**Introduction**

**Online shopping** is a form of [electronic commerce](https://en.wikipedia.org/wiki/Electronic_commerce) which allows consumers to directly buy [goods](https://en.wikipedia.org/wiki/Good_(economics)) or [services](https://en.wikipedia.org/wiki/Service_(economics)) from a seller over the [Internet](https://en.wikipedia.org/wiki/Internet) using a [web browser](https://en.wikipedia.org/wiki/Web_browser) or a [mobile app](https://en.wikipedia.org/wiki/Mobile_app). Consumers find a product of interest by visiting the [website](https://en.wikipedia.org/wiki/Website) of the retailer directly or by searching among alternative vendors using a [shopping search engine](https://en.wikipedia.org/wiki/Shopping_search_engine), which displays the same product's availability and pricing at different e-retailers. As of 2020, customers can shop online using a range of different computers and devices, including [desktop computers](https://en.wikipedia.org/wiki/Desktop_computer), [laptops](https://en.wikipedia.org/wiki/Laptop), [tablet computers](https://en.wikipedia.org/wiki/Tablet_computer) and [smartphones](https://en.wikipedia.org/wiki/Smartphone).

An online shop evokes the physical analogy of buying [products](https://en.wikipedia.org/wiki/Product_(business)) or services at a regular ["bricks-and-mortar"](https://en.wikipedia.org/wiki/Brick_and_mortar_business) [retailer](https://en.wikipedia.org/wiki/Retailing) or [shopping center](https://en.wikipedia.org/wiki/Shopping_center); the process is called business-to-consumer (B2C) online shopping. When an online store is set up to enable businesses to buy from another businesses, the process is called [business-to-business](https://en.wikipedia.org/wiki/Business-to-business) (B2B) online shopping. A typical online store enables the customer to browse the firm's range of products and services, view photos or images of the products, along with information about the product specifications, features and prices.

Online stores usually enable shoppers to use "search" features to find specific models, brands or items. Online customers must have access to the Internet and a valid [method of payment](https://en.wikipedia.org/wiki/Online_shopping#Payment) in order to complete a transaction, such as a [credit card](https://en.wikipedia.org/wiki/Credit_card), an [Interac](https://en.wikipedia.org/wiki/Interac" \o "Interac)-enabled [debit card](https://en.wikipedia.org/wiki/Debit_card), or a service such as [PayPal](https://en.wikipedia.org/wiki/PayPal). For physical products (e.g., paperback books or clothes), the e-tailer ships the products to the customer; for digital products, such as [digital audio files](https://en.wikipedia.org/wiki/Digital_audio_file) of [songs](https://en.wikipedia.org/wiki/Song) or [software](https://en.wikipedia.org/wiki/Software), the e-tailer usually sends the file to the customer over the Internet. The largest of these online retailing corporations are [Alibaba](https://en.wikipedia.org/wiki/Alibaba_Group), [Amazon.com](https://en.wikipedia.org/wiki/Amazon.com), and [eBay](https://en.wikipedia.org/wiki/EBay).

**ER DIAGRAM, SCHEMA DIAGRAM**

This chapter of the report describes the structure of the project, followed by Entity Relationship Diagram, Schema Diagram and the table structures.

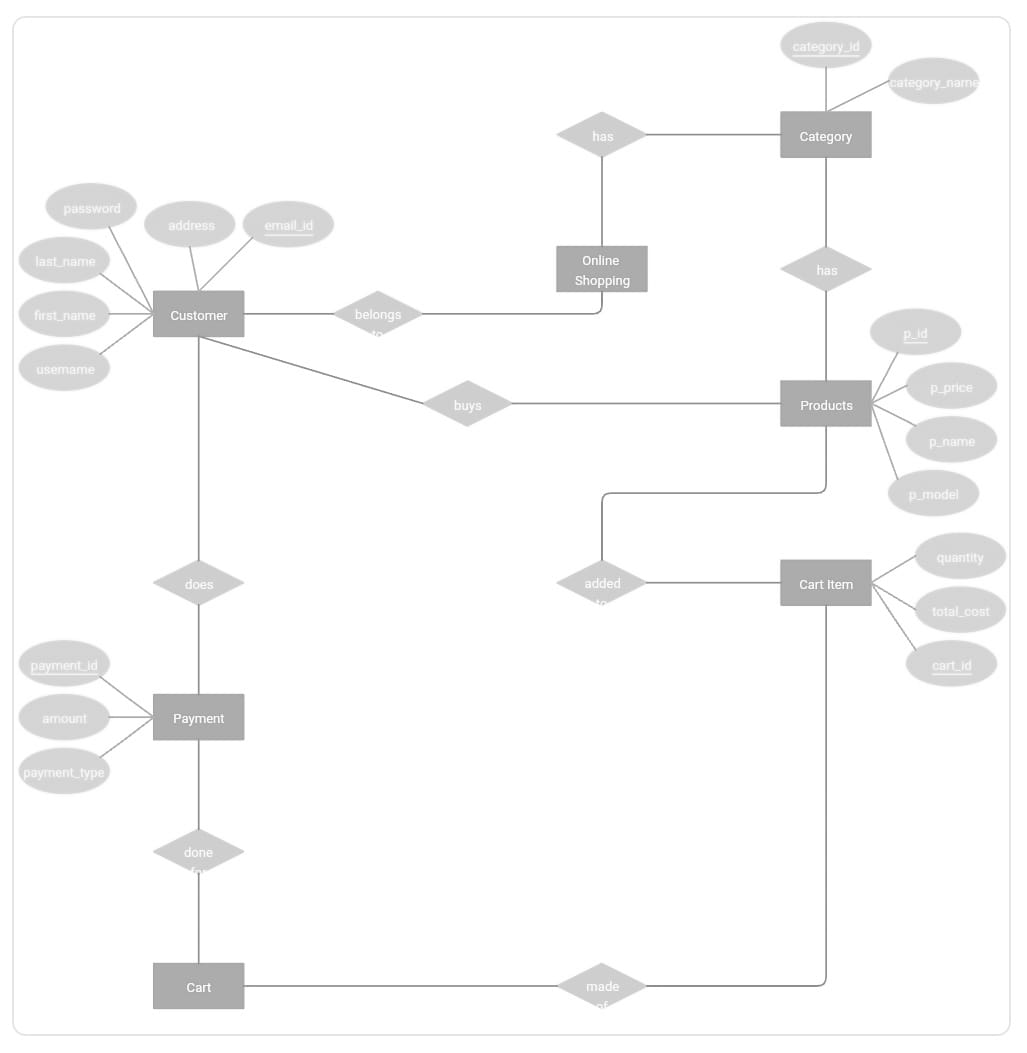
**ER Diagram with relationships and cardinality ratio**

