```sql
                                        /*       DATABASE CREATION      */
CREATE DATABASE PROJECT2;

USE PROJECT2;

 --DROP DATABASE PROJECT2
                                        /*      TABLE CREATION      */

CREATE TABLE CATEGORIES
(
CID INT PRIMARY KEY,
CNAME VARCHAR(33)
);

CREATE TABLE PRODUCTS
(
PID INT,
PNAME VARCHAR(55),
PRICE INTEGER,
CID INT FOREIGN KEY REFERENCES CATEGORIES(CID)
);
-- DROP TABLE PRODUCTS

                                                    /*      DATA INSERTION      */
INSERT INTO CATEGORIES VALUES(1,'FRUITS');
INSERT INTO CATEGORIES VALUES(2,'VEGITABLES');
INSERT INTO CATEGORIES VALUES(3,'SHAMPOOS');
INSERT INTO CATEGORIES VALUES(4,'CONDITIONORS');
INSERT INTO CATEGORIES VALUES(5,'GEL');


INSERT INTO PRODUCTS VALUES(111,'grapes',23,1)
INSERT INTO PRODUCTS VALUES(112,'onion',44,2)
INSERT INTO PRODUCTS VALUES(113,'all clear',55,4)
INSERT INTO PRODUCTS VALUES(114,'livion',66,5)
INSERT INTO PRODUCTS VALUES(115,'mango',23,1)
INSERT INTO PRODUCTS VALUES(116,'SUNSILK',87,3)
INSERT INTO PRODUCTS VALUES(117,'LOREAL',55,5)
INSERT INTO PRODUCTS VALUES(118,'GILLETE',66,5)
INSERT INTO PRODUCTS VALUES(119,'APPLE',76,1)
INSERT INTO PRODUCTS VALUES(120,'BLUNT',54,3)
INSERT INTO PRODUCTS VALUES(121,'BRINJAL',99,2)
INSERT INTO PRODUCTS VALUES(122,'BANANA',43,1)


SELECT * FROM CATEGORIES
SELECT * FROM PRODUCTS
                                                    /*      QUERIES     */

/* 1 LIST ALL THE PRODUCTS THAT BELONGS TO VEGITABLES AND CONDITIONERS*/
SELECT PNAME,CNAME FROM PRODUCTS, CATEGORIES
WHERE CNAME IN ('VEGITABLES', 'CONDITIONORS') AND PRODUCTS.CID =
    CATEGORIES.CID

/* 1. ANOTHER METHOD*/
SELECT PNAME FROM PRODUCTS
WHERE CID IN (SELECT CID FROM CATEGORIES
WHERE CNAME IN ('VEGITABLES', 'CONDITIONORS'))

/* 2 LIST ALL THE PRODUCTS THAT HAD PRICE RANGE 30 - 65*/
SELECT PNAME FROM PRODUCTS
WHERE PRICE>=30 AND PRICE<65

/* 2 ANOTHER METHOD */
SELECT PNAME FROM PRODUCTS
WHERE PRICE BETWEEN 30 AND 65

-- FOR UNDERSTANDING

SELECT PNAME, PRICE FROM PRODUCTS
WHERE PRICE BETWEEN 30 AND 65


/*3 SUM OF TOTAL PRICES FOR EACH CATEGORY*/

SELECT CNAME,SUM(PRICE) AS SUM_PRICES
FROM PRODUCTS AS P, CATEGORIES AS C
WHERE C.CID=P.CID GROUP BY CNAME

/* 3. ANOTHER METHOD*/
SELECT SUM(P.PRICE) AS CATEGORY_PRICE,C.CNAME FROM
CATEGORIES AS C JOIN PRODUCTS AS P ON C.CID = P.CID
GROUP BY C.CNAME

/* 4. PRODUCTS WITH HIGHEST PRICES (5 PRICES) */

SELECT TOP 5 PRICE FROM PRODUCTS ORDER BY PRICE DESC

SELECT PNAME FROM PRODUCTS ORDER BY PRICE DESC
/*4 ANOTHER METHOD*/

SELECT TOP 5 PNAME, PRICE FROM PRODUCTS
ORDER BY PRICE DESC

/* 5 ARRANGE PRODUCTS DESCENDING ORDER OF PRICE */

SELECT * FROM PRODUCTS ORDER BY PRICE DESC



                                                           /*
                    CONCEPTS        */
                                                           /*       SELECT
                        */

select * from CATEGORIES
```

```sql

select * from  PRODUCTS

                                                            /*      DISTINCT                                  */

select distinct cid from products

select distinct cid,pname from products

                                                            /*      WHERE                                 */

select pname, price from products where cid = 1

select pname, price from products where price > 50

                                                            /*      BETWEEN                                   */

