UNIT-3 DATABASE AND SQL

Data :- Raw facts and figures which are useful to an organization. We cannot take decisions on the basis of data.

Information:- Well processed data is called information. We can take decisions on the basis of information

Field: Set of characters that represents specific data element.

Record: Collection of fields is called a record. A record can have fields of different data types.

File: Collection of similar types of records is called a file.

Table: Collection of rows and columns that contains useful data/information is called a table. A table generally refers to the passive entity which is kept in secondary storage device.

Relation: Relation (collection of rows and columns) generally refers to an active entity on which we can perform various operations.

Database: Collection of logically related data along with its description is termed as database.

Tuple: A row in a relation is called a tuple.

Attribute: A column in a relation is called an attribute. It is also termed as field or data item.

Degree: Number of attributes in a relation is called degree of a relation.

Cardinality: Number of tuples in a relation is called cardinality of a relation.

Primary Key: Primary key is a key that can uniquely identifies the records/tuples in a relation. This key can never be duplicated and NULL.

Foreign Key: Foreign Key is a key that is defined as a primary key in some other relation. This key is used to enforce referential integrity in RDBMS.

Candidate Key: Set of all attributes which can serve as a primary key in a relation.

Alternate Key: All the candidate keys other than the primary keys of a relation are alternate keys for a relation.

DBA: Data Base Administrator is a person (manager) that is responsible for defining the data base schema, setting security features in database, ensuring proper functioning of the data bases etc.

Relational Algebra: Relation algebra is a set operation such as select, project, union, & Cartesian product etc.

Select operation : Yields set of set of rows depending upon certain condition. Mathematically it is denoted as (σ)

e.g $\sigma_{rollno>10}$(student) -means show those rows of student table whose roll no.'s are >10.

Project Operation **:** yields set of columns as result which are specified. Mathematically it is denoted as π.

e.g. $\pi_{rollno,name}$(student) - means show only rollno and name column only.

Union Operation : Two relation are said to be union compatible if their degree and column are same.

e.g Relation A

	X	Y	Z
Relation A	x1	y1	z1
	x2	y2	z2

	X	Y	Z
Relation B	x1	y1	z1
	x3	y3	z3

Resultant Relation A∪B=

X	Y	Z
x1	y1	z1
x2	y2	z2
x3	y3	z3

Cartesian product: Cartesian Product of two relation A and B gives resultant relation whose no. of column are sum of degrees of two relation and no. of rows are product of cardinality of two relations.

e.g Relation A

Relation A

X	Y	Z
x1	y1	z1
x2	y2	z2

Relation B

U	V
u1	v1
u2	v2

Resultant Relation A×B =

X	Y	Z	U	V
x1	y1	z1	u1	v1
x1	y1	z1	u2	v2
x2	y2	z2	u1	v1
x2	y2	z2	u2	v2

Structured Query Language

SQL is a non procedural language that is used to create, manipulate and process the databases(relations).

Characteristics of SQL

1. It is very easy to learn and use.
2. Large volume of databases can be handled quite easily.
3. It is non procedural language. It means that we do not need to specify the procedures to accomplish a task but just to give a command to perform the activity.
4. SQL can be linked to most of other high level languages that makes it first choice for the database programmers.

Processing Capabilities of SQL

The following are the processing capabilities of SQL

1. **Data Definition Language (DDL)**

   DDL contains commands that are used to create the tables, databases, indexes, views, sequences and synonyms etc.

   e.g: Create table, create view, create index, alter table etc.

2. **Data Manipulation Language (DML)**

   DML contains command that can be used to manipulate the data base objects and to query the databases for information retrieval.

   e.g Select, Insert, Delete, Update etc.

3. **Data Control Language:**

   This language is used for controlling the access to the data. Various commands like GRANT, REVOKE etc are available in DCL.

4. **Transaction Control Language (TCL)**

   TCL include commands to control the transactions in a data base system. The commonly used commands in TCL are COMMIT, ROLLBACK etc.

Data types of SQL

Just like any other programming language, the facility of defining data of various types is available in SQL also. Following are the most common data types of SQL.