An entity relationship model*,* also called an entity-relationship (ER) diagram*,* is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data-an object or concept about which data is stored.

The cardinalit*y* or fundamental principle of one data aspect with respect to another is a critical feature. The relationship of one to the other must be precise and exact between each other in order to explain how each aspect links together. In simple words Cardinality is a way to define the relationship between two entities.

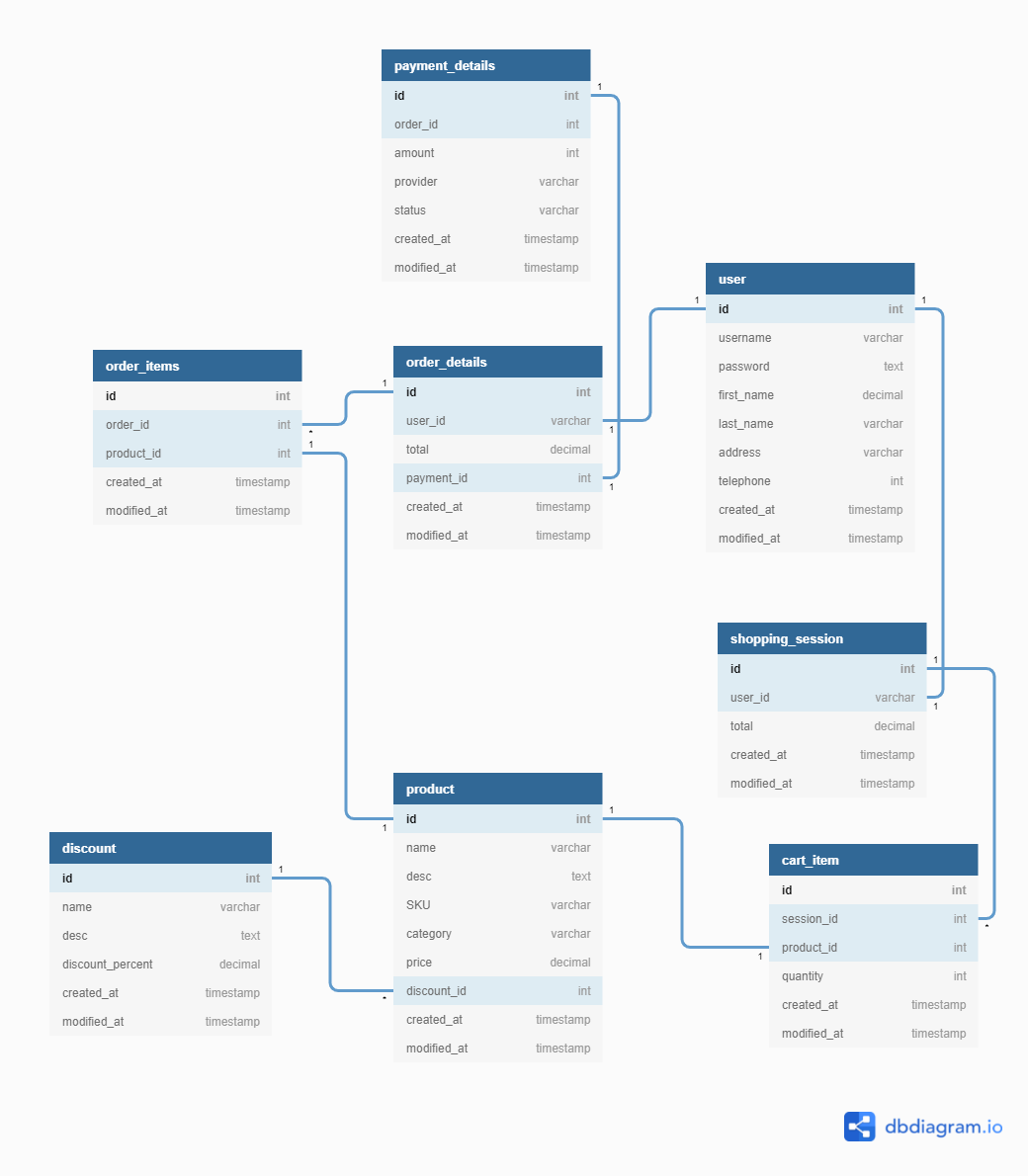
Diagram

Description automatically generatedThe following are the notations of the ER diagram:



**Relational Database**

A relational database is a collection of data items with pre-defined relationships between them. These items are organized as a set of tables with columns and rows. Tables are used to hold information about the objects to be represented in the database. Each column in a table holds a certain kind of data and a field stores the actual value of an attribute. The rows in the table represent a collection of related values of one object or entity. Each row in a table could be marked with a unique identifier called a primary key, and rows among multiple tables can be made related using foreign keys. This data can be accessed in many different ways without reorganizing the database tables themselves.



