Important Questions Part-1

MYSQL REVISION TOUR

1. What do you mean by a Database Management System?

Ans- Database Management is a collection of programs and files that allow a user to define structure of a database, store data into it, modify the structure and manipulate the data.

2. What do you mean by Relational database?

Ans. Relational Database is a type of database in which relation is used as its basic element.Row and columns are used to store data.

3. What is a foreign key?

Ans If a key is available in a table as a primary key then this key is called foreign key in another table.

4. What is primary key?

Ans-Primary key is a unique key in a relation which can uniquely identifies a tuple (row) in a given relation.

5. What is SQL?

Ans- SQL is stands for structured query language. This language is used to manipulate data stored in a table.

6. What is referential integrity?

Ans- This is a rule which ensures that in DBMS relationships between records in related tables are valid. And that user don't accidently delete or change related data.

7. What is MySql?

Ans- Mysql is an open source RDBMS which uses SQL.

8. What is DDL?

Ans- DDL provides commands to define or redefine the schema of a table.Table is created,altered and dropped using DDL.

9. What are DML commands?

Ans- DML commands are used to manipulate data stored in a table.Insertion, deletion and modifications are possible using DML commands.

10. Maximum how many characters can be stored in a (i) text literals (ii) numeric literal

Ans- (i) Text literals can have 4000 bytes (ii) A numeric literals can store 53 digits.

11. What is null value in MySql?

Ans- If a column in a row has no value, then column is said to be null.

12. Which keyword eliminates redundant data in from a query result?

Ans- DISTINCT

13. How would you display system date as the result of a query?

Ans- CURDATE()

14. What is NOW() function in MySql?

Ans- It returns the current date and time.

15. What is NOT NULL constraint?

Ans- NOT NULL constraints impose a condition that value of a row cannot be left blank.

16. What is error in following statement?

UPDATE EMPL;

Ans- WHERE clause is missing in given query.

17. Identify the error?

DELETE ALL FROM TABLE EMP;

Ans-There is no need to write ALL and TABLE word in above query.

Correct form is-DELETE FROM EMP;

18. Differeniate WHERE and HAVING clause?

Ans- Where clause is used to select particular rows that satisfy condition whereas having clause is used in connection with aggregate function, group by clause.

19. How SQL commands are are classified?

Ans-SQL Commands are classified into three categories

(i) Data Definition Language (DDL)-Commands that allow us to perform tasks related to

data definition. E.g. creating, altering and dropping
(ii) Data Manipulation Language (DML) - Commands that allows us to perform data manipulation e.g retrieval, insertion, and modification of data stored in a database.
(iii) Transaction Control Language (TCL)-Commands that allow us to manage and control the transactions.

20. What is difference between char and varchar?
Ans-The difference between char and varchar is that of fixed length and variable length.The CHAR datatypes specifies a fixed length character string.When a column is given datatype as CHAR(n) then MySQL ensures that all values stored in that column have this length.But on other hand when a column is given datatype as VARCHAR(n) ,then the maximum size of a value in this column stores exactly what we specify.

21. What do you understand by the terms primary key and degree of a relation in relational data base?
Ans: Primary Key: A primary key is a set of one or more attributes that can uniquely identify tuples within the relations. The number of attributes in a relation is called Degree of arelation in relational data base.

22. What do you understand by the candidate key and cardinality of a relation in relational data base?
Ans. Candidate Key: All attribute combinations inside a relation that can serve as primary key (uniquely identifies a row in a relation) are Candidate Keys as they are candidates for the primary key position. The number of rows in a relation is known as cardinality of a relation.

23. Consider the following tables Item and Customer. Write SQL commands for the statement (i) to (iv) and give outputs for SQL queries (v) to (viii)
Table: ITEM

S.no	I_ID	Item Name	Manufacturer Price
01	PC01	Personal Computer	ABC 35000
02	LC05	Laptop	ABC 55000
03	PC03	Personal Computer	XYZ 32000

04	PC06	Personal Computer	COMP 37000
05	LC03	Laptop	PQR 57000

Table: CUSTOMER C_ID Customer Name City I_ID

S.no	CUSTOMER C_ID	Customer Name	City	I_ID
01	01	N.Roy	Delhi	LC03
02	06	H.Singh	Mumbai	PC03
03	12	R.Pandey	Delhi	PC06
04	15	C.Sharma	Delhi	LC03
05	16	K.Agrawal	Bangalore	PC01

(i) To display the details of those Customers whose city is Delhi
Ans: Select all from Customer Where City=”Delhi”

(ii) To display the details of Item whose Price is in the range of 35000 to 55000 (Both values included).
Ans: Select all from Item Where Price>=35000 and Price <=55000

(iii) To display the Customer Name, City from table Customer, and Item Name and Price from table Item, with their correspondingmatching I_ID.
Ans: Select Customer Name, City, ItemName, Price from Item, Customer where Item.I_ID=Customer.I_ID.

(iv) To increase the Price of all Items by 1000 in the table Item.
Ans: Update Item set Price=Price+1000

(v) SELECT DISTINCT City FROM Customer.
Ans: City Delhi, Mumbai, Bangalore

(vi) SELECT Item Name, MAX(Price), Count(*) FROM Item GROUP BY Item Name;
Ans: Item Name Max(Price) Count(*) Personal Computer 37000 3 Laptop 57000 2

(vii) SELECT Customer Name, Manufacturer FROM Item, Customer WHERE Item.Item_Id=Customer.Item_Id;

Ans: Customer Name Manufacturer Name

N.Roy PQR

H.Singh XYZ

R.Pandey COMP

C.Sharma PQR

K.Agarwal ABC

(viii) SELECT Item Name, Price * 100 FROM Item WHERE Manufacturer = 'ABC';

Ans: Item Name Price*100

Personal Computer 3500000

Laptop 5500000

Important Questions Part-2

DATABASE TRANSACTIONS

1. Define a transaction.

Ans. A transaction is a logical unit of a work that must succeed or fail in its entirely. It is an atomic operation which can be divided unto smaller operations.

2. What are the two ways in which multiple transactions can be executed?

Ans. Multiple transactions can be executed in one of the following two ways:
(i) Serially (ii) Concurrently

3. What is a savepoint?

Ans. Savepoints are special operations that allow you to divide the work of a transaction into different segments. In case of a failure, you can execute rollbacks to the savepoint only, leaving prior changes intact.

4. What to you understand by a database transaction?

Ans. A database transaction is a logical unit of work that must succeed or fail in its entirely.

5. Why do understand by transaction COMMIT and ROLLBACK?

Ans-COMMITing a transaction means all the steps of a transaction are carried out successfully and all data changes are made permanent in the database. Transaction ROLLBACK means transaction has not been finished completely and hence all data changes made by the transaction in the database if any, are undone and the database returns to the state as it was before this transaction execution started.

6. What do you understand by ACID properties of database transaction?

Ans. To ensure the data-integrity, the database system maintains the following properties of transaction. The properties given below are termed as ACID properties-an acronym derived from the first letter of each of the properties.

(i) Atomicity-This property ensures that either all operations of the transactions are reflected properly in the database, none are. Atomicity ensures either al-or-none operations of a transaction are carried out.

(ii) Consistency-This property ensures that database remains in a consistent state before the

start of transaction and after the transaction is over.

