Evaluation/Assessment Criteria:-

Ex.No	Aim & Algorithm (5)	Program (8)	Result &Output (4)	Viva (3)	Total (20)
1.					
2.					
3.					
4.					
5.					
6.					
7.					
8.					
9.					
10.					
11.					

BASIC SQL COMMANDS

CREATE:

create table employee(eno number(5),ename vaechar2(10),esal number(5));
Table created.

DESCRIPTION:

desc employee;

Name	Null?	Type
ENO		NUMBER(5)
ENAME		VARCHAR2(10)
ESAL		NUMBER(5)

ALTER:

alter table employee add(address varchar2(25));
Table altered.
desc employee;

Name	Null?	Type
ENO		NUMBER
ENAME		VARCHAR2(10)
ESAL		NUMBER(5)
ADDRESS		VARCHAR2(25)

INSERT:
insert into employee values(&eno,'&ename',&esal,'&address');
Enter value for eno:101
Enter value for ename:asha
Enter value for esal:10000
Enter value for address:london
1 row created.

RETRIEVE:
select * from employee;

ENO	ENAME	ESAL	ADDRESS
101	asha	10000	london
102	nasrin	20000	america
103	geetha	30000	indonesia

DELETE:
delete from stud;
3 rows deleted
select * from stud;
no rows selected

DDL AND DML COMMANDS

DDL COMMANDS:

CREATE:

create table employee(name varchar2(20), email varchar2(100), dob date);

DROP

drop table employee;

ALTER

alter table stu_details add(address varchar2(20));

alter table stu_details modify (name varchar2(20));

TRUNCATE:

truncate table employee;

RENAME:

rename employee to emp1;

Table renamed

SQL>desc emp1;

Name	Null?	Type
ENO		NUMBER(5)
ENAME		VARCHAR2(10)
ESAL		NUMBER(5)
ADDRESS		VARCHAR2(15)

DML COMMANDS:

INSERT

create table stud(regno number(5),name varchar2(15),mark number(5),dept varchar2(15));
Table created.
desc stud;

Name	Null?	Type
REGNO		NUMBER(5)
NAME		VARCHAR2(15)
MARK		NUMBER(5)
DEPT		VARCHAR2(15)

insert into stud values(&regno,'&name',&mark,'&dept');
Enter value for regno:1
Enter value for name:asha
Enter value for mark:95
Enter value for dept:cs

1 row created.

select * from stud;

REGNO	NAME	MARK	DEPT
1	asha	95	cs
2	vinodhini	93	IT
3	nasrin	92	EEE

insert a particular column in a single row

insert into stud(regno,name,mark,dept)values('4','priya','96','tamil');
1 row created

SELECT:
i)Select all columns in a table

select * from stud;

REGNO	NAME	MARK	DEPT
1	asha	95	cs
2	vinodhini	93	IT
3	nasrin	92	EEE
4	priya	96	Tamil

ii)Select particular column in a table
select regno,name from stud;

REGNO	NAME
1	asha
2	vinodhini

iii)Select distinct values in particular column

select distinction(mark) from stud;

```
MARK
------
 92
 93
 95
 96
```

iv)Select a particular Row and Column in Table
select regno,mark from stud where mark=96;

```
REGNO     MARK
-------   ------
  4         96
```

UPDATE
i)update stud set dept='maths' where mark=95;

1 row updated

select * from stud;

```
REGNO    NAME       MARK      DEPT
------   --------   -------   -------
1        asha       95        maths
2        vinodhini  93        IT
3        nasrin     92        EEE
4        priya      96        Tamil
```

ii)Update all records in table
update stud set dept='cs';

4 rows updated
select * from stud;

```
REGNO     NAME       MARK     DEPT
------    ------     ------   -------
 1        asha       95       cs
 2        vinodhini  93       cs
 3        nasrin     92       cs
 4        priya      96       cs
```

DELETE
i)Delete a particular row in a table

delete from stud where regno=4;