**TABLE SCREEN SHOTS**

WEEK 1:

In the context of SQL, data definition or data description language (DDL) is a syntax for creating and modifying database objects such as tables, indices, and users.

● CREATE to create a new table or database.

● ALTER for alteration.

● Truncate to delete data from the table.

● DROP to drop a table.

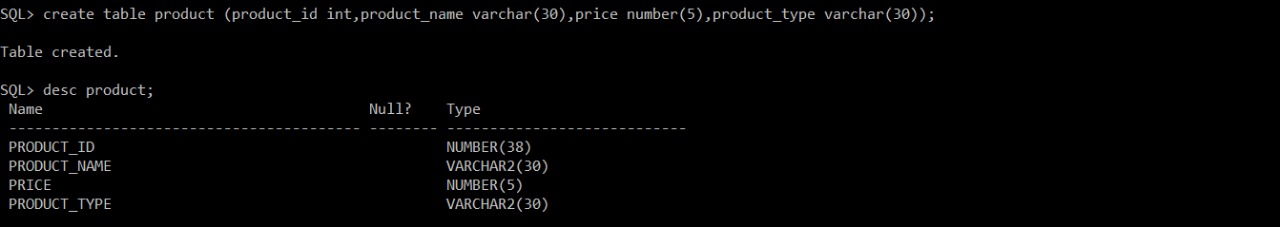
● RENAME to rename a table.

**Product**

SQL> create table product (product\_id int,product\_name varchar(30),price number(5),product\_typevarchar(30));

Table created.

SQL> desc product;

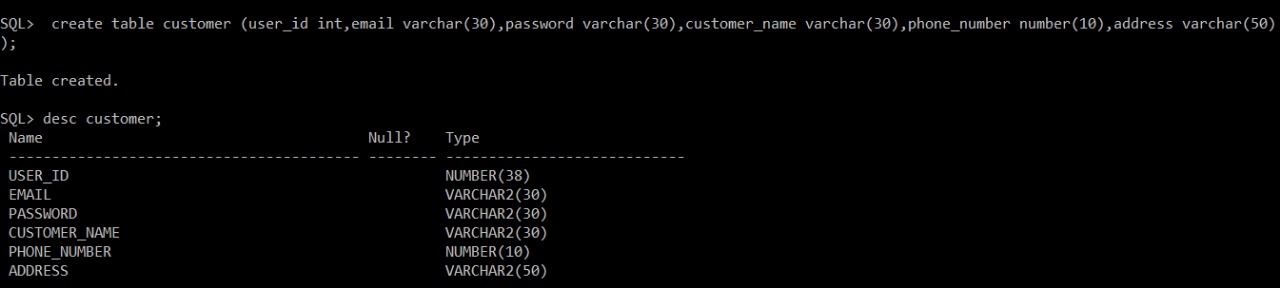


**Customer**

SQL> create table customer (user\_id int,email varchar(30),password varchar(30),customer\_namevarchar(30),phone\_number number(10),address varchar(50));

Table created.

SQL> desc customer;

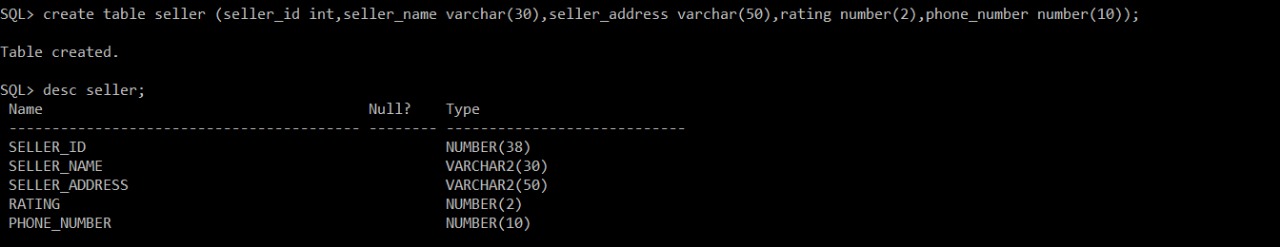


**Seller**

SQL> create table seller (seller\_id int,seller\_name varchar(30),seller\_address varchar(50),rating number(2),phone\_number number(10));

Table created.