select pname, price from PRODUCTS where price between 40 and 80

                                                            /*      AND OR NOT                                */

select pname,price from products where cid = 1 and pname = 'grapes'

select pname, cid from products where cid = 1 or cid = 3

select pname, cid from products where not cid = 1

select pname, price from products where cid = 2 and (pname  = 'onion' or pname = 'BRINJAL')

                                                                /*      ORDER BY      */

/* ascending order by default */

select pname,price from products order by price

/* descending order */

select pname, price from products order by price desc

select pname, cid from products order by cid

select pname, cid from products order by cid,pname
-- previously pname is comming by insertion order
-- now by including pname for order by pname getting sorted

select pname from PRODUCTS order by pname

select pname, cid from products order by cid ,pname desc
-- one column ascending while another descending order

select pname, cid from products order by cid asc,pname desc

select pname, cid from products order by cid desc,pname asc

                                                                        /*                    NULLS       */

INSERT INTO PRODUCTS (PID,PNAME) VALUES (123,'LIMKA')
-- FOR CID AND PRICE NULL

SELECT * FROM PRODUCTS

SELECT * FROM PRODUCTS WHERE CID IS NOT NULL
-- RESULTS WHICH ARE NOT HAVING NULL

SELECT * FROM PRODUCTS WHERE CID IS  NULL
-- RESULTS WHICH ARE HAVING NULL

                                                                        /*                    UPDATE          */
SELECT * FROM PRODUCTS WHERE CID = 2

UPDATE PRODUCTS
SET CID = 2
WHERE PID=123

SELECT * FROM PRODUCTS WHERE CID = 2

SELECT * FROM PRODUCTS

--UPDATE PRODUCTS
--SET CID = 2

SELECT * FROM PRODUCTS

/* If you omit the WHERE clause,
all records in the table will be updated! */

ALTER TABLE PRODUCTS
ADD VALIDITY INT

SELECT * FROM PRODUCTS

UPDATE PRODUCTS
SET VALIDITY = 10
WHERE PID<115
-- ONLY SPECIFIED ROWS ARE UPDATED

UPDATE PRODUCTS
SET VALIDITY = 20
```

```sql
-- ALL ROWS ARE UPDATED BECAUSE IT IS NOT MENTIONED WITH WEHRE CLAUSE

SELECT * FROM PRODUCTS

                                                                                           /*
                                DELETE      */

DELETE FROM PRODUCTS
WHERE PID = 123

SELECT * FROM PRODUCTS

/* If you omit the WHERE clause,
all records in the table will be deleted!    */

--TO DELETE TABLE
/*   DELETE FROM TABLE_NAME */

                                                                                           /*
                                SELECT TOP CLAUSE       */

SELECT * FROM CATEGORIES
SELECT * FROM PRODUCTS

SELECT TOP 5 * FROM PRODUCTS

SELECT TOP 5 PRICE, PNAME FROM PRODUCTS

--FOR PRODUCT PRICES IN DESCEDING ORDER(HIGH - LOW)
SELECT TOP 5 PRICE,PNAME FROM PRODUCTS ORDER BY PRICE DESC

--FOR PRODUCT PRICES IN ASCENDING ORDER(LOW - HIGH)
SELECT TOP 5 PRICE, PNAME FROM PRODUCTS ORDER BY PRICE

-- GROUP BY JUST PRACTICE
 SELECT SUM(PRICE) FROM PRODUCTS GROUP BY CID


                 /*      MIN         */

 --MIN PRICE VALUES IN PRODUCTS
 SELECT MIN(PRICE) FROM PRODUCTS
 -- MIN PRICE IN EACH CATEGORY
 SELECT MIN(PRICE) AS MIN_PRICE,CID FROM PRODUCTS GROUP BY CID

 -- PRODUCT NAME OF MIN PRICE