1 row deleted

select * from stud;

REGNO	NAME	MARK	DEPT
1	asha	95	cs
2	vinodhini	93	cs
3	nasrin	92	cs

ii)Delete all records in a table

delete from stud;

3 rows deleted

select * from stud;

no rows selected

TABLE CREATION WITH CONSTRAINTS

NOT NULL:

CREATE TABLE Persons (ID number NOT NULL,LastName varchar(255), Age number);

Table created

insert into persons(ID,LastName,Age)values(1,'babu',20);

1 row created

insert into persons(ID,LastName)values(2,'ashok');

1 row created

insert into persons(LastName,Age)values('chandran',21);

Error: NOT NULL constraint failed: Persons.ID

CHECK CONSTRAINTS:

create table person3(sno number(5),dno number(15)check(dno>10),dname varchar2(10));

Table created

insert into person3(sno,dno,'dname')values(1,5,'aniruth');

Error: CHECK constraint failed: dno>10

insert into person3(sno,dno,'dname')values(2,11,'anusha');

1 row created

UNIQUE KEY:

create table employee4(eno number(5)unique,ename varchar2(15));

Table created

insert into employee4(eno,ename)values(145,'kayal');

1 row created

insert into employee4(eno,ename)values(145,'kagal');

Error: UNIQUE constraint failed: employee4.eno

PRIMARY KEY:

create table employee5(no number(5) NOT NULL primary key,name varchar2(15),sal number(10));

Table created

insert into employee5(no,name,sal)values(1,'abu',50000);

1 row created

insert into employee5(no,name,sal)values(1,'babu',25000);

Error: UNIQUE constraint failed: employee5.no

create table college1(college_id int,college_code varchar(20) not null,college_name varchar(50),

constraint collegepk primary key (college_id,college_code));

Table created

FOREIGN KEY:

CREATE TABLE Customers (id INT,first_name VARCHAR(40),last_name VARCHAR(40),age INT,
country VARCHAR(10),CONSTRAINT CustomersPK PRIMARY KEY (id));
Table created

CREATE TABLE Orders (order_id INT,product VARCHAR(40),total INT,customer_id INT,CONSTRAINT OrdersPK PRIMARY KEY (order_id),FOREIGN KEY (customer_id) REFERENCES Customers(id));
Table created

INSERT INTO Customers VALUES(1, 'John', 'Doe', 31, 'USA'),(2, 'Robert', 'Luna', 22, 'USA');

2 rows created

INSERT INTO Orders VALUES(1, 'Keyboard', 400, 2),(2, 'Mouse', 300, 2),(3, 'Monitor', 12000, 1);

3 rows created

INSERT INTO Orders VALUES(4, 'Monitor', 12000, 3);

Error: FOREIGN KEY constraint failed

DEFAULT CONSTRAINTS

create table stud(rno number(5),name varchar2(10),avg number(4),result varchar2(15)default('pass'));
Table created

insert into stud(rno,name,avg)values(111,'asha',75);

1 row created

select * from stud;

RNO	NAME	AVG	RESULT
111	asha	75	pass

JOINS AND VIEWS

VIEWS:

TO CREATE THE TABLE 'FVIEWS':-

```
create table fviews(name varchar2(20),no number(5),sal number(5), dno number(5));
```

Table created.

TO INSERT THE VALUES INTO 'FVIEWS':-

```
insert into fviews values('xxx',1,19000,11);
```

1 row created.

```
insert into fviews values('aaa',2,19000,12);
```

1 row created.

```
insert into fviews values('yyy',3,40000,13);
```

1 row created.

```
select * from fviews;
```

NAME	NO	SAL	DNO
xxx	1	19000	11
aaa	2	19000	12
yyy	3	40000	13

TO CREATE THE TABLE 'DVIEWS':-

```
create table dviews( dno number(5), dname varchar2(20));
```

Table created.