1) NUMBER
2) CHAR
3) VARCHAR / VARCHAR2
4) DATE
5) LONG
6) RAW/LONG RAW

1. NUMBER

Used to store a numeric value in a field/column. It may be decimal, integer or a real value. General syntax is

Number(n,d)

Where **n** specifies the number of digits and **d** specifies the number of digits to the right of the decimal point.

e.g marks number(3) declares marks to be of type number with maximum value 999.
pct number(5,2) declares pct to be of type number of 5 digits with two digits to the right of decimal point.

2. CHAR

Used to store character type data in a column. General syntax is

Char (size)

where size represents the maximum number of characters in a column. The CHAR type data can hold at most 255 characters.

e.g name char(25) declares a data item name of type character of upto 25 size long.

3. VARCHAR/VARCHAR2

This data type is used to store variable length alphanumeric data. General syntax is

varchar(size) / varchar2(size)

where size represents the maximum number of characters in a column. The maximum allowed size in this data type is 2000 characters.

e.g address varchar(50); address is of type varchar of upto 50 characters long.

4. DATE

Date data type is used to store dates in columns. SQL supports the various date formats other that the standard DD-MON-YY.

e.g dob date; declares dob to be of type date.

5. LONG

This data type is used to store variable length strings of upto 2 GB size.

e.g description long;

6. RAW/LONG RAW

To store binary data (images/pictures/animation/clips etc.) RAW or LONG RAW data type is used. A column LONG RAW type can hold upto 2 GB of binary data.

e.g image raw(2000);

SQL Commands

CREATE TABLE Command:

Create table command is used to create a table in SQL. It is a DDL type of command.

Syntax :

CREATE TABLE <table name>
(<column name> <data types>[(size)] [,<column name> <data types>[(size)]….);
e.g.
Create table student
(rollno number(2), name
varchar2(20), dob date);

Constraints:

Constraints are the conditions that can be enforced on the attributes of a relation. The constraints come in play when ever we try to insert, delete or update a record in a relation.They are used to ensure integrity of a relation, hence named as **integrity constraints**.

1. NOT NULL
2. UNIQUE

3. PRIMARY KEY
4. FOREIGN KEY
5. CHECK
6. DEFAULT

i. **Not Null constraint** : It ensures that the column cannot contain a NULL value.

ii. **Unique constraint** : A candidate key is a combination of one or more columns, the value of which uniquely identifies each row of a table.

iii. **Primary Key** : It ensures two things : (i) Unique identification of each row in the table. (ii) No column that is part of the Primary Key constraint can contain a NULL value.

iv. **Foreign Key** : The foreign key designates a column or combination of columns as a foreign key and establishes its relationship with a primary key in different table.

```
Create table Fee
(RollNo number(2) Foreign key (Rollno) references Student (Rollno),
Name varchar2(20) Not null, Amount
number(4), Fee_Date date);
```

v. **Check Constraint :** Sometimes we may require that values in some of the columns of our table are to be within a certain range or they must satisfy certain conditions.

Example:

```
Create table Employee
(EmpNo number(4) Primary Key,
Name varchar2(20) Not Null,
Salary number(6,2) check (salary > 0),
DeptNo number(3)
);
```

Data Manipulation in SQL

DML Commands are as under:

SELECT - Used for making queries
INSERT - Used for adding new row or record into table
UPDATE- used for modification in existing data in a table
DELETE – used for deletion of records.

INSERT Statement

To insert a new tuple into a table is to use the insert statement
insert into <table> [(<column i, . . . , column j>)] values (<value i, . . . , value j>);
INSERT INTO student VALUES(101,'Rohan','XI',400,'Jammu');
While inserting the record it should be checked that the values passed are of same data types as the one which is specified for that particular column.
For inserting a row interactively (from keyboard) & operator can be used.
e.g INSERT INTO student VALUES(&Roll_no','&Name','&Class','&Marks','&City');
In the above command the values for all the columns are read from keyboard and inserted into the table student.

NOTE:- In SQL we can repeat or re-execute the last command typed at SQL prompt by typing "/" key and pressing enter.