(iii) Isolation-Isolation ensures that executing transaction execution in isolation i.e. is unaware of other transactions executing concurrently in the system.

(iv) Durability-This property ensures that after the successful completion of a transaction i.e when a transaction COMMITs, the changes made by it to the database persist i. e remain in the database irrespective of other failures.

7. What the function is of redo and undo logs?

Ans. Every database has a set of redo log files. It records all change in data including both committed and uncommitted changes. Undo logs stored roll backed data.

8. What TCL commands are supported by SQL?

Ans. SQL supports following TCL commands

BEGIN |START TRANSACTION-Marks the beginning of a transaction COMMIT-Ends the current transaction by saving database changes and starts a new transaction.

ROLLBACK-Ends the current transaction by discarding changes and starts a new transaction.SAVEPOINT-Defines breakpoints for the transactions to allow partial rollbacks.

SET AUTOCOMMIT-Enables or disable the default autocommit mode.

9. Which two statements complete a transaction?

A. DELETE employees;

B. DESCRIBE employees;

C. ROLLBACK TO SAVEPOINT C;

D. GRANT SELECT ON employees TO SCOTT;

E. ALTER TABLE employees

MODIFY COLUMN sal;

F. Select MAX(sal)

FROM employees

WHERE department_id=20;

Ans. - C, E

Important Questions Part-3

More on SQL-Grouping Records and Table Joins

1. Which of the following will give the same Ans irrespective of the NULL values in the specified column:

a. MIN()

b. MAX()

c. SUM()

d. None of the above

Ans-c

2. An aggregate function:

a. Takes a column name as its arguments

b. May take an expression as its argument

c. Both (a) and (b)

d. None of (a) and (b)

Ans-c

3. HAVING is used in conjunction with

a. WHERE

b. GROUP BY clause

c. Aggregate functions

d. None of the above

Ans-b

4. In the FROM clause of

a. SELECTatement

b. Multiple table names are specified.

c. Multiple Column Names may be specified.

d. Multiple table names may be specified.

Ans-a

5. JOIN in RDBMS refers to

a. Combination of multiple columns
b. Combination of multiple rows
c. Combination of multiple tables
d. Combination of multiple databases
Ans-c

6. Equi-join is formed by equating Equi-join is formed by equating
a. Foreign key with Primary key
b. Each row with all other rows
c. Primary key with Primary key
d. Two tables
Ans-a

7. Referential integrity
a. Must be maintained
b. Cannot be maintained
c. Is automatically maintained by databases
d. Should not be maintained
Ans-a

8. A Primary key column
a. Can have NULL values
b. Can have duplicate values
c. Both (a) and (b)
d. Neither (a) nor (b)
Ans-d

9. Primary Key of a table can be
a. Defined at the time of table creation only.
b. Defined after table creation only.
c. Can be changed after table creation
d. Cannot be changed after table creation
Ans-a

10. Two SELECT commands in a UNION

a. Should select same number of columns.
b. Should have different number of columns
c. Both (a) and (b)
d. Neither (a) nor (b)
Ans-c

Very Short Question Ans

1. Why is it not allowed to givering and Date type arguments for SUM() and AVG() functions? Can we give these type of arguments for other functions?
Ans :ring and dates are not real numbers that we calculate so sum or avg functions are not valid for them.

2. What is default, Autocommit mode in MySQL ?
Ans : By default, Autocommit mode is on in MySQL.

3. Can where be added a savepoint in a transaction ?
Ans : We can add a savepoint anywhere in a transaction.

4. How are NULL values treated by aggregate functions?
Ans : None of the aggregate functions takes NULL into consideration. NULL is simply ignored by all the aggregate functions.

5. There is a column C1 in a table T1. The following twoatements: SELECT COUNT(*)FROM T1; and SELECT COUNT(C1) from T1; are giving different outputs. What may be the possible reason?
Ans : There may be a null value.

6. What is the purpose of GROUP BY clause?
Ans : GROUP BY: GROUP BY clause is used in a SELECTatement in conjunction with aggregate functions to group the result based on distinct values in a column.

7. What is the difference between HAVING and WHERE clauses? Explain with the help of an example.
Ans : WHERE Vs HAVING: WHERE is used to put a condition on individual row of a table whereas HAVING is used to put condition on individual group formed by GROUP BY clause in

a SELECTatement.

8. What is a Foreign key? What is its importance?
Ans : Foreign Key: It is a column of a table which is the primary key of another table in the same database. It is used to enforce referential integrity of the data.

9. What are constraints? Are constraints useful or are they hindrances to effective management of databases?
Ans : These are the rules which are applied on the columns of tables to ensure data integrity and consistency. These play very important role for tables so are not hindrances.

10. In a database there is a table Cabinet. The data entry operator is not able to put NULL in a column of Cabinet? What may be the possible reason(s)?
Ans : Not NULL or Primary key constraints used.

11. In a database there is a table Cabinet. The data entry operator is not able to put duplicate values in a column of Cabinet? What may be the possible reason(s)?
Ans : Primary key constraint used.

12. Do Primary Key column(s) of a table accept NULL values?
Ans : No.

13. There is a table T1 with combination of columns C1, C2, and C3 as its primary key? Is it possible to enter:
a. NULL values in any of these columns?
b. Duplicate values in any of these columns?
Ans : No.

14. What are the differences between DELETE and DROP commands of SQL?
Ans : Delete is used for row removing while drop is used for removing complete table.

15. What are Aggregate Functions ?
Ans : A multiple row function works on multiple values. These functions are called aggregate functions or group functions.

16. for what Data Types aggregate functions : MIN(), MAX(), and COUNT() work?
Ans : on any type of values - Numeric, Date, orring. AVG(), and SUM() work on only

Numeric values (INT and DECIMAL).

17. What is HAVING clause ?

Ans : HAVING clause is used in conjunction with GROUP BY clause in a SELECTatement to put condition on groups.

18. What is Referential Integrity ?

Ans : The property of a relational database which ensures that no entry in a foreign key column of a table can be made unless it matches a primary key value in the corresponding column of the related table.

19. What is Union used for ?

Ans : Union is an operation of combining the output of two SELECTatements.

20. What is ALTER TABLE ?

Ans : ALTER TABLE command can be used to Add, Remove, and Modify columns of a table. It can also be used to Add and Remove constraints.

21. What is DROP TABLE ?

Ans : DROP TABLE command is used to delete tables.

22. What function is used whenever a condition involves an aggregate function ?

Ans : whenever a condition involves an aggregate function, then we use HAVING clause in conjunction with GROUP BY clause.

23. What is Difference between GROUP BY' and Having functions ?

Ans : WHERE function is used for individual records and HAVING for groups . GROUP BY function is used for getting results based on some groups of data while a condition on groups is applied by HAVING clause.

Short Questoins Ans

1. Why are aggregate functions called so? Name some aggregate functions.

Ans : A multiple row function works on multiple values. These functions are called aggregate functions or group functions. Some of the most frequently used. Aggregate functions in MySQL are

: MIN(), MAX(), AVG(), SUM(), COUNT().

Q2. What is ALTER TABLE command ?Write all the commands that can be applied using alter table.

Ans : a new column can be added to a table using ALTER TABLE command. ALTER TABLE can be used:

• to add a constraint

• to remove a constraint

• to remove a column from a table

• to modify a table column

Q3. What is the Cartesian product of two table? Is it same as an Equi-join?