SQL> desc seller;

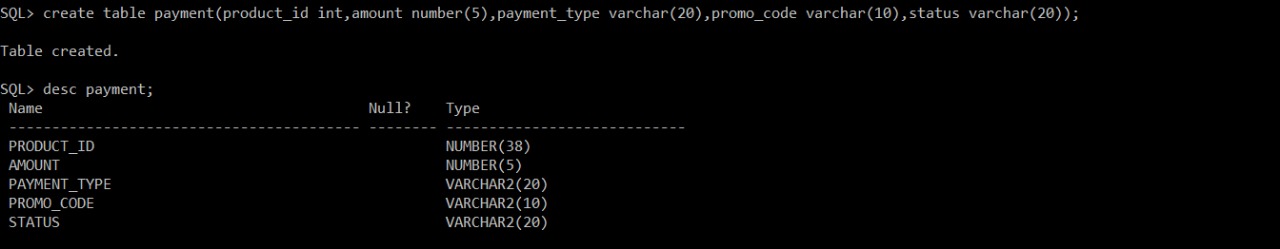


**Payment**

SQL> create table payment(product\_id int,amount number(5),payment\_type varchar(20),promo\_code varchar(10),status varchar(20));

Table created.

SQL> desc payment;

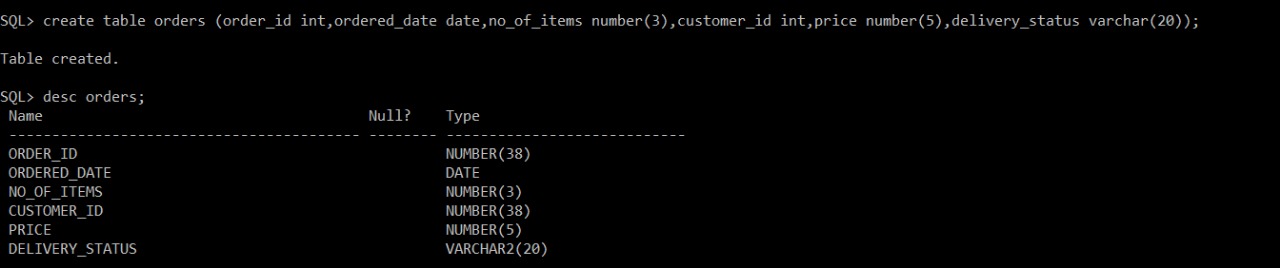


**Orders**

SQL> create table orders (order\_id int,ordered\_date date,no\_of\_items number(3),customer\_id int,price number(5),delivery\_status varchar(20));

Table created.

SQL> desc orders;

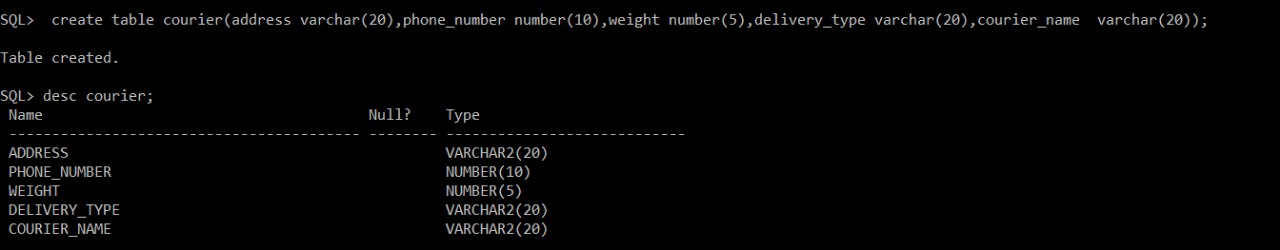


**Courier**

SQL> create table courier(address varchar(20),phone\_number number(10),weight number(5),delivery\_type varchar(20),courier\_name varchar(20));

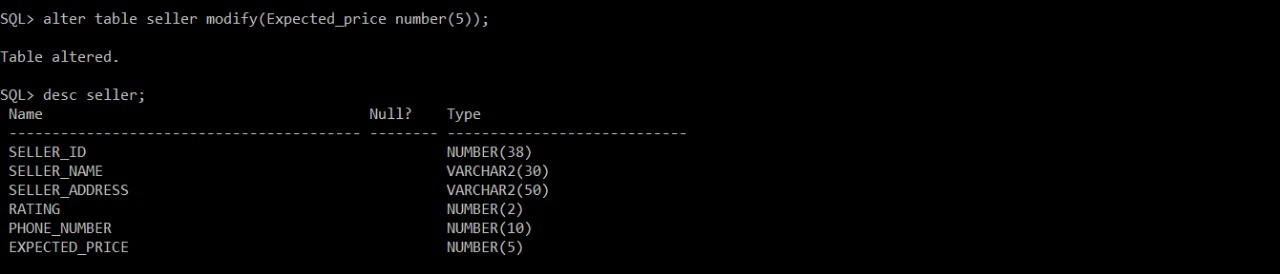
Table created.

SQL> desc courier;



SQL> alter table seller modify(Expected\_price number(5));

Table altered.

SQL>descseller;

**WEEK 2:**

A data manipulation language (DML) is a computer programming language used for adding (inserting), deleting, and modifying (updating) data in a database.

● INSERT – is used to insert data into a table.

● UPDATE – is used to update existing data within a table.

● DELETE – is used to delete records from a database table

SQL>Insertintoproducts values(&product\_id,'&product\_name',&price,'&product\_type');

Enter value for product\_id: 01

Enter value for product\_name: Lux

Enter value for price: 30

Enter value for product\_type: cosmetic

old1:Insertintoproductsvalues(&product\_id,'&product\_name',&price,'&product\_type')

new 1: Insert into products values(01,'Lux',30,'cosmetic')

1rowcreated. Text

Description automatically generated

SQL> /

Enter value for product\_id: 02

Enter value for product\_name: earpods

Enter value for price: 1000

Enter value for product\_type: electronic

old1: Insert into products values(&product\_id,'&product\_name',&price,'&product\_type')

new 1: Insert into products values(02,'earpods',1000,'electronic')

1 row created.