 SELECT PNAME,PRICE FROM PRODUCTS WHERE PRICE
    IN
        (SELECT MIN(PRICE) AS PRICE FROM PRODUCTS GROUP BY CID)

-- TOTAL PRICE OF EACH CATEGORY
SELECT SUM(PRICE) AS TOTAL_PRICE, CID FROM PRODUCTS GROUP BY CID

```

```
                /*      MAX     */
-- MAX PRICE AMONG ALL PRODUCTS
SELECT MAX(PRICE) FROM PRODUCTS

-- MAX PRICE IN CATEGORY 1
SELECT MAX(PRICE) FROM PRODUCTS WHERE CID =1

-- MAX PRICE IN EACH CATEGORY
SELECT MAX(PRICE),CID FROM PRODUCTS GROUP BY CID

                /*          COUNT       */

--TOTAL NO.OF COLUMNS
SELECT COUNT(CID) FROM PRODUCTS

-- NO OF PRODUCTS FOR EACH CATEGORY
SELECT COUNT(*) AS COUNTING,CID FROM PRODUCTS GROUP BY CID

-- NO OF PRODUCTS WHOOSE PRICE IS ABOVE 100
SELECT PNAME,PRICE FROM PRODUCTS WHERE PRICE >= 75

SELECT COUNT(*)FROM PRODUCTS WHERE PRICE >= 75

                    /*          AVG     */
--AVERAGE PRICES IN PRODUCTS
SELECT AVG(PRICE) FROM PRODUCTS

--AVERAGE PRICES IN PRODUCTS BASED ON CIDS
SELECT AVG(PRICE), CID FROM PRODUCTS GROUP BY CID

                 /*          SUM     */
-- SUM OF ALL PRODUCTS PRICES
SELECT SUM(PRICE) FROM PRODUCTS

-- SUM OF ALL THE PRICES WITH RESPECT TO CID
SELECT SUM(PRICE), CID FROM PRODUCTS GROUP BY CID

                /*          LIKE        */
SELECT * FROM PRODUCTS

-- STARTS WITH A
SELECT * FROM PRODUCTS WHERE PNAME LIKE 'A%'

-- ENDS WITH A
SELECT * FROM PRODUCTS WHERE PNAME LIKE '%A'

--ANY IN MIDDLE
SELECT * FROM PRODUCTS WHERE PNAME LIKE '%A%'
```

```
 SELECT * FROM PRODUCTS WHERE PNAME LIKE '%N%'

 -- 2ND POSITION A
 SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '_A%'

 -- 7TH POSITION A
 SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '_______A%'

--STARTS WITH A AND ATLEAST 3 CHARACTER LENGTH
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE 'A__%'

--STARTS WITH A AND ENDS WITH R
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE 'A%R'

            /*      NOT LIKE        */
-- SHOULD NOT STARTS WITH A
SELECT PNAME FROM PRODUCTS WHERE PNAME NOT LIKE 'A%' --ALL CLEAR AND APLLE LEFT

-- A SHOULD NOT BE IN MIDDLE
SELECT PNAME FROM PRODUCTS WHERE PNAME NOT LIKE '%A%'

            /*      wildcards       */
/* A wildcard character is used to substitute
one or more characters in a string*/
-->     %  _  []  ^  -
--  %
-- PNAME STARTS WITH A
SELECT * FROM PRODUCTS WHERE PNAME LIKE 'A%'
-- ALL PNAME CONTAIN AN
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '%AN%'
-- AFTER 4---- ETE STRING
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '____ETE'
-- STARTS WITH ANY LETTER SECOND LETTER IS A
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '_A%'

SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '_A_A_A'

--starting with "b", "s", or "p"
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '[BSP]%'
--ENDS with "b", "s", or "p"
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '%[BSP]'

--starting with "a", "b", or "c":
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '[A-C]%'

-- NOT starting with "b", "s", or "p":
SELECT PNAME FROM PRODUCTS WHERE PNAME LIKE '[^AB]%'
--OR
SELECT PNAME FROM PRODUCTS WHERE PNAME NOT LIKE '[AB]%'

```

```sql
                /*      IN      */
-- MIN PRICE PRODUCTS
SELECT PRICE, PNAME FROM PRODUCTS WHERE PRICE IN(
 SELECT MIN(PRICE) FROM PRODUCTS)

SELECT * FROM PRODUCTS WHERE CID IN (1,2,3)

SELECT * FROM PRODUCTS WHERE CID NOT IN (1,2,3)

            /*      BETWEEN AND NOT BETWEEN     */