Ans : Cartesian Product (or Cross Join): Cartesian product of two tables is a table obtained by pairing each row of one table with each row of the other. A cartesian product of two tables contains all the columns of both the tables.

Equi-Join: An equi join of two tables is obtained by putting an equality condition on the Cartesian product of two tables. This equality condition is put on the common column of the tables. This common column is, generally, primary key of one table and foreign key of the other.

LONG QUESTION-Ans

1. Does Union display any duplicate rows ?

Ans :Union does not display any duplicate rows unless ALL is specified with it.

2. Name the Aggregate Functions.

Ans : These functions are:

S. No.	Name of the Function	Purpose
1	MAX()	Returns the MAXIMUM of the values under the specified column/expression.
2	MIN()	Returns the MINIMUM of the values under the specified column/expression.

3	AVG()	Returns the AVERAGE of the values under the specified column/expression.
4	SUM()	Returns the SUM of the values under the specified column/expression.
5	COUNT()	Returns the COUNT of the number of values under the specified column/expression.

2. What is Max Function ? Give few Examples.

Ans. MAX() function is used to find the highest value of any column or any expression based on a column. MAX() takes one argument which can be any column name or a valid expression.

involving a column name. e.g.,

To find the highest cost of any type of shoe in the factory.

SELECT MAX(cost) FROM shoes;

MAX(cost)

843.00

To find the highest cost of any shoe of type 'School'.

SELECT MAX(cost) FROM shoes WHERE type ='School';

MAX(cost) 320.75 To find the highest selling price of any type of shoe.

SELECT MAX(cost+cost*margin/ 100) FROM shoes;

MAX(cost+cost*margin/100)

828.517500000

To find the highest selling price of any type of shoe rounded to 2 decimal places.

SELECT ROUND(MAX(cost+cost*mar gin/100),2) AS "Max. SP" FROM shoes;

Max. SP

733.36

To find the highest selling price of any type of shoe rounded to 2 decimal places.

SELECT ROUND(MAX(cost+cost*mar gin/100),2) AS "Max. SP" FROM shoes;

Max. SP 733.36

3. What is min() Function ? Give Some Examples.

Ans MIN() :

MIN() function is used to find the lowest value of any column or an expression based on a

column. MIN() takes one argument which can be any column name or a valid expression involving a
column name. e.g.,
To find the lowest cost of any type of shoe in the factory.
SELECT MIN(cost) FROM shoes;
MIN(cost) 43.00 To find the lowest cost of any shoe of type 'School'.
SELECT MIN(cost) FROM shoes WHERE type ='School';
MIN(cost) 320.75To find the lowest selling price of any type of shoe rounded to 2 decimal places. SELECT ROUND(MIN(cost+cost*mar gin/100),2)
AS "Min. SP" FROM shoes;
Min. SP 135.15

4 . What is AVG() Function ? Give Some Examples.

Ans : AVG() function is used to find the average value of any column or an expression based on a column. AVG() takes one argument which can be any column name or a valid expression involving a column name. Here we have a limitation: the argument of AVG() function can be of numeric (int/decimal) type only. Averages ofring and Date type data are not defined. E.g.,
To find the average margin from shoes table.
SELECT AVG(margin) FROM shoes;
AVG(margin) 2.600000To find the average cost from the shoes table.
SELECT AVG(cost) FROM shoes;
AVG(cost) 491.750000 To find the average quantity inock for the shoes of type Sports.
SELECT AVG(qty) FROM shoes WHERE type ='Sports';
AVG(qty) 580.0000

5. What is Sum() Function ? Give Some Examples.

Ans.SUM() function is used to find the total value of any column or an expression based on a column. SUM() also takes one argument which can be any column name or a valid expression involving a column name. Like AVG(), the argument of SUM() function can be of numeric (int/decimal) type only. Sums ofring and Date type data are not defined. e.g.,
To find the total quantity present in theock
SELECT SUM(Qty) FROM Shoes;
SUM(Qty) 10020

To find the total order quantity
SELECT SUM(order_qty) FROM orders;
SUM(order_qty) 2475
To find the the total value (Quanitity x Cost) of Shoes of type 'Office' present in the inventory
SELECT SUM(cost*qty) FROM shoes WHERE type = 'Office';
SUM(cost*qty) 772000.00

6. What is COUNT() Function ? Give Some Examples.
Ans . COUNT() function is used to count the number of values in a column. COUNT() takes one argument which can be any column name, an expression based on a column, or an asterisk (*). When the argument is a column name or an expression based on a column, COUNT() returns the number of non-NULL values in that column. If the argument is a *, then COUNT() counts the total number of rows satisfying the condition, if any, in the table. e.g.,
To count the total number of records in the table Shoes.
SELECT COUNT(*) FROM shoes;
COUNT(*)13

To count the different types of shoes that the factory produces
SELECT COUNT(distinct type) FROM shoes;
COUNT(distinct type) 3
To count the records for which the margin is greater than 2.00
SELECT COUNT(margin) FROM shoes WHERE margin > 2;

COUNT(margin)
To count the number of customers in 'A' category
SELECT COUNT(*) FROM customers WHERE category ='A';
COUNT(*)
To count the number of orders of quantity more than 300
SELECT COUNT(*) FROM orders WHERE order_qty >300;
COUNT(*)

7. Does aggregate Functions consider Null values.Does NULLs play any role in actual calculations ?
Ans : None of the aggregate functions takes NULL into consideration. NULL is simply ignored by all the aggregate functions. For example, theatement:

SELECT COUNT(*) FROM shoes;
Produces the following output:
COUNT(*)
Indicating that there are 13 records in the Shoes table. Whereas the query:
SELECT COUNT(margin) FROM shoes;
produces the output:
COUNT(margin)
This output indicates that there are 10 values in the margin column of Shoes table. This means there are 3 (13-10) NULLs in the margin column.
This feature of aggregate functions ensures that NULLs don't play any role in actual calculations. the followingatement:
SELECT AVG(margin) FROM shoes;

8. What is AVG() Function ? Give Some Examples. Does NULLs play any role in Average calculations ?

Ans This Function is used to get the Average Value.
produces the output:
AVG(margin)
2.600000
The average margin has been calculated by adding all the 10 non NULL values from themargin column and dividing the sum by 10 and not by 13.

9. What is 'GROUP BY' ? Give Examples.

Ans: GROUP BY function is used for getting results based on some groups of data.
For example, o The management of the shoe factory may want to know what is the total quantity of shoes of various types. i.e., what is the total quantity of shoes of type School, Office, and Sports each.The management may also want to know what is the maximum, minimum, and average margin of each type of shoes. o It may also be required to find the total number of customers in each category. There are many such requirements. SQL provides GROUP BY clause to handle all such requirements. For the above three situations, theatements with GROUP BY clause are given below: In the first situation we want MySQL to divide all the records of shoes table into different groups based on their type (GROUP BY type) and for each group it should display the type and the corresponding total quantity (SELECT type, SUM(qty)). So the completeatement to do this is: SELECT type, SUM(qty) FROM

shoes GROUP BY type;
G1 and the corresponding output is:

Type	SUM(qty)
Office	1100
School	7180
Sports	1740

Similarly, for the second situation theatement is:
SELECT type, MIN(margin), MAX(margin), AVG(margin)
FROM shoes GROUP BY type; G2 and the corresponding output is:

Type	MIN(margin)	MAX(margin)	AVG(margin)
Office	3.00	3.00	3.000000
School	2.00	2.00	2.000000
Sports	3.50	3.50	3.500000

category	COUNT(*)
A	2
B	2
C	1

In the third situation we want MySQL to divide all the records of Customers table into different
groups based on the their Category (GROUP BY Category) and for each group it should display the Category and the corresponding number of records (SELECT Category, COUNT(*)).
So the completeatement to do this is:
SELECT category, COUNT(*) FROM customers GROUP BY category;
G3 Let us have some more examples. Consider the followingatement:
SELECT cust_code, SUM(order_qty)
FROM orders GROUP BY cust_code;
This satement produces the following output. Try to explain this this output.