Graphical user interface

Description automatically generated with medium confidence

SQL> /

Enter value for product\_id: 03

Enter value for product\_name: song tv

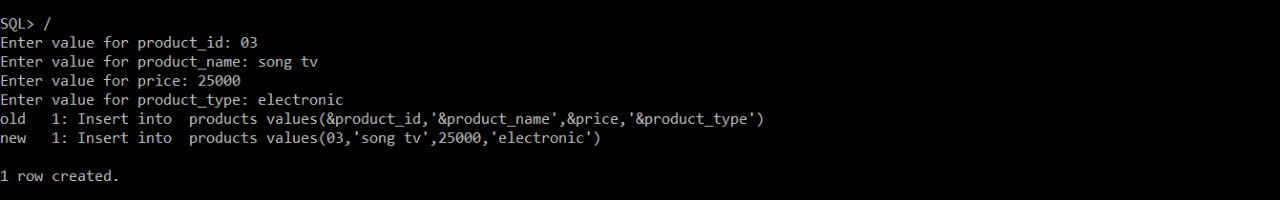
Enter value for price: 25000

Enter value for product\_type: electronic

old1:Insert into products values(&product\_id,'&product\_name',&price,'&product\_type')

new 1: Insert into products values(03,'song tv',25000,'electronic')

1 row created.



Enter value for product\_id: 04

Enter value for product\_name: scorpion helmet

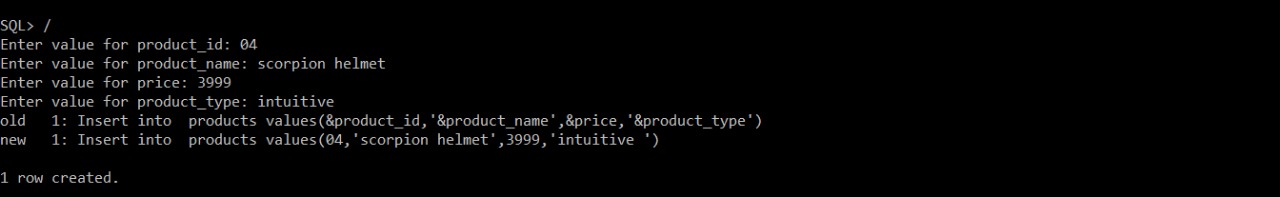
Enter value for price: 3999

Enter value for product\_type: intuitive

old1:Insertinto products values(&product\_id,'&product\_name',&price,'&product\_type')

new 1: Insert into products values(04,'scorpion helmet',3999,'intuitive ')

1 row created.



SQL> select \* from products;



SQL> \_

Enter value for product\_id: 01

Enter value for amount: 20

Enter value for payment\_type: UPI

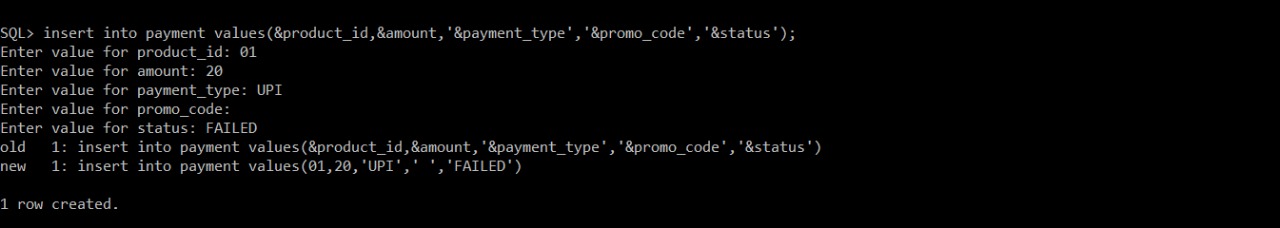
Enter value for promo\_code:

Enter value for status: FAILED

old 1: insert into payment values(&product\_id,&amount,'&payment\_type','&promo\_code','&status')

new 1: insert into payment values(01,20,'UPI',' ','FAILED')

1 row created.



SQL> /

Enter value for product\_id: 02

Enter value for amount: 1000

Enter value for payment\_type: cashondelivery

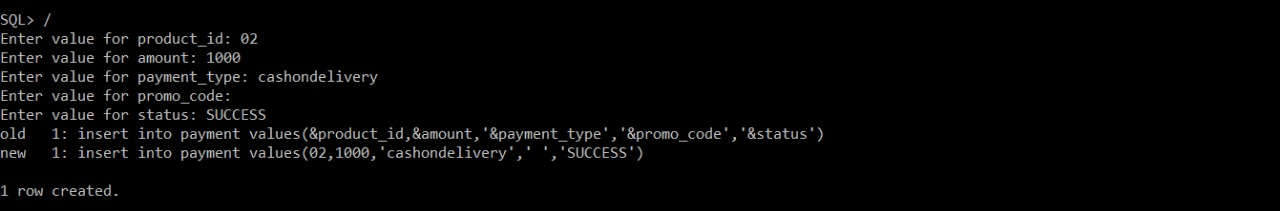
Enter value for promo\_code:

Enter value for status: SUCCESS

old 1: insert into payment values(&product\_id,&amount,'&payment\_type','&promo\_code','&status')

new 1: insert into payment values(02,1000,'cashondelivery',' ','SUCCESS')

1 row created.



