NEW.qty;
        if (quant  = 10) then
                insert into Orders values(SEQ_USER_ID.nextval, snumber, CURRENT_DATE) ;
                insert                                  into                                  SupplierOrder
values(((SEQ_USER_ID.nextval)),((SEQ_USER_ID.nextval)-2), 2, 50, uprice);
        end if;
        close i_cur;
END;
```

2. Checking the store card:

```
CREATE TRIGGER check_store_card AFTER INSERT ON Transactions
FOR EACH ROW EXECUTE PROCEDURE checkstore();


CREATE OR REPLACE FUNCTION checkstore() RETURNS TRIGGER AS $example_table3$
BEGIN
        if (NEW.storecard = 'y') then
                UPDATE Transactions
                set amount = amount-(amount * 0.1)
```

```
            where transactionid =NEW.transactionid;
        END IF;
END;
```

3.Checking the order:

```
CREATE TRIGGER order_check BEFORE INSERT ON supplierOrder
FOR EACH ROW EXECUTE PROCEDURE checkorder();


CREATE OR REPLACE FUNCTION checkorder() RETURNS TRIGGER AS
$example_table4$
BEGIN
   IF (NEW.qty) > 50 THEN
            INSERT INTO Errorlog values(SEQ_USER_ID.nextval, 'SHOULD NOT ORDER
MORE THAN 50',CURRENT_DATE);
            DELETE FROM orders where orderno=NEW.supplierorderid;
   END IF;
END;
```

4.Cannot return an item which is sold after certain days:

```
CREATE TRIGGER not_return BEFORE INSERT ON Return1
FOR EACH ROW EXECUTE PROCEDURE noreturns();


CREATE OR REPLACE FUNCTION noreturns() RETURNS TRIGGER AS
$example_table5$
   DECLARE
            dt date;
            cur1 CURSOR FOR SELECT dateoftransaction FROM Transactions where
transactionid =NEW.transactionid;
   begin
            OPEN cur1;
            FETCH cur1 into dt;
            if((CURRENT_DATE-dt) > 10) then
```

```
                RAISE EXCEPTION 'you can not return, more than 10 days';
        end if;
        close cur1;
    end;
```

5. Same Manager cannot be allocated to two different location:

```
CREATE TRIGGER mgr_on_loc BEFORE INSERT ON location
FOR EACH ROW EXECUTE PROCEDURE notallowed();
CREATE OR REPLACE FUNCTION notallowed() RETURNS TRIGGER AS $example_table6$
    DECLARE
        id integer;
        errormsg varchar(300);
        cur1 CURSOR FOR select locationnumber from location where
lmgrssn=new.lmgrssn;
    BEGIN
        id=0;
        errormsg='delete';
        OPEN cur1;
        FETCH cur1 into id;
        if (id >0) then
            RAISE EXCEPTION 'errormsg';
        END IF;
        close cur1;
    END;
```

6. Cannot Sell Item on a particular day:

```
CREATE TRIGGER can_not_sell BEFORE INSERT ON Sales
FOR EACH ROW EXECUTE PROCEDURE nosale();
```

```
CREATE OR REPLACE FUNCTION nosale() RETURNS TRIGGER AS $example_table7$
    DECLARE
        desc1 varchar(40);
        p_cur CURSOR FOR select description from PriceLookUp where itemnumber=new.itemnumber;
    BEGIN
    open p_cur;
        FETCH p_cur into desc1;
            RAISE DEBUG 'to_char(CURRENT_DATE,Day)';
            RAISE DEBUG 'desc1';
        if (to_char(CURRENT_DATE,'Day')!=desc1) then
            RAISE DEBUG 'desc1';
        else
            RAISE EXCEPTION 'you can not return, more than 10 days';
        end if;
        close p_cur;
    END;
```

LOGIC RULES:

1. **Determining the Total Sales :**

```
CREATE TRIGGER total_sales BEFORE INSERT ON TotalSale
FOR EACH ROW EXECUTE PROCEDURE totalsales();
CREATE OR REPLACE FUNCTION totalsales() RETURNS TRIGGER AS $example_table8$
    DECLARE
        c1 CURSOR FOR select * from transactions where dateoftransaction=CURRENT_DATE;
        acct c1%ROWTYPE;
        total integer;
    BEGIN
        open c1;
```

```
        total=0;
        for acct in c1 loop
                exit if c1 not found then;
                total= total + acct.amount;
        end loop;
        insert into totalsale values(SEQ_USER_ID.nextval, total, CURRENT_DATE);
END;
```