Cust_code	SUM(order_qty)
C001	1025
C002	750
C003	150
C004	200
C005	350

Do the same for the followingatement also:
SELECT shoe_code, SUM(order_qty) FROM orders GROUP BY shoe_code;

Sho_code	SUM(order_qty)
1001	200
1002	200
1011	550
1012	250
1101	300
1102	350
1103	225
1201	200
1203	200

If you carefully observe these examples, you will find that GROUP BY is always used in conjunction with some aggregate function(s). A SELECT command with GROUP BY clause has a column name and one or more aggregate functions which are applied on that column and grouping is also done on this column only.

10. What is Role of HAVING in SQL. Give Examples. How it is related with Group by ?
Ans .Sometimes we do not want to see the whole output produced by aatement with GROUP BY clause. We want to see the output only for those groups which satisfy some condition. It means we want to put some condition on individual groups (and not on individual records). A condition on groups is applied by HAVING clause. As an example reconsider the

statement G1 discussed above. Theatement produced three records in the output - one for each group. Suppose, we are interested in viewing only those groups' output for which the total quantity is more than 1500 (SUM(Qty) > 1500). As this condition is applicable to groups and not to individual rows, we use HAVING clause as shown below:
SELECT type, SUM(qty) FROM shoes GROUP BY type HAVING SUM(qty) > 1500;

type	SUM(qty)
School	7180
Sports	1740

Now suppose for G2 we want the report only for those types for which the average margin is more than 2. For this, following is theatement and the corresponding output:
SELECT type, SUM(qty) FROM shoes GROUP BY type HAVING AVG(margin) >2;

type	SUM(qty)
Office	1100
Sports	1740

In theseatements if we try to put the condition using WHERE instead of HAVING, we shall get an error. Another way of remembering this is that whenever a condition involves an aggregate
function, then we use HAVING clause in conjunction with GROUP BY clause.

11. What Functions are used for conditions on individual records as well as on groups. Give Examples.
Ans : Situations may also arise when we want to put the conditions on individual records as well as on groups. In such situations we use both WHERE (for individual records) and HAVING (for groups) clauses. This can be explained with the help of the following examples:
• The management of the shoe factory may want to know what is the total quantity of shoes, of sizes other than 6, of various types. i.e., what is the total
quantity of shoes (of sizes other than 6) of type School, Office, and Sports each.
Moreover, the report is required only for those groups for which the total quantity is more than 1500.

• The management may also want to know what is the maximum, minimum, and average margin of each type of shoes. But in this reports shoes of sizes 6 and 7 only should be included. Report is required only for those groups for which the
minimum margin is more than 2. Theatements and their outputs corresponding to above requirements are given below:
SELECT type, SUM(qty) FROM shoes
WHERE size <> 6 Checks individual row
GROUP BY type HAVING sum (qty) > 1500; Checks individual group

type	SUM(qty)
School	3780

SELECT type, MIN(margin), MAX(margin), AVG(margin) FROM shoes WHERE size in (6,7)
GROUP BY type having MIN(margin) > 2;

Type	MIN(margin)	MAX(margin)	AVG(margin)
Office	3.00	3.00	3.000000
Sports	3.50	3.50	3.500000

12. How Will you Display Data from Multiple Tables ?
Ans: To understand this consider the following situations:
• The management of the shoe factory wants a report of orders which lists three columns: Order_No,
corresponding customer name, and phone number. - (MT-1)
In this case order number will be taken from Orders table and corresponding customer name from Customers table.
• The management wants a four-column report containing order_no, order_qty, name of the corresponding shoe and its cost. - (MT-2)
In this case order number and order quantity will b+e taken from Orders table and corresponding shoe name and cost from Shoes table.
• The management wants the names of customers who have placed any order of quantity more than 300. - (MT-3)

In this case Order quantity will be checked in Orders table and for each record with quantity more than 300, corresponding Customer name will be taken from Customers table.

• The management wants a report in which with each Order_No management needs name of the corresponding customer and also the total cost (Order quantity x Cost of the shoe) of the order are shown. - (MT-4) In this case order number will be taken from Orders table and corresponding customer name from Customers table. For the cost of each order the quantity will be taken from Orders table and the Cost from Shoes table.

In all these cases, the data is to be retrieved from multiple tables. SQL allows us to writeatements which retrieve data from multiple tables.

To understand how this is done, consider the following tables of a database.Product

Code	Name
P001	Toothpaste
P002	Shampoo
P003	Conditioner

Supplier

Sup_Code	Name	Address
S001	DC & Company	Uttam Nagar
S0002	SURY Traders	Model Town

Order_table

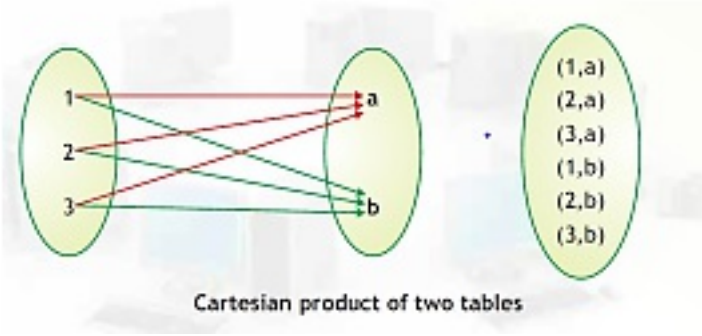
Order_No	P_Code	Sup_Code
1	P001	S002
2	P002	S002

These tables are taken just to explain the current concept.