SQL> /

Enter value for product\_id: 03

Enter value for amount: 25000

Enter value for payment\_type: UPI

Enter value for promo\_code: HAPPY

Enter value for status: SUCCESS

old 1: insert into payment values(&product\_id,&amount,'&payment\_type','&promo\_code','&status')

new 1: insert into payment values(03,25000,'UPI','HAPPY','SUCCESS')

1 row created.

Text

Description automatically generated with medium confidence

SQL> /

Enter value for product\_id: 04

Enter value for amount: 3999

Enter value for payment\_type: cashondelivery

Enter value for promo\_code: JOURNEY

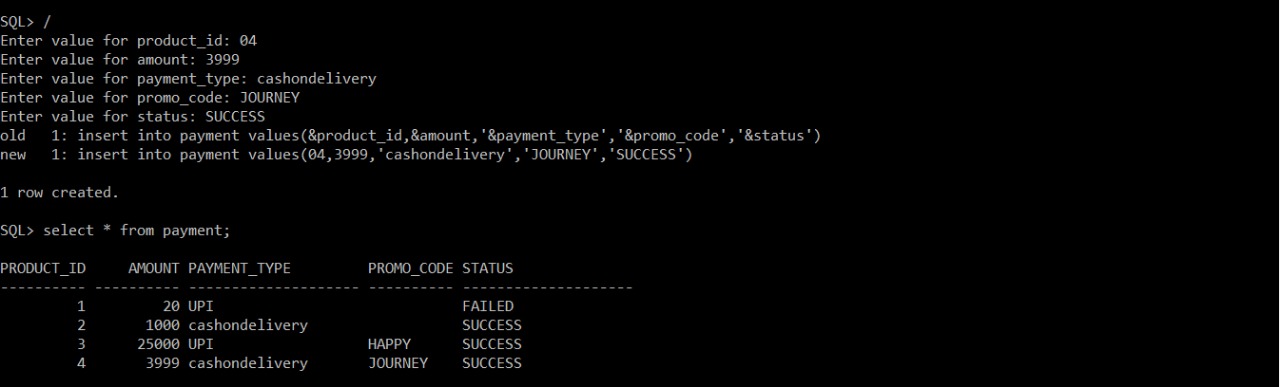
Enter value for status: SUCCESS

old 1: insert into payment values(&product\_id,&amount,'&payment\_type','&promo\_code','&status')

new 1: insert into payment values(04,3999,'cashondelivery','JOURNEY','SUCCESS')

1 row created.

SQL> select \* from payment;



SQL> update payment set status = 30 where status='FAILED';

1 row updated.

SQL> select \* from payment;

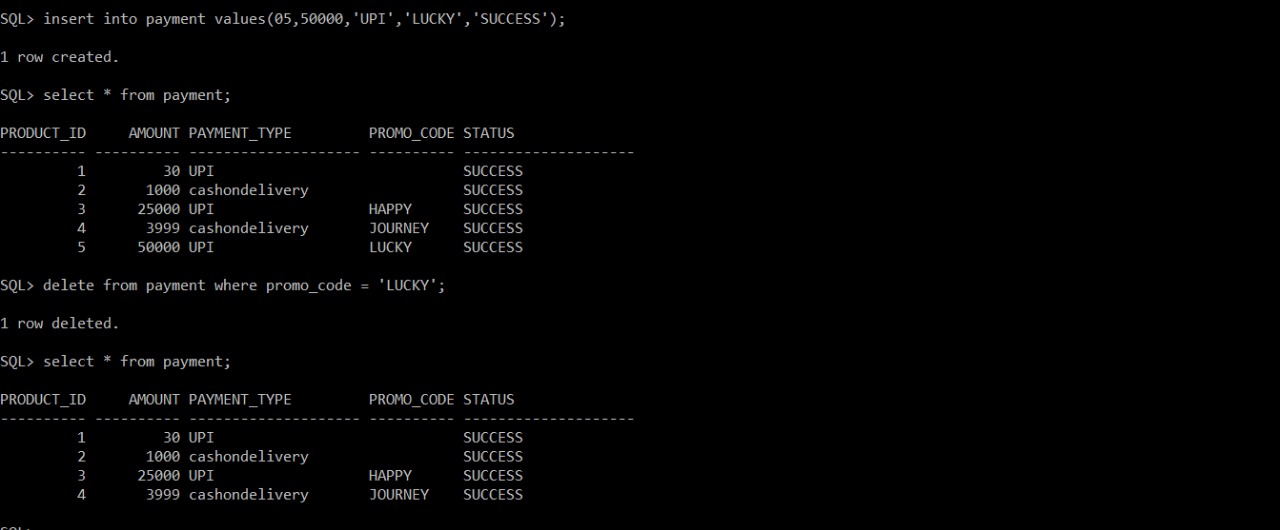
Graphical user interface

Description automatically generated with medium confidence

SQL> insert into payment values(05,50000,'UPI','LUCKY','SUCCESS');

1 row created.

SQL> select \* from payment;



**WEEK 4(TCL)**

SQL> select \* from payment;

PRODUCT\_ID AMOUNT PAYMENT\_TYPE PROMO\_CODE STATUS

---------- ---------- -------------------- ---------- --------------------

1 30 UPI SUCCESS

2 1000 cashondelivery SUCCESS

3 25000 UPI HAPPY SUCCESS

4 3999 cashondelivery JOURNEY SUCCESS

SQL> commit;

Commit complete.