TO INSERT THE VALUES INTO 'DVIEWS':-

```
insert into dviews values(11,'x');
```

1 row created.

```
insert into dviews values(12,'y');
```

1 row created.

```
select * from dviews;
```

DNO	DNAME
11	x
12	y

CREATING THE VIEW 'SVIEW' ON 'FVIEWS' TABLE:-

```
create view sview as select name,no,sal,dno from fviews where dno=11;
```

View created.

select * from sview;

NAME NO SAL DNO

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

xxx 1 19000 11

insert into sview values ('zzz',4,20000,14);

1 row created.

select * from sview;

NAME NO SAL DNO

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

Xxx 1 19000 11

CREATING A VIEW 'IVIEW' FOR THE TABLE 'FVIEWS':-

create view iview as select * from fviews;

View created.

select * from iview;

NAME	NO	SAL	DNO
xxx	1	19000	11
aaa	2	19000	12
yyy	3	40000	13
zzz	4	20000	14

PERFORMING UPDATE OPERATION:-

insert into iview values ('bbb',5,30000,15);

1 row created.

select * from iview;

NAME NO SAL DNO

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

xxx 1 19000 11

bbb 5 30000 15

select * from fviews;

NAME NO SAL DNO

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

xxx 1 19000 11

aaa 2 19000 12

yyy 3 40000 13

zzz 4 20000 14

bbb 5 30000 15

CREATE A NEW VIEW 'SSVIEW' AND DROP THE VIEW

create view ssview(cusname,id) as select name, no from fviews where dno=12;

View created.

select * from ssview;

CUSNAME ID

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

Aaa 2

drop view ssview;

View dropped.

TO CREATE A VIEW 'COMBO' USING BOTH THE TABLES 'FVIEWS' AND 'DVIEWS'

create view combo as select name,no,sal,dviews.dno,dname from fviews,dviews where fviews.dno=dviews.dno;

View created.

select * from combo;

NAME	NO	SAL	DNO	DNAME
xxx	1	19000	11	x
aaa	2	19000	12	y

TO PERFORM MANIPULATIONS ON THIS VIEW

insert into combo values('ccc',12,1000,13,'x');

insert into combo values('ccc',12,1000,13,'x')

*ERROR at line 1:

ORA-01779: cannot modify a column which maps to a non key-preserved table

This shows that when a view is created from two different tables no manipulations can be performed using that view and the above error is displayed.

select * from fviews;

NAME	NO	SAL	DNO
Xxx	1	19000	11
aaa	2	19000	12
yyy	3	40000	13
zzz	4	20000	14

Updates made on the view are reflected on both the view and the table when the structure of the table and the view are similar – proof

JOINS

CREATING TABLES FOR DOING JOIN OPERATIONS

TO CREATE SSTUD1 TABLE:-

```
create table sstud1 ( sname varchar2(20) , place varchar2(20));

Table created.

insert into sstud1 values ( 'prajan','chennai');

1 row created.

insert into sstud1 values ( 'anand','chennai');

1 row created.

insert into sstud1 values ( 'kumar','chennai');

1 row created.

insert into sstud1 values ( 'ravi','chennai');

1 row created.

select * from sstud1;
```

SNAME	PLACE
prajan	chennai
anand	chennai
kumar	chennai
ravi	chennai

TO CREATE SSTUD2 TABLE:-

```
create table sstud2 ( sname varchar2(20), dept varchar2(10), marks number(10));

Table created.

insert into sstud2 values ('prajan','cse',700);

1 row created.

insert into sstud2 values ('anand','it',650);

1 row created.

insert into sstud2 values ('vasu','cse',680);

1 row created.

insert into sstud2 values ('ravi','it',600);

1 row created.

select * from sstud2;
```

SNAME	DEPT	MARKS
Prajan	cse	700
anand	it	650
vasu	cse	680
ravi	it	600

select sstud1.sname, dept from sstud1 inner join sstud2 on (stud1.sname=sstud2.name) ;