13. What do you understand by Cartesian Product or Cross Join of tables .Give Example.

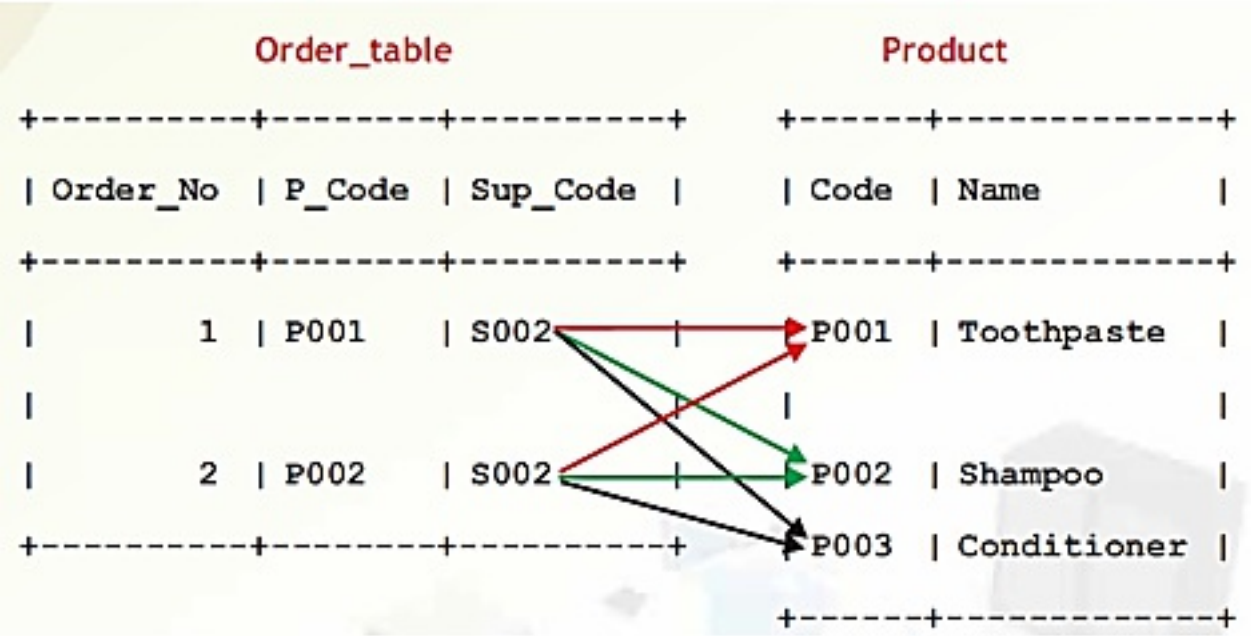
Ans Cartesian product (also called Cross Join) of two tables is a table obtained by pairing up each row of one table with each row of the other table. This way if two tables contain 3 rows

and 2 rows respectively, then their Cartesian product will contain 6 (=3x2) rows. This can be illustrated as follows:



Cartesian product of two tables

Notice that the arrows indicate the 'ordered pairing'. The number of columns in the Cartesian product is the sum of the number of columns in both the tables. In SQL, Cartesian product of two rows is obtained by giving the names of both tables in FROM clause. An example of Cartesian product is shown below: **SELECT * FROM order_table, product;**
To give the output of this query, MySQL will pair the rows of the mentioned tables as follows:



Order No.	p_code	Sup_code	Code	Name
1	P001	S002	P001	Toothpaste
2	P002	S002	P001	Toothpaste
1	P001	S002	P002	Shampoo
2	P002	S002	P002	Shampoo
1	P001	S002	P003	Conditioner
2	P002	S002	P003	Conditioner

Here we observe that the Cartesian product contains all the columns from both tables. Each row of the first table (Order_table) is paired with each row of the second table (Product).B
If we change the sequence of table names in the FROM clause, the result will remain the same but the sequence of rows and columns will change. This can be observed in the followingatement and the corresponding output.
SELECT * FROM product, order_table;

Code	Name	Order_No	P_Code	Sup_Code
P001	Toothpaste	1	P001	S002
P001	Toothpaste	2	P002	S002
P002	Shampoo	1	P001	S002
P002	Shampoo	2	P002	S002
P003	Conditioner	1	P001	S002
P003	Conditioner	2	P002	S002

Q14. Show the Cartesian product of three tables(more than two tables.
Ans : We can have Cartesian product of more than two tables also. Following is the Cartesian Product of three tables:
SELECT * FROM order_table, supplier, product; -(CP-3)

Order_No	P_Code	Sup_Code	Sup_Code	Name	Address	Code	Name
1	P001	S002	S001	DC & Company	Uttam nagar	P001	Toothpaste
2	P002	S002	S00	DC & Company	Uttam nagar	P001	Toothpaste
1	P001	S002	S002	SURY traders	Model town	P001	Toothpaste
				SURY	Model		

2	P002	S002	S002	traders	town	P001	Toothpaste
1	P001	S002	S001	DC & Company	Uttam nagar	P002	Shampoo
2	P002	S002	S001	DC & Company	Uttam nagar	P002	Shampoo
1	P001	S002	S002	SURY traders	Model town	P002	Shampoo
2	P002	S002	S002	SURY traders	Model town	P002	Shampoo
1	P001	S002	S001	DC & Company	Uttam nagar	P003	Conditioner
2	P002	S002	S001	DC & Company	Uttam nagar	P003	Conditioner
1	P001	S002	S002	SURY traders	Model town	P003	Conditioner
2	P002	S002	S002	SURY traders	Model town	P003	Conditioner

The complete Cartesian product of two or more tables is, generally, not used directly. But,sometimes it is required. Suppose the company with the above database wants to send information of each of its products to each of its suppliers. For follow-up, the management wants a complete list in which each Supplier's detail is paired with each Product's detail. For this, the computer department can produce a list which is the Cartesian product of Product and Supplier tables, as follows:

SELECT *, ' ' AS Remarks FROM Product, Supplier;

to get the following report:

Code	Name	Sup_code	Name	Address	Remarks

P001	Toothpaste	S001	Dc & company	Uttam nagar	
P001	Toothpaste	S002	Sury traders	Model town	
P002	Shampoo	S001	Dc & company	Uttam nagar	
P002	Shampoo	S002	Sury traders	Model town	
P003	Conditioner	S001	Dc & company	Uttam nagar	
P003	Conditioner	S002	Sury traders	Model town	

Q15. What is Equi- Join of tables .Show by examples.

Ans. The complete Cartesian product of two or more tables is, generally, not used directly. Sometimes the complete Cartesian product of two tables may give some confusing information also. For example, the first Cartesian product (CP-1) indicates that each order (Order Numbers 1 and 2) is placed for each Product (Code 'P001', 'P002', 'P003'). But this is incorrect! Similar is the case with CP-2 and CP-3 also. But we can extract meaningful information from the Cartesian product by placing someconditions in theatement. For example, to find out the product details corresponding to each Order details,we can enter the following statement:

SELECT * FROM order_table, product WHERE p_code = code;

Order_no IP_Code	Sup_code	Code	Name
1 P001	S002	P001	Toothpaste
2 P002	S002	P002	Shampoo

Two table names are specified in the FROM clause of thisatement, therefore MySQL creates a Cartesian product of the tables. From this Cartesian product MySQL selects only those records for which P_Code (Product code specified in the Order_table table) matches Code (Product code in the Product table). These selected records are then displayed.

It always happens that whenever we have to get the data from more than one tables, there is some common column based on which the meaningful data is extracted from the tables. We specify table names in the FROM clause of SELECT command. We also give the condition

specifying the matching of common column. (When we say common column, it does not mean that the column names have to be the same. It means that the columns should represent the same data with the same data types.) Corresponding to thisatement, internally the Cartesian product of the tables is made. Then based on the specified condition the meaningful data is extracted from this Cartesian product and displayed. Let us take another example of producing a report which displays the supplier name and address corresponding to each order.

SELECT Order_No, Order_table.Sup_Code, Name, Address FROM
order_table, supplier
WHERE order_table.sup_code = supplier.sup_code;

Order_No	Sup_Code	Name	Address
1	S002	SURY Traders	Model Town
2	S002	SURY Traders	Model Town

In thisatement the tables referred are Order_table and Supplier. In these tables sup_code is the common column. This column exists with same name in both the tables. Therefore whenever wemention it, we have to specify the table from which we want to extract this column. This is known as qualifying the column name. If we don't qualify the common column name, theatement would result into an error due to the ambiguous the column names.

