DBMS -(Unit-III)

1 Mark.

1) What is Decomposition?

Ans Decomposition: Decomposition in DBMS is used to break a relation into multiple relations to bring it into an appropriate Normal Form.
It helps to reduce redundancy, anomalies and inconsistencies.
If a relation is not properly decomposed then it may lead to other Problems like information loss.

2) Define union operator with example?

Ans UNION (U): →
Returns a relation instance containing all tuples that occur in either relation instance R or relational instance S or both.

Ex: S1

S_ID	S_name	Rating	age
22	Dustin	7	45
31	Rusty	8	55
58	Puppy	10	35

S2

S_ID	S_Agene	rating	age
28	yuppy	9	35
31	Rusty	8	55
44	Gumpy	5	35
58	puppy	10	35

S1 ∪ S2

S_ID	S_name	Rating	age
22	Dustin	7	45
31	Rusty	8	55
28	yuppy	9	35
44	Gumpy	5	35
58	Puppy	10	35

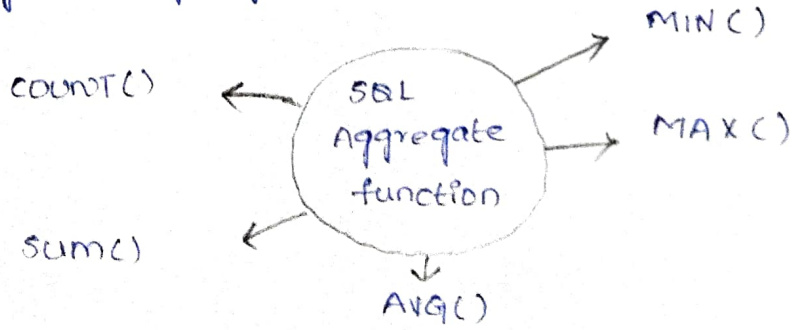
3) Define Normalization?

Ans Normalization: Normalization is a process of minimizing redundancy from a relation or a set of relations.
→ Redundancy in a relation may cause insertion, deletion & update Anomalies so it helps to reduce redundancy in relations.

4) What are the types of Aggregate functions?

Ans Aggregate-functions: An Aggregate-function in SQL perform calculations on multiple values and returns a single value.

Types of Aggregate functions:



5) what is Nested Query?

Ans Nested Query: The nested query is a query that has another query embedded within it.
→ The embedded query is called a subquery.
→ It consists of IN, NOT IN, EXIST, NOT EXIST operators.

3MARKS

1) Describe basic form of a SQL Query with example?

Ans The basic form of a SQL query is as follows:

SELECT (DISTINCT)	Select_list
FROM	from_list
WHERE	qualification

Every query must have a -
SELECT clause which specifies the column to be retained in the result
FROM clause which specifies across product of the table.
WHERE specifies selection condition on the tables mentioned in FROM clause.
DISTINCT - can be used to return only unique values.

Ex: student

S-ID	S-name	S_age
1	Ram	18
2	Ravi	19
3	Raju	17
4	Ram	16

Select DISTINCT S_name from student;

Output: →

S_name
Ram
Ravi
Raju.

2) write about database anomalies?

Ans A database Anomaly is a inconsistency in the data resulting from

a operation like update, delete and insertion.

Redundant storage: some information which is stored repeatedly.

Update Anomalies: when we update some rows in the table, and if it leads to the inconsistency of the table then the anomaly occurs.

Insertion Anomalies: If there is a new row inserted in the table and it creates the inconsistency in the table.

Deletion Anomalies: If we delete some rows from the table and any other information or data which is required is also deleted from the database.

* TO remove this type of anomalies we will normalize the tables or split the table or join the table.

3) Briefly explain Null values with syntaxes?

Ans NULL Values:

A NULL value in a relational database is used when the value in a column is unknown or missing.

→ A NULL value is neither a empty string nor a zero value.

→ The term NULL in SQL is used to specify that a data value does not exist in the database.

→ We must use "IS NULL" or "IS NOT NULL" operators to check if a value is NULL.

IS NULL – to test empty values

IS NOT NULL – Specifies that a column should always accept an explicit values of the given data type.