SNAME	DEPT
Anand	it
Prajan	cse
ravi	it

select sstud1.sname, dept from sstud1 join sstud2 on (sstud1.sname= sstud2.sname);

SNAME	DEPT
anand	it
prajan	cse
ravi	it

select sstud1.sname, dept from sstud1 left outer join sstud2 on (sstud1.sname= sstud2.sname);

SNAME	DEPT
prajan	cse
anand	it
ravi	it

select sstud1.sname, dept from sstud1 right outer join sstud2 on (sstud1.sname= sstud2.sname);

SNAME	DEPT
prajan	cse
anand	it
ravi	it

select sstud1.sname, dept from sstud1 full outer join sstud2 on (sstud1.sname= sstud2.sname);

SNAME	DEPT
Prajan	cse
anand	it
ravi	it
kumar	cse

PL/SQL - PROCEDURES

create table stud(rno number(2),mark1 number(3),mark2 number(3),total number(3),primary key(rno));
Table created.

desc stud;
Name Null? Type
RNO NOT NULL NUMBER(2)
MARK1 NUMBER(3)
MARK2 NUMBER(3)
TOTAL NUMBER(3)

select * from stud;

RNO	MARK1	MARK2	TOTAL
1	80	85	0
2	75	84	0
3	65	80	0
4	90	85	0

SQL> create or replace procedure studd (rnum number) is
2 m1 number;
3 m2 number;
4 total number;
5 begin
6 select mark1,mark2 into m1,m2 from stud where rno=rnum; 7 if m1<m2 then
8 update stud set total=m1+m2 where rno=rnum;
9 end if;
10 end;
11 /

Procedure created.

exec studd(1);

PL/SQL procedure successfully completed.

select * from stud;

RNO	MARK1	MARK2	TOTAL 1
1	80	85	165
2	75	84	0
3	65	80	0
4	90	85	0

exec studd(4);
PL/SQL procedure successfully completed.

select * from stud;

RNO	MARK1	MARK2	TOTAL 1
1	80	85	165
2	75	84	0
3	65	80	0
4	90	85	0

exec studd(2);
PL/SQL procedure successfully completed.

exec studd(3);
PL/SQL procedure successfully completed.

select * from stud;

RNO	MARK1	MARK2	TOTAL 1
1	80	85	165
2	75	84	159
3	65	80	145
4	90	85	0

CURSORS

create table employe(eid number(4),fname varchar2(10),lname varchar2(10),joindate date,jobid varchar2(15),salary number(10),deptid number(5));

Table created.

desc employe;

Name	Null?	Type
EID		NUMBER(4)
FNAME		VARCHAR2(10)
LNAME		VARCHAR2(10)
JOINDATE		DATE
JOBID		VARCHAR2(15)
SALARY		NUMBER(10)
DEPTID		NUMBER(5)

insert into employe values(100,'permila','rosy','25-may-1995','itprogrammer',55000,10);

1 row created.

insert into employe values(101,'john','son','19-aug-1994','account',50000,20);

1 row created.

insert into employe values(102,'Adhitya','Birla','9-jun-1972','GM',150000,30);

1 row created.

insert into employe values(102,'Kamal','Hasan','30-Dec-1960','ADpress',85000,40);

1 row created.

insert into employe values(103,'James','vasanth','20-Oct-1970','ADvp',45000,50);

1 row created.

insert into employe values(104,'James','William','28-Sep-2001','Itprogrammer',40000,10);

1 row created.

insert into employe values(105,'Sarath','William','23-Jul-1989','account',70000,20);

1 row created.

insert into employe values(106,'prema','latha','20-Aug-1999','AGM',75000,60);

1 row created.

insert into employe values(107,'kavi','malar','05-Apr-2003','ADpress',40000,40);

1 row created.

insert into employe values(108,'mohammed','ismail','12-jan-2000','ADvp',20000,50);

1 row created.