Roll_no	Name	Class	Marks	City
101	Rohan	XI	400	Jammu
102	Aneeta Chopra	XII	390	Udhampur
103	Pawan Kumar	IX	298	Amritsar
104	Rohan	IX	376	Jammu
105	Sanjay	VII	240	Gurdaspur
113	Anju Mahajan	VIII	432	Pathankot

Operators in SQL:

The following are the commonly used operators in SQL

1. Arithmetic Operators +, -, *, /
2. Relational Operators =, <, >, <=, >=, <>
3. Logical Operators OR, AND, NOT

Arithmetic operators are used to perform simple arithmetic operations.
Relational Operators are used when two values are to be compared and
Logical operators are used to connect search conditions in the WHERE Clause in SQL.
Other operators :

4. Range check – between low and high
5. List check – in
6. Pattern check – like , not like (% and _ 'under score' is used)

Queries:

To retrieve information from a database we can query the databases. SQL SELECT statement is used to select rows and columns from a database/relation.

SELECT Command

This command can perform **selection** as well as **projection.**

Selection: This capability of SQL can return you the tuples form a relation with all the attributes.
e.g. SELECT name, class FROM student;

The above command displays only name and class attributes from student table.

Projection: This is the capability of SQL to return only specific attributes in the relation. Use of **where clause** is required when specific tuples are to be fetched or manipulated.

e.g SELECT * FROM student; command will display all the tuples in the relation student SELECT * FROM student WHERE Roll_no <=102;

The above command display only those records whose Roll_no less than or equal to 102.

Select command can also display specific attributes from a relation.

* SELECT count(*) AS "Total Number of Records" FROM student;
Display the total number of records with title as "Total Number of Records" i.e an alias

We can also use arithmetic operators in select statement, like

* SELECT Roll_no, name, marks+20 FROM student;
* SELECT name, (marks/500)*100 FROM student WHERE Roll_no > 103;

Eliminating Duplicate/Redundant data

DISTINCT keyword is used to restrict the duplicate rows from the results of a SELECT statement.
e.g. SELECT DISTINCT name FROM student;

The above command returns

Name
Rohan
Aneeta Chopra
Pawan Kumar

Conditions based on a range

SQL provides a BETWEEN operator that defines a range of values that the column value must fall for the condition to become true.

e.g. SELECT Roll_no, name FROM student WHERE Roll_no BETWENN 100 AND 103;

The above command displays Roll_no and name of those students whose Roll_no lies in the range 100 to 103 (both 100 and 103 are included in the range).

Conditions based on a list

To specify a list of values, IN operator is used. This operator select values that match any value in the given list.

e.g. SELECT * FROM student WHERE city IN ('Jammu','Amritsar','Gurdaspur');

The above command displays all those records whose city is either Jammu or Amritsar or Gurdaspur.

Conditions based on Pattern

SQL provides two wild card characters that are used while comparing the strings with LIKE operator.

a. percent(%) Matches any string
b. Underscore(_) Matches any one character

e.g SELECT Roll_no, name, city FROM student WHERE Roll_no LIKE "%3";

displays those records where last digit of Roll_no is 3 and may have any number of characters in front.

e.g SELECT Roll_no, name, city FROM student WHERE Roll_no LIKE "1_3";

displays those records whose Roll_no starts with 1 and second letter may be any letter but ends with digit 3.

ORDER BY Clause

ORDER BY clause is used to display the result of a query in a specific order(sorted order).

The sorting can be done in ascending or in descending order. It should be kept in mind that the actual data in the database is not sorted but only the results of the query are displayed in sorted order.

e.g. SELECT name, city FROM student ORDER BY name;

The above query returns name and city columns of table student sorted by name in increasing/ascending order.

e.g. SELECT * FROM student ORDER BY city DESC;

It displays all the records of table student ordered by city in descending order.

Note:- If order is not specifies that by default the sorting will be performed in ascending order.

SQL Functions :

SQL supports functions which can be used to compute and select numeric, character and date columns of a relations. These functions can be applied on a group of rows. The rows are grouped on a common value of a column in the table. These functions return only one value for a group and therefore, they are called aggregate or group functions.

1. **SUM() :**
   It returns the sum of values of numeric type of a column.
   Eg. Select sum(salary) from employee;
2. **AVG() :**
   It returns the average of values of numeric type of a column.