Syntax for IS NULL: → Select column_name from table_name WHERE column_name is NULL;

Ex: Student

S_ID	S_name	S_age	S_email
1	ABC	20	ABC@gmail.com
2	XYZ	30	XYZ@gmail.com
3	Raju	19	Raju@gmail.com
4	Ravi	18	NULL

Select * from Student WHERE S_email is NULL;

S_ID	S_name	S_age	S_email
4	Ravi	18	NULL

Syntax for IS NOT NULL :→ select column_name from table_name
WHERE column_name is NOT NULL;

Ex: Select * from student WHERE s_email is NOT NULL;

O/p:

S_ID	S_name	S_age	S_email
1	ABC	20	ABC@gmail.com
2	XYZ	30	XYZ@gmail.com
3	Raju	19	Raju@gmail.com

4) what are the basic SQL commands with example.

Ans Basic SQL commands :→

(i) CREATE TABLE: The create table statement is used to define a new table to create the students relation we can use the following statements.

Ex: CREATE TABLE students (sid char(20), name char(30), login char(20), age integer, gpa REAL);

(ii) INSERT: Tuples are inserted using the INSERT command and we can use Insert to insert a single tuple into the students table as follows

INSERT into students (sid, name, login, age, gpa)
VALUES (53688, 'smith', 'smith@cse', 18, 3.2), (53689, 'Jack', 'Jack@cse', 19, 8.5);

(iii) DELETE: We can delete tuples using the DELETE command. We can delete all students tuples with name = smith using this command.

DELETE FROM students WHERE name = 'smith';

(iv) UPDATE: We can modify the values in the existing row using the UPDATE command.

UPDATE students SET age = age+1, gpa = gpa-1
WHERE sid = 53689;

sid	name	login	age	gpa
53689	Jack	Jack@cse	20	7.5

5) List out the rules of Armstrong Axioms?

Ans Armstrong Axioms :→

The Armstrong Axioms are a set of rules, that when applied repeatedly, generates a closure of functional dependencies. If F is a set of functional dependencies then the closure of F, denoted as F^+, is the set of all functional dependencies logically implied by F.

Rules of Armstrong Axioms :→

(i) Axioms of Reflexivity: If A is a set of attributes and B is a subset of A, then A holds B. If $B \subseteq A$ then $A \rightarrow B$. This property is trivial property.

(ii) Axioms of Augmentation: If $A \rightarrow B$ holds & Y is the attribute set, then $AY \rightarrow BY$ also holds. That is adding attributes to dependencies, does not change the basic dependencies. If $A \rightarrow B$, then $AC \rightarrow BC$ for any C.

(iii) Axioms of transitivity: Same as the transitive rule in Algebra, if $A \rightarrow B$ holds and $B \rightarrow C$ holds, then $A \rightarrow C$ also holds. $A \rightarrow B$ is called A functionally which determines B.

If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$.

5MARKS

i) Explain Functional dependency with closure and Attribute closure.

Ans Functional dependency: Functional dependency is a relation between the attributes of a table related to eachother. It helps to maintain the quality of the data in the database.

Functional dependency is denoted by "$\longrightarrow$". The F.D of x on y is represented by $x \rightarrow y$.

closure of a set of FD's :→

→ The set of all F.D's implied by a given set of F of FDs is called the closure of F, denoted by F^+.

→ The following three rules called Armstrong's Axioms can be applied repeatedly to infer all FD's.

X, Y, Z are set of attributes over a relation schema R

Reflexivity: If $X \supseteq Y$ then $X \rightarrow Y$

Augmentation: If $X \to Y$, then $XZ \to YZ$ for any Z (partial dependency).

Transitivity: If $X \to Y$ and $Y \to Z$ then $X \to Z$.

union: If $X \to Y$ and $X \to Z$ then $X \to YZ$.

Decomposition: If $X \to YZ$ then $X \to Y$ and $X \to Z$.

Attribute closure: The Attribute closure X^+ with respect to F, which is the set of attributes A such that $X \to A$ can be inferred using the Armstrongs Axioms.
The Algorithm for compuling the Attribute closure of a set X of attributes is shown below:

```
closure = X;
repeat until there is no change: {
        if there is an FD U → V in F such that U ⊆ closure,
            then set closure = closure ∪ V.
}
```

2) Explain Aggregate operators with example.

Ans Aggregate operators: An Aggregate operators in SQL performs calculations on multiple values and return a single value.
→ An Aggregate operators ignores NULL values when it performs calculations except for the count function.
→ SQL provides many aggregate functions that include avg(), count(), sum(), MIN(), MAX(), etc.