Following is another example of equi-join. This time with three tables.

Select Order_no, Product.name as Product, Supplier.Name as Supplier From order_table,
Product,
Supplier
WHERE order_table.Sup_Code = Supplier.Sup_Code and P_Code = Code;

The output produced by thisatement is:

Order_no Product Supplier
1 Toothpaste SURY Traders
2 Shampoo SURY Traders

Let us now get back to our original Shoe database and see how Ms. Akhtar uses the concept of joins to extract data from multiple tables.

For the situation MT-1, she writes the query:

SELECT order_no , name, phone FROM orders, customers WHERE orders.cust_code =

customers.cust_code;
and get the following required output:

Order_no	Name	Phone
1	Novelty Shoes	4543556, 97878989
2	Novelty Shoes	4543556, 97878989
5	Novelty Shoes	4543556, 97878989
9	Novelty Shoes	4543556, 97878989
4	Aaram Footwear	NULL
6	Aaram Footwear	NULL
10	Aaram Footwear	NULL
3	Foot Comfort	51917142, 76877888
7	Pooja Shoes	61345432, 98178989
8	Dev Shoes	NULL

Following are the queries and corresponding outputs for the situations MT-2, MT-3, and MT-4 respectively:
SELECT order_no , Order_Qty, name, cost
FROM orders, shoes WHERE Shoe_Code = code;

order_no	Order_Qty	Name	Cost
1	200	School Canvas	132.50
2	200	School Canvas	135.50
3	150	School Leather	232.50
4	250	School Leather	270.00
5	400	School Leather	232.50

6	300	Galaxy	640.00
7	200	Tracker	700.00
8	350	Galaxy	712.00
9	225	Galaxy	720.00
10	200	Tracker	800.50

SELECT name, address FROM orders, customers WHERE
orders.cust_code = customers.cust_code and order_qty > 300;
name address

Name	**Address**
Novelty Shoes	**Raja nagar**
Dev shoes	**Bhopal**

Dev Shoes Mohan Nagar, Ghaziabad
SELECT order_no, Order_Qty, customers.name,cost*order_qty as 'Order Cost' FROM orders, shoes,
Customers WHERE Shoe_Code = code and Orders.Cust_Code = Customers.Cust_Code order by order_no;

order_no	order_no	Name	Order_cost
1	200	Novelty Shoes	26500.00
2	200	Novelty Shoes	27100.00
3	150	Foot Comfort	34875.00
4	250	Aaram Footwear	67500.00
5	400	Novelty Shoes	93000.00
6	300	Aaram Footwear	192000.00

7	200	Pooja Shoes	140000.00
8	350	Dev Shoes	249200.00
9	225	Novelty Shoes	162000.00
10	200	Aaram Footwear	160100.00

Here is anotheratement extracting data from multiple tables. Try to find out what will be its output and then try thisatement on computer and check whether you thought of the correct output. SELECT order_no , Order_Qty, name, cost FROM orders, shoes WHERE Shoe_Code = code and order_qty > 200;

Q16. Explain the Foreign Key .

As we have just seen, in a join the data is retrieved from the Cartesian product of two tables by giving a condition of equality of two corresponding columns - one from each table. Generally, this column is the Primary Key of one table. In the other table this column is the Foreign key. Such a join which is obtained by putting a condition of equality on cross join is called an 'equi-join'. As an example, once again consider the Product, Supplier, and Order tables referenced earlier. For quick reference these tables are shown once again:

Product

Code	Name
P001	Toothpaste
P002	Shampoo
P003	Conditioner

Supplier

Sup_Code	Name	Address
S001	DC & Company	Uttam Nagar
S002	SURY Traders	Model Town

1	P001	S002
2	P002	S002

In these tables there is a common column between Product and Order_table tables (Code and P_Code respectively) which is used to get the Equi-Join of these two tables. Code is the Primary Key of Product table and in Order_table table it is not so (we can place more than one orders for the same product). In the order_table, P_Code is a Foreign Key. Similarly, Sup_Code is the primary key in Supplier table whereas it is a Foreign Key is Order_table table. A foreign key in a table is used to ensure referential integrity and to get Equi-Join of two tables.

Q17. What do you understand by Referential Integrity ?

Ans : Suppose while eering data in Order_table we enter a P_Code that does not exist in the Product table. It means we have placed an order for an item that does not exist! We should and can always avoid such human errors. Such errors are avoided by explicitly making P_Code a foreign key of Order_table table which always references the Product table to make sure that a non-existing product code is not entered in the Order_table table. Similarly, we can also make Sup_Code a Foreign key in Order_table table which always references Customer table to check validity of Cust_code. This property of a relational database which ensures that no entry in a foreign key column of a table can be made unless it matches a primary key value in the corresponding related table is called Referential Integrity.

Q18. Describe Union operation by giving examples.

Ans Union is an operation of combining the output of two SELECTatements. Union of two SELECT statements can be performed only if their outputs contain same number of columns and data types of corresponding columns are also the same. The syntax of UNION in its simplest form is:

SELECT <select_list> FROM
<tablename> [WHERE
<condition>]
UNION [ALL]
SELECT <select_list> FROM
<tablename> [WHERE
<condition>];

Union does not display any duplicate rows unless ALL is specified with it.
Example:
Suppose a company deals in two different categories of items. Each category contains a number of items and for each category there are different customers. In the database there are two customer tables: Customer_Cat_1 and Customer_Cat_2. If it is required to produce a combined list of all the
customers, then it can be done as follows:

```
SELECT Cust_Code from Customer_Cat_1
UNION
SELECT Cust_Code from Customer_Cat_2;
```

If a customer exists with same customer code in both the tables, its code will be displayed only once - because Union does display duplicate rows. If we explicitly want the duplicate rows, then we can enter theatement:

```
SELECT Cust_Code from Customer_Cat_1
UNION ALL
SELECT Cust_Code from Customer_Cat_2;
```

Q19. What are Constraints for a table ? List all the constraints with their purpose. How these are applied?

Ans Many times it is not possible to keep a manual check on the data that is going into the tables using INSERT or UPDATE commands. The data entered may be invalid. MySQL provides some rules, called Constraints, which help us, to some extent, ensure validity of the data. These constraints are: S.No. Constraint Purpose
2. PRIMARY KEY Sets a column or a group of columns as the Primary Key of a
table. Therefore, NULLs and Duplicate values in this column are not accepted.
3. NOT NULL Makes sure that NULLs are not accepted in the specified column.
4. FOREIGN KEY Data will be accepted in this column, if same data value exists in a column in another related table. This other related table name and column name are specified while creating the foreign key constraint.
5. UNIQUE Makes sure that duplicate values in the specified column are not accepted.
6. ENUM Defines a set of values as the column domain. So any value in this
column will be from the specified values only.
7. SET Defines a set of values as the column domain. Any value in this column will be a seubset of the specied set only. We shall discuss only the PRIMARY KEY and NOT NULL

constraints in this book. Other constraints are beyond the scope of this book.

Q20. What is PRIMARY KEY ? Give Examples.