Eg. Select avg(salary) from employee;

3. **MIN() :**
   It returns the minimum of the values of a column of a given relation.
   Eg. Select min(salary) from employee;
4. **MAX() :**
   It returns the maximum of the values of a column of a given relation.
   Eg. Select max(salary) from employee;
5. **COUNT():**
   It returns the number of rows in a relation.
   Eg. Select count(*) from employee;

Group By Clause :
The rows of a table can be grouped together based on a common value by using the Group By clause of SQL in a select statement.
Syntax :
SELECT <attribute name>, <attribute name> ---- [functions]
FROM <relation name>
GROUP BY <group by column>;
Eg. Select age, count (rollno)
From students Group by age;
Output :

Age	Count(rollno)
15	2
14.5	2
14	5

Group-By-Having Clause :
It is used to apply some condition to the Group By clause.
Eg.
Select class
From students Group by class
Having count(*) > 5;

DELETE Command
To delete the record fro a table SQL provides a delete statement. General syntax is:-

DELETE FROM <table_name> [WHERE <condition>];
e.g. DELETE FROM student WHERE city = 'Jammu';
This command deletes all those records whose city is Jammu.
NOTE: It should be kept in mind that while comparing with the string type values lowercase and uppercase letters are treated as different. That is 'Jammu' and 'jammu' is different while comparing.

UPDATE Command
To update the data stored in the data base, UPDATE command is used.

e. g. UPDATE student SET marks = marks + 100;
Increase marks of all the students by 100.

e. g. UPDATE student SET City = 'Udhampur' WHERE city = 'Jammu';
changes the city of those students to Udhampur whose city is Jammu.

We can also update multiple columns with update command, like
e. g. UPDATE student set marks = marks + 20, city = 'Jalandhar'
WHERE city NOT IN ('Jammu','Udhampur');

CREATE VIEW Command

In SQL we can create a view of the already existing table that contains specific attributes of the table.
e. g. the table student that we created contains following fields:
Student (Roll_no, Name, Marks, Class, City)
Suppose we need to create a view **v_student** that contains Roll_no,name and class of student table, then Create View command can be used:
CREATE VIEW v_student AS SELECT Roll_no, Name, Class FROM student;
The above command create a virtual table (view) named v_student that has three attributes as mentioned and all the rows under those attributes as in student table.

We can also create a view from an existing table based on some specific conditions, like
CREATE VIEW v_student AS SELECT Roll_no, Name, Class FROM student WHERE City <>'Jammu';

The main difference between a Table and view is that

A **Table** is a repository of data. The table resides physically in the database.
A **View** is not a part of the database's physical representation. It is created on a table or another view. It is precompiled, so that data retrieval behaves faster, and also provides a secure accessibility mechanism.

ALTER TABLE Command

In SQL if we ever need to change the structure of the database then ALTER TABLE command is used. By using this command we can add a column in the existing table, delete a column from a table or modify columns in a table.
Adding a column : The syntax to add a column is:-

ALTER TABLE table_name
ADD column_name datatype;
e.g ALTER TABLE student ADD(Address varchar(30));
The above command add a column Address to the table atudent.
If we give command
SELECT * FROM student;
The following data gets displayed on screen:

Roll_no	Name	Class	Marks	City	Address
101	Rohan	XI	400	Jammu	
102	Aneeta Chopra	XII	390	Udhampur	
103	Pawan Kumar	IX	298	Amritsar	
104	Rohan	IX	376	Jammu	
105	Sanjay	VII	240	Gurdaspur	
113	Anju MAhajan	VIII	432	Pathankot	

Note that we have just added a column and there will be no data under this attribute. UPDATE command can be used to supply values / data to this column.

Removing a column

ALTER TABLE table_name
DROP COLUMN column_name;

e.g ALTER TABLE Student
DROP COLUMN Address;
The column Address will be removed from the table student.