insert into employe values(109,'James','king','27-mar-1998','itprogrammer',40000,10);

1 row created.

select * from employe;

EID	FNAME	LNAME	JOINDATE	JOBID	SALARY	DEPTID
100	permila	rosy	25-MAY-95	itprogrammer	55000	10
101	john	son	19-AUG-94	account	50000	20
102	Adhitya	Birla	09-JUN-72	GM	150000	30
102	Kamal	Hasan	30-DEC-60	ADpress	85000	40
103	James	vasanth	20-OCT-70	ADvp	45000	50
104	James	William	28-SEP-01	Itprogrammer	40000	10
105	Sarath	William	23-JUL-89	account	70000	20
106	prema	latha	20-AUG-99	AGM	75000	60
107	kavi	malar	05-APR-03	ADpress	40000	40
108	mohammed	ismail	12-JAN-00	ADvp	20000	50
109	James	king	27-MAR-98	itprogrammer	40000	10

11 rows selected.

IMPLICIT CURSOR

```
SQL> set serveroutput on
SQL> DECLARE
        total_rows number(10);
     BEGIN
     UPDATE employe
     SET salary = salary + 500;
     IF sql%notfound THEN
       dbms_output.put_line('no employees updated');
     ELSIF sql%found THEN
       total_rows := sql%rowcount;
      dbms_output.put_line( total_rows || ' employees were updated ');
     END IF;
   END;
   /
11 employees were updated
PL/SQL procedure successfully completed.
```

```
select * from employe;
```

EID	FNAME	LNAME	JOINDATE	JOBID	SALARY	DEPTID
100	permila	rosy	25-MAY-95	itprogrammer	55500	10
101	john	son	19-AUG-94	account	50500	20
102	Adhitya	Birla	09-JUN-72	GM	150500	30
102	Kamal	Hasan	30-DEC-60	ADpress	85500	40
103	James	vasanth	20-OCT-70	ADvp	45500	50
104	James	William	28-SEP-01	Itprogrammer	40500	10
105	Sarath	William	23-JUL-89	account	70500	20
106	prema	latha	20-AUG-99	AGM	75500	60
107	kavi	malar	05-APR-03	ADpress	40500	40
108	mohammed	ismail	12-JAN-00	ADvp	20500	50
109	James	king	27-MAR-98	itprogrammer	40500	10

11 rows selected.

EXPLICIT CURSOR

```
SQL> set serveroutput on
SQL> DECLARE
  2    e_id employe.eid%type;
  3    e_fname employe.fname%type;
  4    e_jobid employe.jobid%type;
  5    CURSOR e_employe is
  6       SELECT eid, fname, jobid FROM employe;
  7  BEGIN
  8    OPEN e_employe;
  9    LOOP
 10     FETCH e_employe into e_id, e_fname, e_jobid;
 11       EXIT WHEN e_employe%notfound;
 12       dbms_output.put_line(e_id || ' ' || e_fname || ' ' || e_jobid);
 13    END LOOP;
 14    CLOSE e_employe;
 15  END;
 16  /

100   permila   itprogrammer
101    john      account
102   Adhitya    GM
102    Kamal     ADpress
103    James     ADvp
104    James     Itprogrammer
105    Sarath    account
106    prema     AGM
107    kavi      ADpress
108   mohammed   ADvp
109    James     itprogrammer

PL/SQL procedure successfully completed.
```

TRIGGERS AND FUNCTIONS

```
create table itempls (ename varchar2(10), eid number(5), salary number(10));

Table created.

insert into itempls values('xxx',11,10000);

1 row created.

insert into itempls values('yyy',12,10500);

1 row created.

insert into itempls values('zzz',13,15500);

1 row created.

select * from itempls;

ENAME    EID SALARY
---------- --------- --------------
xxx          11       10000
yyy          12       10500
zzz          13       15500
```