Ans : Primary key of a table is a column or a group of columns that uniquely identifies a row of the table. Therefore no two rows of a table can have the same primary key value. Now suppose that the table Shoes is created with the followingatement:

CREATE TABLE Shoes
(Code CHAR(4), Name VARCHAR(20), type VARCHAR(10), size INT(2),
cost DECIMAL(6,2), margin DECIMAL(4,2), Qty INT(4));

We know that in this table Code is the Primary key. But, MySQL does not know that. Therefore it is possible to enter duplicate values in this column or to enter NULLs in this column. Both these situations are unacceptable. To make sure that such data is not accepted by MySQL, we can set Code as the primary key of Shoes table. It can be done by using the PRIMARY KEY clause at the time of table creation as
follows:

CREATE TABLE Shoes
(Code CHAR(4) PRIMARY KEY, Name VARCHAR(20),type VARCHAR(10), size
INT(2), cost DECIMAL(6,2), margin DECIMAL(4,2), Qty INT(4));

or as follows:

CREATE TABLE Shoes
(Code CHAR(4), Name VARCHAR(20), type VARCHAR(10), size INT(2),
cost DECIMAL(6,2), margin DECIMAL(4,2), Qty INT(4), **PRIMARY KEY** (Code));

To create a table Bills with the combination of columns Order_No and Cust_Code as the primary key, we enter theatement:

CREATE TABLE bills
(Order_Num INT(4) PRIMARY KEY, cust_code
VARCHAR(4) PRIMARY KEY, bill_Date DATE,
Bill_Amt DECIMAL(8,2));

Contrary to our expectation, we get an error (Multiple primary key defined) with thisatement.

The reason is that MySQL interprets thisatement as if we are trying to create two primary keys of the table - Order_Num, and Cust_code. But a table can have at most one primary key. To set this combination of columns a primary key we have to enter theatement as

follows:CREATE TABLE bills
(Order_Num INT(4), cust_code VARCHAR(4), bill_Date
date, Bill_Amt DECIMAL(8,2), PRIMARY
KEY(Order_Num, cust_code));

Q21. How 'Dese' is used for showingructure of the table ?
Ans : We may check the tableructure with the command: DESC bills;
The tableructure is as shown below:
Field Type Null Key Default Extra

Field	type	null	key	default	Extra
Order_Num	INT(4)	NO	PRI	0	
cust_code	VARCHAR(4)	NO	PRI		
bill_Date	date	YES		NULL	
Bill_Amt	DECIMAL(8,2)	YES		NULL	

These columns constitute the primary key of the table. NULLs cannot be accepted in these columns.

Q22. How will you a create table in which NULL values should not be accepted ?
Ans : Many times there are some columns of a table in which NULL values should not be accepted. We always want some known valid data values in these columns. For example, we cannot have an order for which the customer code is not known. It means whenever we enter a row in the orders table, corresponding customer code cannot be NULL. Similarly while entering records in the Shoes table, we have to mention the Shoe size, it cannot be set NULL. There may be any number of such situations. While creating a table we can specify in which columns NULLs should not be accepted as follows:
CREATE TABLE Shoes
(Code CHAR(4) PRIMARY KEY, Name VARCHAR(20), type
VARCHAR(10), size INT(2) NOT NULL,
cost DECIMAL(6,2), margin DECIMAL(4,2), Qty INT(4)); CREATE TABLE bills
(Order_Num INT(4), cust_code VARCHAR(4), bill_Date DATE,

Bill_Amt DECIMAL(8,2) NOT NULL, PRIMARY KEY
(Order_Num, cust_code));
Now if we try to enter a NULL in the specified column, MySQL will reject the entry and give an
error.

Q23. How can we view the Columns Associated with Constraints ?

Ans. After creating a table, we can view itsructure using DESC command. The tableructure also
includes the constraints, if any. Therefore, when we use DESC command, we are shown the table structure as well as constraints, if any. A constraint is shown beside the column name on which it is
applicable. E.g., theatement:
DESC Shoes;
displays the tableructure as follows:
Field Type Null Key Default Extra

Field	type	null	key	default	Extra
Code	CHAR(4)	NO	PRI	NULL	
Name	VARCHAR(20)	YES			
type	VARCHAR(10)	YES		NULL	
size	INT(2)	NO		0	
cost	DECIMAL(6,2) YES		NULL		
margin	DECIMAL(4,2) YES		NULL		
Qty	INT(4)	YES		NULL	

Q24. Show Add, Modify, and Remove constraints for altering a table.

Ans. If we create a table without specifying any primary key, we canill specify its primary key by ALTER TABLE command. Suppose we have created the Shoes table without specifying any Primary key, then later we can enter theatement as follows:

ALTER TABLE Shoe ADD PRIMARY KEY(code);
This will set Code as the primary key of the table. But if the Code column already contains some duplicate values, then thisatement will give an error. In MySQL, it is also possible to change the primary key column(s) of a table. Suppose, in the Shoes table, istread of Code, we want to set the combination of 'Name' and 'Size' as the primary key. For this first we have to DROP the already existing primary key (i.e., Code) and then add the new primary key (i.e., Name and Size). The correspondingatements are as follows:
ALTER TABLE Shoes DROP PRIMARY KEY;
After thisatement, there is no primary key of Shoe table. Now we can add the new primary key as follows:
ALTER TABLE Shoe ADD PRIMARY KEY (Name, Size);
Now if we see the tableructure by DESC Shoes;atement, it will be shown as follows:

Field	Type	Null	Key	Default	Extra
Code	CHAR(4)	NO		NULL	
Name	VARCHAR(20)	NO	PRI		
type	VARCHAR(10)	YES		Null	
size	INT(2)	NO		0	
cost	DECIMAL(6,2)	YES		Null	
Margin	DECIMAL(4,2)	YES		NULL	
Qty	INT(4)	YES		NULL	

In MySQL, it is not possible to add or drop NOT NULL constraint explicitly after the table creation. But it can be done using MODIFY clause of ALTER TABLE command. As an example, suppose we don't want to accept NULL values in bill_date column of bills table, we can issue the statement:
ALTER TABLE bills MODIFY bill_date DATE NOT NULL;
Later on if we wish to change thisatus again, we can do so by entering the command:
ALTER TABLE bills MODIFY bill_date DATE NULL;
Remove and Modify columns :
ALTER TABLE can be used to remove a column from a table. This is done using DROP clause in
ALTER TABLE command. The syntax is as follws:
ALTER TABLE <tablename> DROP <columnname>