SELECT * FROM PRODUCTS WHERE CID BETWEEN 2 AND 4

SELECT * FROM PRODUCTS WHERE CID NOT BETWEEN 2 AND 4


SELECT * FROM PRODUCTS WHERE PRICE BETWEEN 51 AND 99 AND CID NOT IN(2,4)
-- 99 NOT INCLUDED

SELECT * FROM PRODUCTS
WHERE PNAME BETWEEN 'A' AND 'G' -- G NOT INCLUDED

SELECT * FROM PRODUCTS
WHERE PNAME BETWEEN 'A' AND 'H'


SELECT * FROM PRODUCTS
WHERE PNAME BETWEEN 'A' AND 'H' ORDER BY PNAME

--CAN ALSO USED FOR DATES


                /*          ALIAS       */

SELECT CID AS CUSTOMERID FROM PRODUCTS

SELECT DISTINCT CID AS UNIQUECUSTOMER FROM PRODUCTS

SELECT MAX(PRICE) AS MAXPRICE FROM PRODUCTS

SELECT P.PID,P.PNAME FROM PRODUCTS AS P


                /*              JOIN        */
/* A JOIN clause is used to combine rows from two or more tables,
based on a related column between them.*/

SELECT * FROM PRODUCTS
SELECT * FROM CATEGORIES

SELECT * FROM PRODUCTS AS P, CATEGORIES AS C WHERE P.CID = C.CID
```

```sql
--ABOVE BELOW SAME
SELECT * FROM PRODUCTS JOIN CATEGORIES ON PRODUCTS.CID = CATEGORIES.CID

/*  (INNER) JOIN: Returns records that have matching values in both tables
    LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
    RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
    FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table
*/
-----> INNER JOIN
SELECT * FROM CATEGORIES AS C INNER JOIN PRODUCTS AS P ON C.CID = P.CID
-----> LEFT JOIN
SELECT * FROM CATEGORIES AS C LEFT JOIN PRODUCTS AS P ON C.CID = P.CID
-----> RIGHT JOIN
SELECT * FROM CATEGORIES AS C RIGHT JOIN PRODUCTS AS P ON C.CID = P.CID
-----> FULL OUTER JOIN
SELECT * FROM CATEGORIES AS C FULL OUTER JOIN PRODUCTS AS P ON C.CID = P.CID

SELECT * FROM CATEGORIES
SELECT * FROM PRODUCTS
                                                                                                     /*      UNION       */


SELECT CID FROM PRODUCTS
UNION
SELECT CID FROM CATEGORIES
--UNION selects only distinct values.

--UNION ALL
SELECT CID FROM PRODUCTS
UNION ALL
SELECT CID FROM CATEGORIES

-- UNION WHERE
SELECT CID FROM PRODUCTS WHERE CID>3
UNION
SELECT CID FROM CATEGORIES WHERE CID>3

-- UNION ALL WHERE
SELECT CID FROM PRODUCTS WHERE CID>3
UNION ALL
SELECT CID FROM CATEGORIES WHERE CID>3

SELECT * FROM PRODUCTS WHERE CID IN(
SELECT CID FROM PRODUCTS WHERE CID>3
UNION ALL
SELECT CID FROM CATEGORIES WHERE CID>3)


```

```sql
                     /*      GROUP BY        */
SELECT * FROM CATEGORIES
SELECT * FROM PRODUCTS

SELECT CID FROM PRODUCTS
GROUP BY (CID)
ORDER BY(CID)

SELECT COUNT(CID),CID FROM PRODUCTS
GROUP BY CID

SELECT COUNT(CID) COUNT_CID,CID FROM PRODUCTS
GROUP BY CID
ORDER BY COUNT_CID DESC

SELECT SUM(PRICE) FROM PRODUCTS WHERE CID IN(
SELECT CID FROM PRODUCTS
GROUP BY(CID))

                                                                                             /*      GROUP BY HAVING     */
-- USING SPECIFIC CONDITION CID > 3
SELECT COUNT(CID) AS COUNTING,CID FROM PRODUCTS
GROUP BY CID
HAVING CID>3

-- CONDITION OF CID BETWEEN 1 AND 4
SELECT SUM(PRICE) SUM ,CID FROM PRODUCTS
GROUP BY CID
HAVING CID BETWEEN 1 AND 4

                                                                                             /*      EXISTS          */
SELECT PNAME, PRICE FROM PRODUCTS WHERE EXISTS(
SELECT CID FROM PRODUCTS)

-- DIFFERENCE BETWEEN IN AND EXISTS

-- EXISTS
/* -> The EXISTS operator is used to check the existence of rows returned by a subquery.