Ex: EMP_DATA Table

EMP_ID	Name	Dept	Salary
1	Ram	Marketing	80,000
2	Henry	production	76,000
3	Disha	R&D	76,000
4	Helen	R&D	84,000
5	Meera	Marketing	80,000
6	Aisha	production	64,000
7	Bhuvan	production	60,000
8	Hari	R&D	60,000
9	Preethi	Marketing	NULL
10	Mark	production	66,000

COUNT(): COUNT() returns the total no.of rows from a database table that matches the defined criteria in the SQL query
Syntax: COUNT(*) OR
COUNT(COLUMN_NAME)

Ex: SELECT COUNT(salary) FROM EMP_DATA;

O/p: 9

SUM(): The SUM() function takes the name of the column as an argument and returns the sum of the all non-NULL values in the column. It works only in numeric fields.

Ex: select SUM(salary) FROM EMP_DATA;

O/p: 646,000

AVG(): The AVG() function uses the name of the column as an argument and returns the average values of all the non NULL values in the column. It works only for numeric fields.

Ex: select AVG(salary) FROM EMP_DATA;

O/p: 71777.477

MIN(): MIN() function takes the name of the column as an argument and returns the minimum value present in the column.

Ex: select MIN(salary) FROM EMP_DATA;

O/p: 60,000.

MAX(): MAX() function takes the name of the column as an argument and returns the maximum value present in the column.

Ex: select MAX(salary) FROM EMP_DATA;

O/p: 84,000.

GROUP BY: The GROUP BY statements in SQL is used to arrange the identical data into groups with the help of some functions.

Syntax: select NAME, SUM(COLUMN-NAME) FROM TABLE_NAME
GROUP BY COLUMN-NAME;

HAVING: The HAVING clause is used in DBMS to fetch the data/values from the groups according to the given condition.

3) Define Schema Refinement and explain problems caused by redundancy.

Ans Schema Refinement: Schema Refinement is a technique of organizing the data in the database.

* It is a systematic approach of decomposing tables to eliminate data redundancy and undesirable characteristics like insertion, update and deletion anomalies.
* Redundancy means duplicate copies of the same data.

Problems caused by Redundancy:-

* Storing the same information redundantly, that is in more than one place within a database, can lead to several problems.

→ A database Anomaly is an inconsistency in the data resulting from an operation like an update, delete & insertion.

→ Redundant storage: some information is stored repeatedly.

→ update Anomalies: when we update some rows in the table, and if it leads to the inconsistency of the table then this anomaly occurs.

→ Insertion Anomalies: If there is a new row inserted in the table and it creates the inconsistency in the table.

→ Deletion Anomaly: If we delete some rows from the table and if any other information or data which is required is also deleted from the database.

→ To remove this type of anomalies we will normalize the table or split the table or join the tables.

4) Explain the following operators in SQL with an example
1. UNION 2. INTERSECTION 3. EXCEPT

Ans Operators in SQL :→

(i) UNION (U): (RUS)

→ returns a relation instance containing all tuples that occur in either relation instance R or relational instance S or both.

Example:

S_1

S-ID	S-name	Rating	age
22	Dustin	7	45
31	Rusty	8	55
58	Puppy	10	35

S_2

S-ID	S-name	rating	age
28	yuppy	9	35
31	Rusty	8	55
44	Gumpy	5	35
58	puppy	10	35

$S_1 \cup S_2$

S-ID	S-name	Rating	age
22	Dustin	7	45
31	Rusty	8	55
28	yuppy	9	35
44	Gumpy	5	35
58	puppy	10	35

(ii) INTERSECTION (R∩S) :→

→ Returns a relationship instance containing all tuples that occur in both R and S.

→ The relation R and S must be union compatible and the schema of the result is defined to be identical to schema of R

Ex: $S_1 \cap S_2$

S-ID	S-name	Rating	age
31	Rusty	8	55
58	Puppy	10	35

(iii) EXCEPT: Finds the difference between two queries and the result compares the two rows that belongs only to the first query.

Ex:

S-ID	S-name	Rating	age
22	Dustin	7	45

5) Describe the DDL (Data Definition language)
1. create 2. Alter 3. Drop 4. Truncate

Ans (i) CREATE(): It defines each column of the table uniquely. Each column has minimum of three attributes a name, datatype & size.

(ii) ALTER(): Alter command is used to add the column to the already created table.

(iii) DROP(): Drop command is used to delete a row or even a complete table.

(iv) TRUNCATE(): Truncate command is used to delete all the rows of a table for permanent.