DROP TABLE Command
Sometimes you may need to drop a table which is not in use. DROP TABLE command is used to Delete / drop a table permanently. It should be kept in mind that we can not drop a table if it contains records. That is first all the rows of the table have to be deleted and only then the table can be dropped. The general syntax of this command is:-
DROP TABLE <table_name>;
e.g DROP TABLE student;
This command will remove the table student

1&2 mark questions

Q2. Define the terms:
i. Database Abstraction
ii. Data inconsistency
iii. Conceptual level of database implementation/abstraction
iv. Primary Key
v. Candidate Key
vi. Relational Algebra
vii. Domain

Ans:. Define the terms:

i. Database Abstraction
Ans: Database system provides the users only that much information that is required by them, and hides certain details like, how the data is stored and maintained in database at hardware level. This concept/process is Database abstraction.

ii. Data inconsistency
Ans: When two or more entries about the same data do not agree i.e. when one of them stores the updated information and the other does not, it results in data inconsistency in the database.

iii. Conceptual level of database implementation/abstraction
Ans: It describes what data are actually stored in the database. It also describes the relationships existing among data. At this level the database is described logically in terms of simple data-structures.

iv. Primary Key
Ans : It is a key/attribute or a set of attributes that can uniquely identify tuples within the relation.

v. Candidate Key

Ans : All attributes combinations inside a relation that can serve as primary key are candidate key as they are candidates for being as a primary key or a part of it.

vi. Relational Algebra

Ans : It is the collections of rules and operations on relations(tables). The various operations are selection, projection, Cartesian product, union, set difference and intersection, and joining of relations.

vii. Domain

Ans : it is the pool or collection of data from which the actual values appearing in a given column are drawn.

2 marks Practice questions

1. What is relation? What is the difference between a tuple and an attribute?
2. Define the following terminologies used in Relational Algebra:
   (i) selection (ii) projection (iii) union (iv) Cartesian product
3. What are DDL and DML?
4. Differentiate between primary key and candidate key in a relation?
5. What do you understand by the terms **Cardinality** and **Degree** of a relation in relational database?
6. Differentiate between DDL and DML. Mention the 2 commands for each category.

Database and SQL : 6 marks questions

1. Write SQL Command for (a) to (d) and output of (g)

TABLE : GRADUATE

S.NO	NAME	STIPEND	SUBJECT	AVERAGE	DIV
1	KARAN	400	PHYSICS	68	I
2	DIWAKAR	450	COMP. Sc.	68	I
3	DIVYA	300	CHEMISTRY	62	I
4	REKHA	350	PHYSICS	63	I
5	ARJUN	500	MATHS	70	I
6	SABINA	400	CEHMISTRY	55	II
7	JOHN	250	PHYSICS	64	I
8	ROBERT	450	MATHS	68	I
9	RUBINA	500	COMP. Sc.	62	I
10	VIKAS	400	MATHS	57	II

(a) List the names of those students who have obtained DIV I sorted by NAME.

(b) Display a report, listing NAME, STIPEND, SUBJECT and amount of stipend received in a year assuming that the STIPEND is paid every month.

(c) To count the number of students who are either PHYSICS or COMPUTER SC graduates.

(d) To insert a new row in the GRADUATE table: 11,"KAJOL", 300, "computer sc", 75, 1

(e) Give the output of following sql statement based on table GRADUATE:

(i) Select MIN(AVERAGE) from GRADUATE where SUBJECT="PHYSICS";
(ii) Select SUM(STIPEND) from GRADUATE WHERE div=2;
(iii) Select AVG(STIPEND) from GRADUATE where AVERAGE>=65;
(iv) Select COUNT(distinct SUBJECT) from GRADUATE;

Sol :

(b) SELECT NAME from GRADUATE where DIV = 'I' order by NAME;

(c) SELECT NAME,STIPEND,SUBJECT, STIPEND*12 from GRADUATE;

(d) SELECT SUBJECT,COUNT(*) from GRADUATE group by SUBJECT having SUBJECT='PHYISCS' or SUBJECT='COMPUTER SC';

(e) INSERT INTO GRADUATE values(11,'KAJOL',300,'COMPUTER SC',75,1);

(f) i) 63
ii) 800
iii) 475
iv) 4

2. Consider the following tables Sender and Recipient. Write SQL commands for the statements (i) to (iv) and give the outputs for SQL queries (v) to (viii).

