****

**American University of Sharjah**

**College of Engineering**

**Department of Computer Engineering**

**Database Systems (CMP 320)**

**Project**

**Amir Mohideen - b00074559**

**Karim – b00074792**

**Areej – g00078126**

**Mini world:**

This project will develop a database for owners or managers of online e-commerce watch stores in order to store order information about customers that bought watches from specific stores. Stores will have customers. Customers can buy from stores. Stores will have stocks of different watch brands. Customers will have a unique SSN, first name, last name, email, mobile number, and gender. Stores will have unique store id, store name, website address, and telephone number. Watch will have unique watch id, watch name, and price.

**Functionalities:**

* Will be able to allow users to view all details about customers, online stores and watches
* Insert, update, delete, view customers
* Insert, update, delete, view online stores
* Insert, update, delete, view watches
* Update the order details of customers that bought from a specific store and watch
* Show all orders that were done by a specific customer across all stores and the watches he/she purchased
* Show the most sold watch name and the most sold number
* Have two types of users that can login to the app with following privileges: the admin privilege (only admin can add/update/delete other users) is for owner and the user privilege is for the manager

**ER Diagram:**

Diagram

Description automatically generated

Figure 1 shows the ER diagram of the database that this project aims to develop

**Customer** entity has 6 attributes where SSN, email and mobile are keys.

**Stores** entity has 4 attributes where store\_id, website, and Phone are keys

**Watch** has 3 attributes where Watch\_id is a key

**Customers, Stores,** and **watch Entities** have a M-N-L ternary relationship called orders

Whenever an order is made, the customer that buys from a specific store and specific watch is recorded. Multiple watches can be bought from same store and same person.

**Loginusers** entity has 4 attributes with one key called username

**Schema design:**

A picture containing text, screenshot, monitor

Description automatically generated

Chart

Description automatically generatedFigure 2: Schema Diagram

From the rules studied in this course of translating an ER diagram to schema diagram, we result in the Schema shown in Figure 2.

We have in total 5 tables.

The 3 tables, namely, **Customers**, **Stores** and **Watch** each contain one primary key.

The **orders** table have an artificial primary key order\_id

* The **orders** table holds three foreign keys. The **Cust\_SSN** key references the **Customer** table while the **S\_Num** key references the **Stores** Table and the **wat\_id** key references the watches table.

The **loginusers** table contains 4 attributes where username is a primary key

**SQL Statements used to implement Database schema:**

--customers table has constraints for attributes such as mobile number should start with +9715

--and gender should either be M or F and fields should not be null

--and mobile and email should be unique.

--It has one primary key

CREATE TABLE customers (

Ssn NUMBER(9) CONSTRAINT ssn\_pk PRIMARY KEY,

Fname VARCHAR2(50) CONSTRAINT fname\_nn NOT NULL,

Lname VARCHAR2(50) CONSTRAINT lname\_nn NOT NULL,

Mobile VARCHAR2(13) CONSTRAINT mobile\_ck CHECK(Mobile like ‘+9715\_\_\_\_\_\_\_\_’) constraint mobile\_unq unique,

Email VARCHAR2(25) CONSTRAINT email\_nn NOT NULL constraint email\_unq unique,

Gender CHAR(1) CONSTRAINT gender\_ck CHECK (Gender IN (‘M’, ‘F’))

);

insert into customers values(123456789,’Amir’,’Mohideen’, ‘+971563786304’ ,’amirmohideen99@gmail.com’,’M’ );

insert into customers values(999999999,’Karim’,’Hassan’, ‘+971508823454’ ,’karimhassan99@gmail.com’,’M’ );

insert into customers values(112233445,’Areej’,’Gad’, ‘+971503454333’ ,’areejgad99@gmail.com’,’F’ );

--insert into customers values(112233446,’Ayesha’,’fatima’, ‘+971503454322’ ,’ayeshafatima99@gmail.com’,’F’ );

select \* from customers;

--/%%%%%%%%%%%%%%%%%%%%%%%%%%

--stores table has constraints for attributes such as phone number should start with 06

--and fields should not be null

--and website and phone should be unique.

--It has one primary key.