TO CREATE A SIMPLE TRIGGER THAT DOES NOT ALLOW INSERT UPDATE AND DELETE OPERATIONS ON THE TABLE:-

```
create trigger ittrigg before insert or update or delete on itempls for each row

2 begin

3 raise_application_error(-20010,'You cannot do manipulation');

4 end;

5 /

Trigger created.
```

DELETE OPERATION:-

```
delete from itempls where ename='xxx';
delete from itempls where ename='xxx'
*
ERROR at line 1:
ORA-20010: You cannot do manipulation
ORA-06512: at "STUDENT.ITTRIGG", line 2
ORA-04088: error during execution of trigger 'STUDENT.ITTRIGG'
```

UPDATE OPERATION:-

```
 update itempls set eid=15 where ename='yyy';
update itempls set eid=15 where ename='yyy'
*
ERROR at line 1:
ORA-20010: You cannot do manipulation
ORA-06512: at "STUDENT.ITTRIGG", line 2
ORA-04088: error during execution of trigger 'STUDENT.ITTRIGG'
```

TO DROP THE CREATED TRIGGER:-

drop trigger ittrigg;

Trigger dropped.

TO CREATE A TRIGGER THAT RAISES AN USER DEFINED ERROR MESSAGE AND DOES NOT ALLOW UPDATION AND INSERTION:-

create trigger ittriggs before insert or update of salary on itempls for each row

declare

2 triggsal itempls.salary%type;

3 begin

4 select salary into triggsal from itempls where eid=12;

5 if(:new.salary>triggsal or :new.salary<triggsal) then

6 raise_application_error(-20100,'Salary has not been changed');

7 end if;

8 end;

9 /

Trigger created.

INSERT OPERATION:-

insert into itempls values ('bbb',16,45000);
insert into itempls values ('bbb',16,45000)
*
ERROR at line 1:
ORA-04098: trigger 'STUDENT.ITTRIGGS' is invalid and failed re-validation

UPDATE OPERATION:-

update itempls set eid=18 where ename='zzz';
update itempls set eid=18 where ename='zzz'
*
ERROR at line 1:
ORA-04298: trigger 'STUDENT.ITTRIGGS' is invalid and failed re-validation

FUNCTION

.

FACTORIAL OF A NUMBER USING FUNCTION — PROGRAM AND EXECUTION:-

```
create function itfact (a number) return number is
fact number:=1;
b number;
begin
b:=a;
while b>0
loop
fact:=fact*b;
b:=b-1;
end loop;
return(fact);
end;
/
```

Function created.

```
SQL> set serveroutput on;
SQL> declare
a number:=7;
f number(10);
begin
f:=itfact(a);
dbms_output.put_line('The factorial of the given number is'||f);
end;
/
```

The factorial of the given number is 5040

PL/SQL procedure successfully completed.

DATA CLEANING AND EXPLORATION

SQL FOR DATA CLEANING

Removing duplicate rows:

```
DELETE FROM Customers
WHERE CustomerID NOT IN (
SELECT MIN(CustomerID)
FROM Customers
GROUP BY Email
HAVING COUNT(*) > 1);
```

3 rows deleted.

Updating incorrect or missing data:

```
UPDATE Customers
SET Email = 'unknown@example.com'
WHERE Email IS NULL;
```

10 rows updated

Removing outliers:

```
DELETE FROM Customers
WHERE CreditLimit > 100000;
```

5 rows deleted.

Replacing NULL values:

```
UPDATE Customers
SET Phone = 'unknown'
WHERE Phone IS NULL;
```

6 rows updated

Standardizing data:

```
UPDATE Customers
SET City = UPPER(City);
UPDATE Customers
SET City = TRIM(City);
```

6 rows updated.