SQL> update payment set amount=2000 where product\_id=2;

1 row updated.

SQL> select \* from payment;

PRODUCT\_ID AMOUNT PAYMENT\_TYPE PROMO\_CODE STATUS

---------- ---------- -------------------- ---------- --------------------

1 30 UPI SUCCESS

2 2000 cashondelivery SUCCESS

3 25000 UPI HAPPY SUCCESS

4 3999 cashondelivery JOURNEY SUCCESS

SQL> rollback;

Rollback complete.

SQL> select \* from payment;

PRODUCT\_ID AMOUNT PAYMENT\_TYPE PROMO\_CODE STATUS

---------- ---------- -------------------- ---------- --------------------

1 30 UPI SUCCESS

2 1000 cashondelivery SUCCESS

3 25000 UPI HAPPY SUCCESS

4 3999 cashondelivery JOURNEY SUCCESS

SQL> savepoint s1;

Savepoint created.

SQL> delete from payment where amount = 3999;

1 row deleted.

SQL> select \* from payment;

PRODUCT\_ID AMOUNT PAYMENT\_TYPE PROMO\_CODE STATUS

---------- ---------- -------------------- ---------- --------------------

1 30 UPI SUCCESS

2 1000 cashondelivery SUCCESS

3 25000 UPI HAPPY SUCCESS

SQL> rollback to s1;

Rollback complete.

SQL> select \* from payment;

PRODUCT\_ID AMOUNT PAYMENT\_TYPE PROMO\_CODE STATUS

---------- ---------- -------------------- ---------- --------------------

1 30 UPI SUCCESS

2 1000 cashondelivery SUCCESS

3 25000 UPI HAPPY SUCCESS

4 3999 cashondelivery JOURNEY SUCCESS

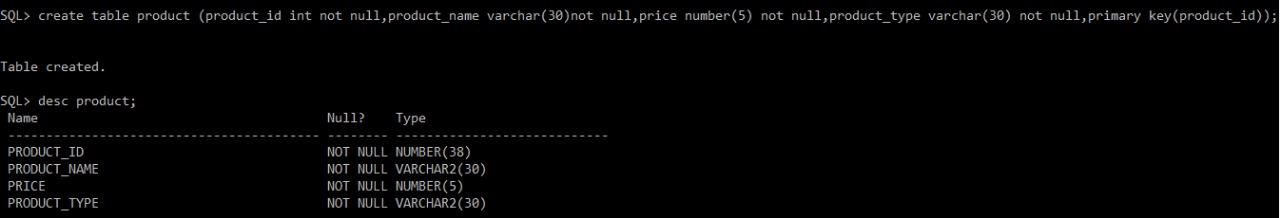
**WEEK 5(SQL CONSTRAINT)**

***Products***

SQL> create table products (product\_id int not null,product\_name varchar(30)not null,price number(5) not null,product\_type varchar(30) not null,primary key(product\_id));

Table created.

SQL> desc product;

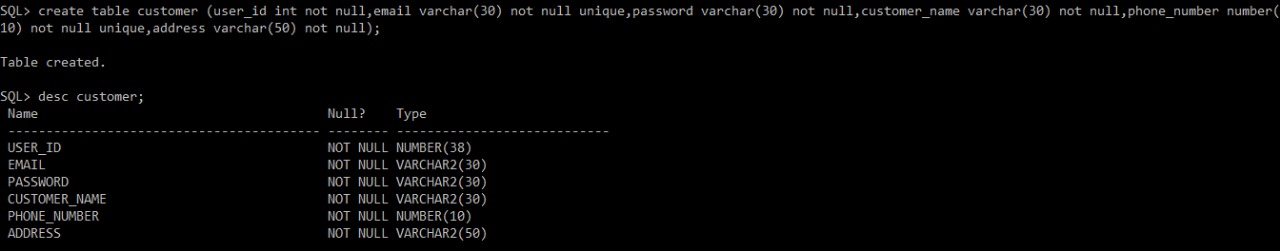


***Customer***

SQL> create table customer (user\_id int not null,email varchar(30) not null unique,password varchar(30) not null,customer\_name varchar(30) not null,phone\_number number(10) not null unique,address varchar(50) not null);

Table created.

SQL> desc customer;

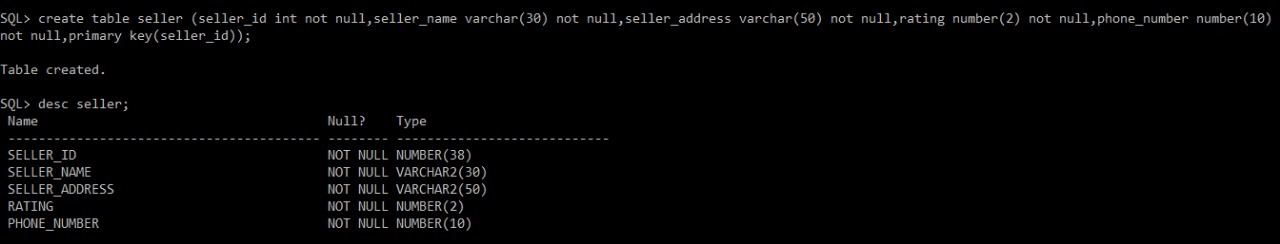


***Seller***

SQL> create table seller (seller\_id int not null,seller\_name varchar(30) not null,seller\_address varchar(50) not null,rating number(2) not null,phone\_number number(10) not null,primary key(seller\_id));

Table created.