Sender

SenderID	SenderName	SenderAddress	City
ND01	R Jain	2, ABC Appts	New Delhi
MU02	H Sinha	12, Newtown	Mumbai
MU15	S Jha	27/A, Park Street	Mumbai
ND50	T Prasad	122 – K, SDA	New Delhi

Recipient

RecID	SenderID	RecName	RecAddress	RecCity
KO05	ND01	R Bajpayee	5, Central Avenue	Kolkata
ND08	MU02	S Mahajan	116, A-Vihar	New Delhi
MU19	ND01	H Singh	2A, Andheri East	Mumbai
MU32	MU15	P K Swamy	B5, C S Terminus	Mumbai
ND48	ND50	S Tripathi	13, B I D, Mayur Vihar	New Delhi

(i) To display the names of all Senders from Mumbai

Ans. SELECT sendername from Sender where sendercity='Mumbai';

(ii) To display the RecIC, Sendername, SenderAddress, RecName, RecAddress for every Recipient.

Ans. Select R.RecIC, S.Sendername, S.SenderAddress, R.RecName, R.RecAddress from Sender S, Recepient R where S.SenderID=R.SenderID;

(iii) To display Recipient details in ascending order of RecName

Ans. SELECT * from Recipent ORDER By RecName;

(iv) To display number of Recipients from each city

Ans. SELECT COUNT(*) from Recipient Group By RecCity;

(v) SELECT DISTINCT SenderCity from Sender; Ans.

SenderCity
Mumbai
New Delhi

(vi) SELECT A.SenderName, B.RecName
From Sender A, Recipient B
Where A.SenderID = B.SenderID AND B.RecCity ='Mumbai';

Ans.

A.SenderName	B.RecName
R Jain	H Singh
S Jha	P K Swamy

(vii) SELECT RecName, RecAddress
From Recipient Where RecCity NOT IN ('Mumbai', 'Kolkata');

Ans. RecName RecAddress
S Mahajan 116, A Vihar
S Tripathi 13, BID, Mayur Vihar

(viii) SELECT RecID, RecName
FROM Recipent Where SenderID='MU02' or SenderID='ND50';

Ans. RecID RecName ND08
S Mahajan ND48 S
Tripathi

3. Write SQL command for (i) to (vii) on the basis of the table SPORTS

Table: SPORTS

Student NO	Class	Name	Game1	Grade	Game2	Grade2
10	7	Sammer	Cricket	B	Swimming	A
11	8	Sujit	Tennis	A	Skating	C
12	7	Kamal	Swimming	B	Football	B
13	7	Venna	Tennis	C	Tennis	A
14	9	Archana	Basketball	A	Cricket	A
15	10	Arpit	Cricket	A	Atheletics	C

(a) Display the names of the students who have grade 'C' in either Game1 or Game2 or both.
(b) Display the number of students getting grade 'A' in Cricket.
(c) Display the names of the students who have same game for both Game1 and Game2.
(d) Display the games taken up by the students, whose name starts with 'A'.
(e) Add a new column named 'Marks'.
(f) Assign a value 200 for Marks for all those who are getting grade 'B' or grade 'A' in both Game1 and Game2.

Ans : a) SELECT Name from SPORTS where grade='C' or Grade2='C';
b) SELECT Count(*) from SPORTS where grade='A';
c) SELECT name from SPORTS where game1 = game2;
d) SELECT game,game2 from SPORTS where name like 'A%';
e) ALTER TABLE SPORTS add (marks int(4));
f) UPDATE SPORTS set marks=200 where grade='A';

4. Consider the following tables Stationary and Consumer. Write SQL commands for the statement (i) to (iv) and output for SQL queries (v) to (viii):

Table: Stationary

S_ID	StationaryName	Company	Price
DP01	Dot Pen	ABC	10
PL02	Pencil	XYZ	6
ER05	Eraser	XYZ	7
PL01	Pencil	CAM	5
GP02	Gel Pen	ABC	15

Table: Consumer

C_ID	ConsumerName	Address	S_ID
01	Good Learner	Delhi	PL01
06	Write Well	Mumbai	GP02
12	Topper	Delhi	DP01
15	Write & Draw	Delhi	PL02
16	Motivation	Banglore	PL01

(i) To display the details of those consumers whose Address is Delhi.
(ii) To display the details of Stationary whose Price is in the range of 8 to 15. (Both Value included)
(iii) To display the ConsumerName, Address from Table Consumer, and Company and Price from table Stationary, with their corresponding matching S_ID.
(iv) To increase the Price of all stationary by 2.
(v) SELECT DISTINCT Address FROM Consumer;
(vi) SELECT Company, MAX(Price), MIN(Price), COUNT(*) from Stationary GROUP BY Company;
(vii) SELECT Consumer.ConsumerName, Stationary.StationaryName, Stationary.Price FROM Strionary, Consumer WHERE Consumer.S_ID=Stationary.S_ID;
(viii) Select StationaryName, Price*3 From Stationary;

5. Consider the following tables GARMENT and FABRIC. Write SQL commands for the statements (i) to (iv) and give outputs for SQL queries (v) to (viii).