Removing a specific character from the beginning and end of a column:

```
UPDATE Customers
SET Phone = TRIM('+' FROM Phone);
```

Removing multiple characters from the beginning and end of a column:

```
UPDATE Customers
SET Email = TRIM(TRAILING '.' FROM TRIM(LEADING '.' FROM Email));
```

SQL FOR EXPLORATORY DATA ANALYSIS

Counting the number of rows in a table:

```
SELECT COUNT(*) FROM Customers;
COUNT(*)
---------
1000
```

Finding the minimum and maximum values in a column:

```
SELECT MIN(CreditLimit), MAX(CreditLimit) FROM Customers;

MIN(CreditLimit) | MAX(CreditLimit)
---------------- | ----------------
1000             | 50000
```

Finding the average value in a column:

```
SELECT AVG(CreditLimit) FROM Customers;

AVG(CreditLimit)
----------------
25000
```

Finding the sum of a column:

```
SELECT SUM(UnitPrice*Quantity) as TotalSales FROM Orders;


TOTALSALES
-----------
500000
```

Grouping data by a column:

```
SELECT CustomerID, SUM(UnitPrice*Quantity) as TotalSales
FROM Orders GROUP BY CustomerID;

CUSTOMERID | TOTALSALES
-----------|-----------
1001       | 10000
1002       | 20000
1003       | 15000
```

Pivot table:

```
SELECT *
FROM Orders
PIVOT (SUM(UnitPrice*Quantity)
FOR OrderDate
IN ([2022–01–01], [2022–02–01

CUSTOMERID | TOTALSALES
-----------|-----------
1001       | 10000
1002       | 20000
```

DATA TRANSFORMATION

CASE Statement:

```
SELECT id,name,sal,dept,
CASE
WHEN sal>100000 AND dept='sales' THEN 10000
WHEN sal>80000 AND dept='Marketing' THEN 8000
WHEN sal>60000 AND dept='IT' THEN 6000
ELSE 0
END AS bonus
FROM emp;
```

ID Name Sal Dept Bonus

101 Alice 120000 sales 10000

102 Bob 85000 Marketing 8000

103 Carol 70000 IT 6000

104 Dave 50000 Finance 0

COALESCE Function:

```
SELECT id,name,email,phone,
COALESCE(email,phone,'No contact')AS
Contact FROM customers
```

ID	Name	Email	Phone	Contact
201	Alice	alice@mail.com	1234567890	alice@mail.com
202	Bob	NULL	9876543210	9876543210
203	Carol	NULL	NULL	No contact

CONCATENATE (CONCAT) Function:

```
SELECT id,name,price,category,
CONCATENATE(name,'is a',category,'product that cost',price) AS description FROM
products
```

ID	Name	Price	Category	Description
301	Laptop	50000	Electronics	Laptop is a Electronics product that costs 50000
302	Pen	10	Stationery	Pen is a Stationery product that costs 10
303	T-shirt	500	Clothing	T-shirt is a Clothing product that costs 500

CAST and CONVERT Functions:

```
SELECT id,cus_id,pro_id,quantity,order_date,
CAST(order_date AS VARCHAR(4)) AS order_year FROM orders;

SELECT id,cus_id,pro_id,quantity,order_date,
CONVERT( VARCHAR(4), order_date) AS order_year FROM orders;
```

ID	Cus_ID	Pro_ID	Quantity	Order_Date	Order_Year
401	201	301	2	2024-03-15	2024
402	202	302	5	2023-11-20	2023
403	203	303	1	2022-07-05	2022

DATA ANALYSIS

Retail database with sales data

CREATE TABLE Customers (customer_id INT PRIMARY KEY, name VARCHAR(255) NOT NULL, signup_date DATE);

CREATE TABLE Products (product_id INT PRIMARY KEY, product_name VARCHAR(255) NOT NULL, category VARCHAR(100), price DECIMAL(10, 2) NOT NULL);

CREATE TABLE Sales (sale_id INT PRIMARY KEY, customer_id INT, product_id INT, sale_date DATE, quantity INT NOT NULL, total_amount DECIMAL(10, 2), FOREIGN KEY (customer_id) REFERENCES Customers(customer_id), FOREIGN KEY (product_id) REFERENCES Products(product_id));