Ex:
```
create database student;
use student;
create table stud (std_id int, stdname varchar(30), std_age int);
Insert into stud values (1,"Ram", 18), (2, "Raju", 19), (3, "shyam", 20);
```
Select * from stud;

std_id	std_name	std_age
1	Ram	18
2	Raju	19
3	shyam	20

```
Alter table student add (smarks int);
```
Select * from stud;

std_id	std_name	std_age	s_marks
1	Ram	18	
2	Raju	19	
3	shyam	20	

```
Alter table stud drop column s_marks;
Select * from stud;
```

stdid	std_name	std_age ~~s_marks~~
1	Ram	18
2	Raju	19
3	shyam	20

10MARKS

1) What is Trigger? Explain how to implement trigger in SQL.

Ans Triggers: A Trigger is a procedure which is automatically invoked by the DBMS in response to changes to the database and is specified by the DBA.

A database with a set of associated triggers is generally called an active database.

Parts of trigger:

(a) event: An event is a change to the database which activates the trigger.

(b) condition: A query or test that is run when the trigger is activated.

(c) Action: A procedure that is executed when the trigger is activated and its condition is true.

An Insert, delete, update statement could activate a trigger.

A Query is interpreted as true if the answer set is nonempty and false if the query has no answers.

→ If the condition part evaluates to true, the action associated with the trigger is executed.

→ A trigger action can examine the answers to the query in the condition part of the trigger.

→ An important issue is when the action part of a trigger executes in a relation to the statement that activated the trigger.

For example, a statement that inserts record into a student table may activate a trigger that is used to maintain statistics on how many students younger than 18 are inserted at a time by a typical insert statement.

Example of triggers

```
create trigger init count Before Insert on students /* event */
Declare
    count Integer;
Begin                          /* Action */
    count := 0;
```

```
END
Create trigger incr count After Insert on student /* Event */
when (new.age <18) / * condition */
for each row
 Begin                    /* Action */
    count := count + 1;
 End
```

The trigger called incr count increment the counter for each inserted tuple that satisfies the condition age<18.

Statement level trigger: It is fixed only once for DML statement irrespective of no. of rows affected by the statement

These are default type of trigger.

Row level trigger: It is fixed for each row that is affected by DML command.

2) Explain Normalization with 1NF, 2NF, 3NF, BCNF, 4NF and 5NF?

Ans Normalization: Normalization is a process of minimizing redundancy from a relation or a set of relations.

→ Redundancy in a relation may cause insertion, deletion and update Anomalies. So it helps to reduce redundancy in relations.

→ In DBMS, normal forms are a series of guidelines that helps to ensure that the design of a database is efficient, organized and free from data anomalies.

First Normal Form: (1NF)

This is the most basic level of normalization. In 1NF, each table cell should contain only a single value, and each column should have a unique name. The 1NF helps to eliminate duplicate data.

ex:

course	content
programming	Java, C++
Web	HTML, PHP, ASP

Course	content
Programming	Java
Programming	C++
Web	HTML
Web	PHP
Web	ASP

Second normal Form (2NF): 2NF eliminates redundant data by requiring that each non-key attribute be dependent on the primary key

→ This means that each column should be directly related to the primary key and not to other columns.

Ex: student

RNO	Name	Group	Fee
1	Ram	CSE	10K
2	Shyam	ECE	15K
3	Raju	ECE	12K

Group

R.NO	Lang-Known
1	C
1	DBMS
2	C
2	Java
3	DBMS

R.NO	Name	Lang-Known	Group	Fee
1	Ram	C	CSE	10K
1	Ram	DBMS	CSE	10K
2	Shyam	C	ECE	15K
2	Shyam	Java	ECE	15K
3	Raju	DBMS	ECE	12K

Third normal form (3NF): For a relation in 3NF, it must be in 2NF and (a) no non prime attribute is transitively dependent on prime key attribute (b) For any non trivial functional dependency, $X \rightarrow A$, then either - x is a super key or A is a prime Attribute.