   -> It evaluates the subquery and returns true if the subquery returns at least one row;
   -> otherwise, it returns false.
*/

-- IN
/* ->The IN operator is used to compare a value against a list of values or a subquery.
   -> It checks if a value matches any value in the specified list or the result set of a subquery.
*/
SELECT CID FROM PRODUCTS WHERE CID > 3

SELECT PNAME, PRICE,CID FROM PRODUCTS WHERE EXISTS(
SELECT CID FROM PRODUCTS WHERE CID > 3)

SELECT PNAME, PRICE,CID FROM PRODUCTS WHERE CID IN (
SELECT CID FROM PRODUCTS WHERE CID > 3)

SELECT CID FROM PRODUCTS WHERE CID BETWEEN 1 AND 4

SELECT PNAME,PRICE,CID FROM PRODUCTS WHERE EXISTS(
SELECT CID FROM PRODUCTS WHERE CID BETWEEN 1 AND 4
)

SELECT PNAME,PRICE,CID FROM PRODUCTS WHERE CID IN(
SELECT CID FROM PRODUCTS WHERE CID BETWEEN 1 AND 4
)

----> EXISTS ONLY ACTS AS A CONDITIONER
----> IF TRUE OUTER QUERY WILL BE EXICUTED
----> IF NOT IT WON'T EXICUTED



        /*    ANY ALL       */

SELECT CID FROM PRODUCTS
WHERE CID > 4

SELECT * FROM PRODUCTS
WHERE CID = ANY(
SELECT CID FROM PRODUCTS
WHERE CID > 4)


SELECT CID FROM PRODUCTS
WHERE CID > 5

SELECT * FROM PRODUCTS
WHERE CID = ANY(
SELECT CID FROM PRODUCTS
WHERE CID > 5)

SELECT CID FROM PRODUCTS
WHERE CID BETWEEN 1 AND 6

SELECT * FROM PRODUCTS
WHERE CID = ALL(
SELECT CID FROM PRODUCTS
WHERE CID = 1)

SELECT * FROM PRODUCTS
WHERE CID = ALL(
SELECT CID FROM PRODUCTS
WHERE CID BETWEEN 1 AND 6)



                                                                                                    /*      SELECT INTO         */


/*The SELECT INTO statement copies data from one table into a new table.*/

--SYNTAX

/*
        SELECT column1, column2, column3, ...
        INTO newtable [IN externaldb]
        FROM oldtable
        WHERE condition;
*/
SELECT * INTO PRODUCTS2
FROM PRODUCTS

SELECT * FROM PRODUCTS2

SELECT PNAME, PRICE INTO PRODUCTS3 FROM PRODUCTS

SELECT * FROM PRODUCTS3

                                                                                                /*      INSERT INOT SELECT      */


/*The INSERT INTO SELECT statement copies data from one table and inserts it into another table.

The INSERT INTO SELECT statement requires that the data types in source and target tables match.

Note: The existing records in the target table are unaffected.
TABLE NEED TO BE EXISTS PREVIOUSLY

*/
CREATE TABLE PRO (PNAME VARCHAR(25), PRICE INT)
INSERT INTO PRO SELECT * FROM PRODUCTS3

SELECT * FROM PRO

/*
The CASE expression goes through conditions and returns a value when the first condition is met (like an if-then-else statement).

So, once a condition is true, it will stop reading and return the result.

If no conditions are true, it returns the value in the ELSE clause.

If there is no ELSE part and no conditions are true, it returns NULL.

CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;

*/

SELECT PNAME, CASE
    WHEN PRICE >=20 AND PRICE <50 THEN 'CHEAP'
    WHEN PRICE >=50 AND PRICE <75 THEN 'VALUE FOR MONEY'
    WHEN PRICE >=75 AND PRICE <100 THEN 'EXPENSIVE'
ELSE 'UN LABELED'
END AS REVIEW
FROM PRO;
```