Table : GARMENT

GCODE	DESCRIPTION		PRICE	FCODE	READYDATE
10023	PENCIL	SKIRT	1150	F03	19–DEC–08
10001	FORMAL	SHIRT	1250	F01	12–JAN–08
10012	INFORMAL	SHIRT	1550	F02	06–JUN–08
10024	BABY	TOP	750	F03	07–APR–07
10090	TULIP	SKIRT	850	F02	31–MAR–07
10019	EVENING	GOWN	850	F03	06–JUN–08
10009	INFORMAL	PANT	1500	F02	20–OCT–08
10007	FORMAL	PANT	1350	F01	09–MAR–08
10020	FROCK		850	F04	09–SEP–07
10089	SLACKS		750	F03	20–OCT–08

Table : FABRIC

FCODE	TYPE
F04	POLYSTER
F02	COTTON
F03	SILK
F01	TERELENE

(i) To display GCODE and DESCRIPTION of a each dress in descending order of GCODE.
(ii) To display the details of all the GARMENTs, which have READYDATE in between 08–DEC–07 and 16–JUN–08 (inclusive of both the dates).
(iii) To display the average PRICE of all the GARMENTs, which are made up of FABRIC with FCODE as F03.

(iv) To display FABRIC wise highest and lowest price of GARMENTs from DRESS table.
(Display FCODE of each GARMENT along with highest and lowest price)
(v) SELECT SUM (PRICE) FROM GARMENT WHERE FCODE= 'F01';
(vi) SELECT DESCRIPTION, TYPE FROM GARMENT, FABRIC WHERE
GARMENT.FCODE = FABRIC. FCODE AND GARMENT. PRICE>=1260;
(vii) SELECT MAX (FCODE) FROM FABRIC;
(viii) SELECT COUNT (DISTINCT PRICE) FROM FABRIC;

6. Consider the following WORKERS and DESIG. Write SQL commands for the statements (i) to (iv) and give outputs for SQL queries (v) to (vi)

WORKERS

W_ID	FIRSTNAME	LASTNAME	ADDRESS	CITY
102	Sam	Tones	33 Elm St.	Paris
105	Sarah	Ackerman	440 U.S. 110	New York
144	Manila	Sengupta	24 Friends Street	New Delhi
210	George	Smith	83 First Street	Howard
255	Mary	Jones	842 Vine Ave.	Losantiville
300	Robert	Samuel	9 Fifth Cross	Washington
335	Henry	Williams	12Moore Street	Boston
403	Ronny	Lee	121 Harrison St.	New York
451	Pat	Thompson	11 Red Road	Paris

DESIG

W_ID	SALARY	BENEFITS	DESIGNATION
102	75000	15000	Manager
105	85000	25000	Director
144	70000	15000	Manager
210	75000	12500	Manager
255	50000	12000	Clerk
300	45000	10000	Clerk
335	40000	10000	Clerk
403	32000	7500	Salesman
451	28000	7500	Salesman

(i) To display the content of workers table in ascending order of first name.
(ii) To display the FIRSTNAME, CITY and TOTAL SALARY of all Clerks from the tables workers and design, where TOTAL SALARY = SALARY + BENEFITS.
(iii) To display the minimum SALARY among Managers and Clerks from the table DESIG.
(iv) Increase the BENEFITS of all Salesmen by 10% in table DESIG.
(v) SELECT FIRSTNAME, SALARY FROM WORKERS, DESIG WHERE DESIGNATION = 'Manager' AND WORKERS.W_ID = DESIG.W_ID;
(vi) SELECT DESIGNATION, SUM(SALARY) FROM DESIG
GROUP BY DESIGNATION HAVING COUNT(*)>=2 ;