Ex: student

R.NO	Name	Group
1	Ram	CSE
2	Shyam	ECE
3	Raju	ECE

Fee

Group	Fee
CSE	10K
ECE	15K
ECE	12K

Group

R.NO	Lang-Known
1	C
1	DBMS
2	C
2	Java
3	DBMS

BCNF: BCNF is the advanced version of 3NF

Section	Subject	Faculty	Time
A	Comp	Sudhakar	8:00
A	Phy	Raju	9:00
B	Comp	Ram	10:00
D	Comp	Shyam	11:00

Section	Faculty	Time
A	Sudhakar	8:00
A	Raju	9:00
B	Ram	10:00
D	Shyam	11:00

Subject	Faculty
comp	sudhakar
phy	Raju
comp	Ram
comp	Shyam

Fourth Normal Form (4NF): For a relation in 4NF, it must satisfies BCNF and no multivalued dependency

Ex: Student

St_Id	course	Hobby
21	comp	Dancing
24	maths	singing
34	chem	Dancing
74	Bio	cricket
59	phy	Hockey

std-course

Std_Id	course
21	comp
24	maths
34	chem
74	Bio
39	phy

std-Hobby

Std_Id	Hobby
21	Dancing
24	singing
34	Dancing
74	Cricket
39	Hockey

Fifth normal Form (5NF): For a relation in 5NF, it must satisfies 4NF and does not contain any join dependencies

Ex:

subject	Lecturer	sem
comp	Anshika	1
comp	John	1
maths	John	1
maths	Akash	2
chem	praveen	1

subject	Lecturer
comp	1
comp	1
maths	1
maths	2
chem	1

subject	sem
comp	1
comp	1
maths	1
maths	2
chem	1

Lecturer	sem
Anshika	1
John	1
John	1
Akash	2
praveen	1

3) a) Explain lossless join decomposition?
b) Enumarate Multi-valued Dependancy (MVD)?

Ans **Lossless join decomposition**: lossless join decomposition in DBMS, a relation R has to undergo decomposition in two relations R_1 & R_2, the relation R which is original relation is obtained.

→ In lossless join decomposition, a relation is decomposed into two or more relations. Hence, the original data is preserved & it is ensured that the original data and data after reconstruction should be same.

→ Another name for lossless join decomposition is non-additive join decomposition.

→ for lossless join decomposition, we select a common attribute.

→ The criteria for selecting the same is that it has to be a candidate key or foreign key or superkey in either relations R_1, R_2 or both.

→ In addition (a) the union of the subrelations relation1 & relation2 should be containing all the attributes that may present in the original relation R which got decomposed.

(b) The intersection of subrelations relation1 & relation2 is not allowed to be NULL. It is necessary for the subrelation to contain a common attribute.

Ex: consider the following relation $R = \{A, B, C\}$. The relation R has attributes A, B, C. The relation R decomposed into 2 relations R_1 and R_2 with common attribute B.

$R_1 = (A, B)$ and $R_2 = (B, C)$. Also, it is important to remember that the value present in the column B should be unique. If there is any presence of a duplicate value, it is not possible for lossless join decomposition to take place.

R

A	B	C
1	2	1
2	5	3
3	3	3

$R = \{A, B, C\}$

R_1

A	B
1	2
2	5
3	3

$R_1 (A, B)$

R_2

B	C
2	1
5	3
3	3

$R_2 (B, C)$

=>

A	B	C
1	2	1
2	5	3
3	3	3

=> $R = (A, B, C)$

Multi-valued Dependency (MVD): Multi-valued dependency states that two independent data variables B and c in a table can be dependent on third variable A respectively.

→ MVD occurs when two attributes in a table are independent of eachother but are dependent on third variable.

→ since, MVD consists of a minimum of 3 variables to be dependent on third variable. so, the minimum amount of variables

required is '2'.

* Representation of MVD is B, C → A.

MVD: a —→ —→ b a is multivalued dependent on B.

If we say MVD exist by these following conditions or not.

If any Attribute A has multiple dependency on B for any Relation R, for all pairs of data values in row R_1 & row R_2 in a table such that the relation $R_1(a) = R_2(a)$ exists, and the relation b/w row R_3 & row R_4 exists in a table such that

$R_1(a) = R_2(a) = R_3(a) = R_4(a)$

$R_1(b) = R_3(b)$; $R_2(b) = R_4(b)$ then we can say that multivalued dependency exist which means, in the row R_1, R_2, R_3 & R_4.

Ex:

Name	Course	Hobby
Rahul	c++	painting
Rahul	Python	music
Rahul	c++	music
Rahul	Python	painting

Verification:

Condition 1 MVD $R_1(a) = R_2(a) = R_3(a) = R_4(a) =$ Rahul

Condition 2 MVD $R_1(b) = R_3(b) =$ c++ and $R_2(b) = R_4(b) =$ Python

Condition 3 MVD $R_1(c) = R_4(c) =$ painting and $R_2(c) = R_3(c) =$ music

Name —→ —→ Course and Name —→ —→ Hobby.