CREATE TABLE stores (

Store\_Num NUMBER(4) CONSTRAINT storenum\_pk PRIMARY KEY,

Store\_Name VARCHAR2(25) CONSTRAINT storename\_nn NOT NULL, --store can have same name cuz maybe diff website address

Website VARCHAR2(25) CONSTRAINT web\_nn NOT NULL CONSTRAINT web\_unq UNIQUE,

Phone VARCHAR2(9) CONSTRAINT phone\_ck CHECK (Phone like ‘06\_\_\_\_\_\_\_’) CONSTRAINT phone\_unq UNIQUE –two stores cant have same number

);

insert into stores values(1,’Velocity Watches’, ‘velocity-watches.com’ ,’062234543’ );

insert into stores values(2,’City Watches’, ‘citywatches.com’ ,’062224543’ );

insert into stores values(3,’Luxury Watches’, ‘luxury-watches.com’ ,’061234567’ );

insert into stores values(4,’oppo’, ‘oppo.com’ ,’065544320’ );

select \* from stores;

--/%%%%%%%%%%%%%%%%%%%%%%%%%%

--watch\_stock table has constraints for attributes such as

--fields should not be null

--price should be greater than 0

--It has one primary key.

CREATE TABLE watch\_stock(

Watch\_id NUMBER(4) CONSTRAINT watch\_id\_pk PRIMARY KEY,

WatchName VARCHAR2(50) CONSTRAINT watch\_name\_nn NOT NULL,

Price NUMBER(10) CONSTRAINT price\_nn NOT NULL CONSTRAINT price\_ck CHECK (price >= 0)

);

drop table watch\_stock;

insert into watch\_stock values(1,’NISMO’, 200 );

insert into watch\_stock values(2,’GTR’, 200 );

insert into watch\_stock values(3,’Rolex’, 130000 );

select \* from watch\_stock;

--/%%%%%%%%%%%%%%%%%%%%%%%%%%

--The tertiary referencing table. It has 3 foreign key reference having cascade delete function

CREATE TABLE buys\_from2(

Cust\_ssn NUMBER(9) ,

S\_Num NUMBER(4) ,

Wat\_id NUMBER(4),

order\_id NUMBER(4),

primary key(order\_id),

CONSTRAINT bf\_custssn\_fk\_custssn2 FOREIGN KEY (Cust\_ssn) REFERENCES customers (ssn) ON DELETE CASCADE,

CONSTRAINT bf\_snum\_fk\_str\_snum2 FOREIGN KEY (S\_Num) REFERENCES stores (Store\_Num) ON DELETE CASCADE,

CONSTRAINT bf\_watid\_fk\_wat\_watid2 FOREIGN KEY (Wat\_id) REFERENCES watch\_stock (Watch\_id) ON DELETE CASCADE

);

insert into buys\_from2 values(123456789,1,1,300);

insert into buys\_from2 values(123456789,2,2,301);

insert into buys\_from2 values(123456789,2,2,302);

select \* from buys\_from2;

//----stores users of database, username is primary key, fields cant be null

CREATE TABLE loginusers(

username varchar2(25) CONSTRAINT username\_pk PRIMARY KEY,

password varchar2(25) CONSTRAINT pass\_nn NOT NULL,

name varchar2(25) CONSTRAINT name\_nn NOT NULL,

type NUMBER(1) CONSTRAINT type\_nn NOT NULL,

);

insert into loginusers values (‘amir’,’amir’,’Amir’,0 );

insert into loginusers values (‘areej’,’areej’,’areej’,1 );

--Queries used in interface

select order\_id,fname,lname,store\_name,watchname

from customers,watch\_stock,stores,buys\_from2

where customers.ssn = buys\_from2.cust\_ssn

and stores.Store\_Num = buys\_from2.S\_Num

and watch\_stock.watch\_id = buys\_from2.wat\_id

and ssn = ‘123456789’ ;

select count(order\_id) as s,watchname

from customers,watch\_stock,stores,buys\_from2

where customers.ssn = buys\_from2.cust\_ssn

and stores.Store\_Num = buys\_from2.S\_Num

and watch\_stock.watch\_id = buys\_from2.wat\_id

group by watchname

order by count(order\_id) desc;

**Application Interface:**

We developed the app on Apache Netbeans

**Graphical user interface, application

Description automatically generatedLogin menu** allows the user to log in with username and password as admin or normal user. It also has the option to show password or hide. It prints error message if wrong password or username is entered.

Figure 3: login menu

Graphical user interface

Description automatically generated**Main UI menu:**

Figure 4:Main UI Menu

* File tab: Logout button logs out of app and exit button closes app
* orders tab allows mangaer/owner to add,update, delete orders of customers that bought a watch from a store.
* Stats tab allows manager/owner to show all order history about one customer that bought what all watches from which all stores
* Customers tab: allows managers/owners to add, update delete customers
* Stores tab: allows managers/owners to add, update delete stores
* Watch tab: allows mangers/owners to add, update delete watches
* Users tab: allows only owner (admin account) to add, update, delete other users.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 5: Update/Delete Users

Graphical user interface, application

Description automatically generated

Figure 6: View stats

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 7: Update/Delete Watches

Graphical user interface, application

Description automatically generated

Figure 8: Add Stores

Graphical user interface

Description automatically generated

Figure 9: View customers