SQL> desc seller;

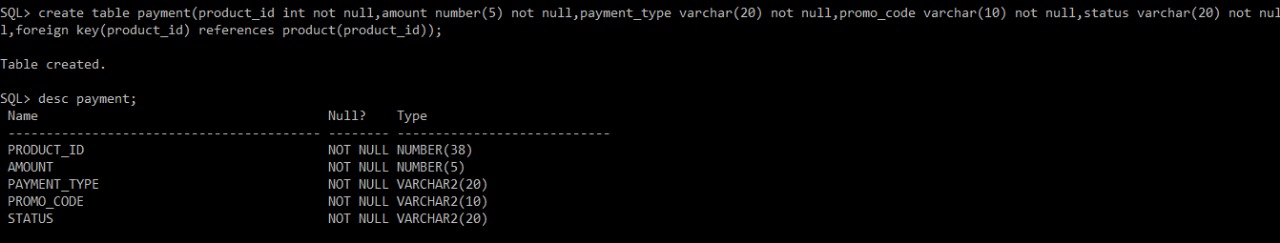


***Payment***

SQL> create table payment(product\_id int not null,amount number(5) not null,payment\_type varchar(20) not null,promo\_code varchar(10) not null,status varchar(20) not null,foreign key(product\_id) references products(product\_id));

Table created.

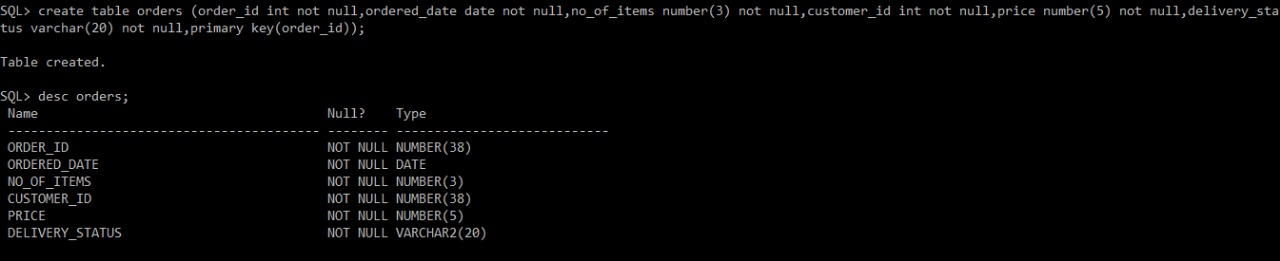
SQL> desc payment;



create table orders (order\_id int not null,ordered\_date date not null,no\_of\_items number(3) not null,customer\_id int not null,price number(5) not null,delivery\_status varchar(20) not null,primary key(order\_id));

Table created.

SQL> desc orders;

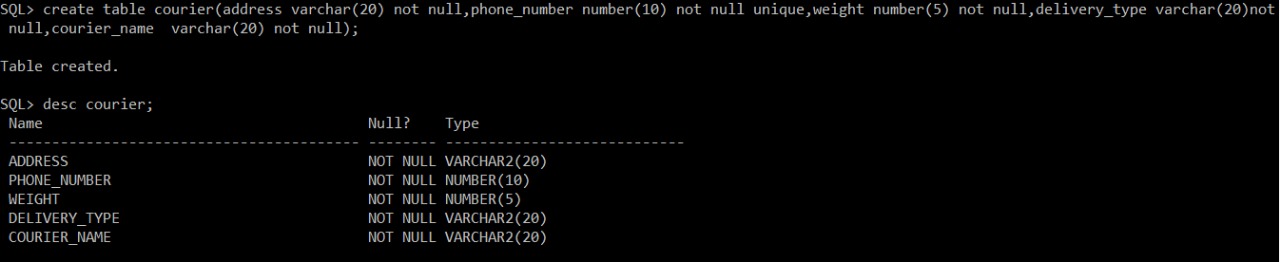


***Courier+***

SQL> create table courier(address varchar(20) not null,phone\_number number(10) not null unique,weight number(5) not null,delivery\_type varchar(20)not null,courier\_name varchar(20) not null);

Table created.

SQL> desc courier;



WEEK 6 (INBUILT FUNCTION)

SQL> select upper(product\_name) from products;

UPPER(PRODUCT\_NAME)

------------------------------

LUX

EARPODS

SONG TV

SCORPION HELMET

SQL> select lower(payment\_type) from payment;

LOWER(PAYMENT\_TYPE)

--------------------

upi

cashondelivery

upi

cashondelivery

SQL> select lpad(product\_name,20,'\*' ) from products;

LPAD(PRODUCT\_NAME,20,'\*')

--------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Lux

\*\*\*\*\*\*\*\*\*\*\*\*\*earpods

\*\*\*\*\*\*\*\*\*\*\*\*\*song tv

\*\*\*\*\*scorpion helmet

SQL> select rpad(product\_name,20,'\*' ) from products;

RPAD(PRODUCT\_NAME,20,'\*')

--------------------------------------------------------------------------------

Lux\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

earpods\*\*\*\*\*\*\*\*\*\*\*\*\*

song tv\*\*\*\*\*\*\*\*\*\*\*\*\*

scorpion helmet\*\*\*\*\*