1. Customers Table:

customer_id	name	signup_date
1	Alice	2024-01-10
2	Bob	2024-02-15
3	Charlie	2024-03-05

2. Products Table:

product_id	product_name	category	price
101	Laptop	Electronics	1200.00
102	Smartphone	Electronics	800.00
103	Headphones	Accessories	150.00
104	Mouse	Accessories	30.00
105	Monitor	Electronics	350.00

3.Sales Table:

sale_id	customer_id	product_id	sale_date	quantity	total_amount
1	1	101	2024-04-01	1	1200.00
2	2	102	2024-04-05	2	1600.00
3	3	103	2024-04-10	3	450.00
4	1	104	2024-04-12	5	150.00
5	2	105	2024-04-15	1	350.00

Find the total sales revenue.

SELECT SUM(total_amount) AS total_revenue FROM Sales;

total_revenue

4800.00

Find the top 5 products by total revenue.

```
SELECT p.product_name,
    SUM(s.total_amount) AS revenue
FROM Sales s
JOIN Products p ON s.product_id = p.product_id
GROUP BY p.product_name
ORDER BY revenue DESC
LIMIT 5;
```

product_name	revenue
Smartphone	1600.00
Laptop	1200.00
Headphones	450.00
Monitor	350.00
Mouse	150.00

Calculate the average purchase amount per customer.

```
SELECT c.customer_id,
    c.name,
    AVG(s.total_amount) AS avg_purchase_amount
FROM Sales s
JOIN Customers c ON s.customer_id = c.customer_id
GROUP BY c.customer_id, c.name
ORDER BY avg_purchase_amount DESC;
```

customer_id	name	avg_purchase_amount
2	Bob	950.00
1	Alice	675.00
3	Charlie	450.00

Find the category with the highest sales.

```
SELECT p.category,
    SUM(s.quantity) AS total_quantity_sold
FROM Sales s
JOIN Products p ON s.product_id = p.product_id
GROUP BY p.category
ORDER BY total_quantity_sold DESC
LIMIT 1;
```

category	total_quantity_sold
Electronics	4

Data Modeling

E-commerce Database Schema

```
CREATE TABLE Customers (
   customer_id INT PRIMARY KEY, name VARCHAR(255) NOT NULL,
   email VARCHAR(255) UNIQUE NOT NULL,
   signup_date DATE);

CREATE TABLE Products (
   product_id INT PRIMARY KEY, product_name VARCHAR(255) NOT NULL,
   category VARCHAR(100),
   price DECIMAL(10, 2) NOT NULL CHECK (price > 0));

CREATE TABLE Orders (
   order_id INT PRIMARY KEY,    customer_id INT NOT NULL,
   order_date DATE,    FOREIGN KEY (customer_id) REFERENCES
Customers(customer_id));

CREATE TABLE Order_Items (
   order_item_id INT PRIMARY KEY,
   order_id INT NOT NULL, product_id INT NOT NULL,
   quantity INT NOT NULL CHECK (quantity > 0),
   FOREIGN KEY (order_id) REFERENCES Orders(order_id),
   FOREIGN KEY (product_id) REFERENCES Products(product_id));
```

1. Customers Table

customer_id	name	email	signup_date
1	Alice Johnson	alice@example.com	2024-01-15
2	Bob Smith	bob@example.com	2024-02-20
3	Charlie Brown	charlie@example.com	2024-03-10

---

2. Products Table

product_id	product_name	category	price
1	Laptop	Electronics	1200.00
2	Phone	Electronics	800.00
3	Book	Books	20.00

---

3. Orders Table

order_id	customer_id	order_date
1	1	2024-04-01
2	2	2024-04-05
3	3	2024-04-10

---

4. **Order_Items Table**

order_item_id	order_id	product_id	quantity
1	1	1	1
2	1	3	2
3	2	2	1
4	3	3	5