[, DROP <columnname> [, DROP <columnname> [, . . .]]];
Following are some self-explanatory examples of SQLatemenets to remove columns from tables:
ALTER TABLE Shoes DROP Qty;
ALTER TABLE Orders DROP Cust_Code;
ALTER TABLEudent DROP Class, DROP RNo, DROP Section;
Although any column of a table can be removed, MySQL puts the restriction that a primary key column can be removed only if the remaining, primary key columns, if any, do not contain any duplicate entry. This can be understood more clearly with the help of following example:
The Name and Size columns of the Shoe table constitute its primary key. Now if we drop the Name column from the table, Size will be the remaining Primary Key column of the table. Therefore, duplicate entries in the Size column should not be allowed. To ensure this, before removing Name column from the table, MySQL checks that there are no duplicate entries present in the Size column of the table. If there are any, then theatement trying to remove Name column from the table will result in an error and the Name column will not be removed. If there are no duplicate enteries in the Size column, then Name column will be removed. Similar will be the case with the Name column if we try to remove Size column. But there won't be any problem if we try to remove both the primary key columns simultaneously with one ALTER TABLEatement as follows:
ALTER TABLE Shoes DROP name, DROP size;
ALTER TABLE can also be used to change the data type of a table column. For this the syntax is as follows:
ALTER TABLE <tablename> MODIFY <col_name> <new datatype> [,MODIFY
<col_name> <new datatype>
[,MODIFY <col_name> <new data type> [, . . .]]];
e.g., theatement:
ALTER TABLE shoes modify code CHAR(5), modify type VARCHAR(20);
changes the data type of column Code to CHAR(5) and that of type to VARCHAR(20).
When we give aatement to chage the data type of a column, MySQL executes thatatement correctly only if the change in data type does not lead to any data loss. E.g., if we try to change the data type of order_date column of orders table from date to int, we'll get an error. This is because the data alreadyored in this column cannot be converted into int type.

Similarly, if a column of VARCHAR(10) type contains some data value which is 10 characters long, then the data type of this column cannot be converted to VARCHAR(n), where n is an integer less than 10.

Q25. What is DROPPING a TABLE ?

Ans. Sometimes there is a requirement to remove a table from the database. In such cases we don't want merely to delete the data from the table, but we want to delete the table itself. DROP TABLE command is used for this purpose. The syntax of DROP TABLE command is as follows:

DROP TABLE <tablename>;

e.g.to remove the table Orders from the database we enter theatement:

DROP TABLE Orders;

And after thisatement orders table is no longer available in the database. It has been removed.Aggregate or Group functions: MySQL provides Aggregate or Group functions which work on a number of values of a column/expression and return a single value as the result.

Important Questions Part-4
More RDBMS(Relational Database Management System)

Q1. What do you mean by DBMS and Transaction Management ?

Ans. Suppose Raunak's account number is 3246 and his aunt's account number is 5135. In order to process the cheque presented by Raunak, the following two SQL commands need to be executed on the database maintained by the bank:
UPDATE Savings SET balance = balance – 2000 WHERE account_no = 5135; For Aunt's account UPDATE Savings SET balance = balance + 2000 WHERE account_no = 3246; For Raunak's account. The above two Updates should both take place. If the first Update takes place and there is a system failure, the first updation should be undone. Either both the updations should be done and if it is notpossible for both the updations to be done, then no updation should be done.

Q2. What is a Transaction?

Ans.A Transaction is a unit of work that must be done in logical order and successfully as a group or not done at all. Unit of work means that a Transaction consists of different tasks - but together they are considered as one unit. Each transaction has a beginning and an end. If anything goes wrong in between the execution of transaction, the entire transaction (No matter to what extent has been done) should be cancelled. If it is successful, then the entire transaction should be saved to the database.
A transaction is a unit of work that must be done in logical order and successfully as a group or not done at all. In Raunak's case, both the updation statements constitute a transaction. Both are together treated as a single unit.

Q3. how transactions are managed ?

Ans : let us study the following 3 statements of SQL:
• START TRANSACTION statement
• COMMIT statement
• ROLLBACK statement
START TRANSACTION Statement :
START TRANSACTION statement commits the current transaction and starts a new

transaction. It tells MySQL that the transaction is beginning and the statements that follow should be treated as a unit, until the transaction ends. It is written like this:

START
TRANSACTION;
The START TRANSACTION statement has no clauses.
COMMIT Statement :
The COMMIT statement is used to save all changes made to the database during the transaction to the database.Commit statement is issued at a time when the transaction is complete- all the changes have been successful and the changes should be saved to the database. COMMIT ends the current
transaction.
COMMIT statement is used like this:
COMMIT;

Or

COMMIT WORK;
Here WORK is a keyword and is optional.
In the following example, the table named savings has 2 rows. A transaction is started and balance in Siddharth's account (with account number 1004) is increased by Rs. 2000.00 and the balance in Akriti's account (with account number 1006) is decreased by Rs. 2000.00. COMMIT statement makes the changes made by the transaction permanent.
Example 1:
mysql> select * from savings;

account_no	Name	balance
1004	Siddharth Sehgal	87000.00
1006	Akriti Malik	87000.00

mysql> START TRANSACTION;
mysql> UPDATE Savings
-> SET balance = balance + 2000
-> WHERE account_no = 1004; mysql>

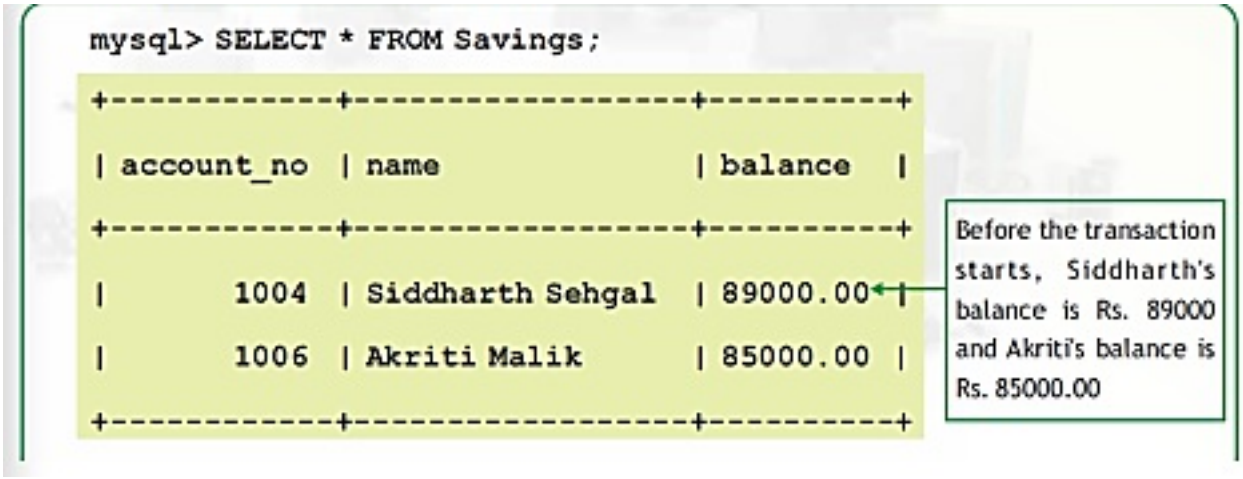
UPDATE Savings
-> SET balance = balance - 2000
-> WHERE account_no = 1006; mysql>
SELECT * FROM Savings;

account_no	Name	Balance
1004	Siddharth Sehgal	89000.00
1006	Akriti Malik	85000.00

2 rows in set (0.00 sec)
mysql> COMMIT;
ROLLBACK Statement :
When a transaction is being executed, some type of error checking is usually performed to check whether it is executing successfully or not. If not, the entire transaction is undone using the ROLLBACK statement. The ROLLBACK statement cancels the entire transaction i.e. It rolls the transaction to the beginning. It aborts any changes made during the transaction and the state of database is returned to what it was before the transaction began to execute and does not save any of the changes made to the database during the transaction.
ROLLBACK statement is used like this:
ROLLBACK;

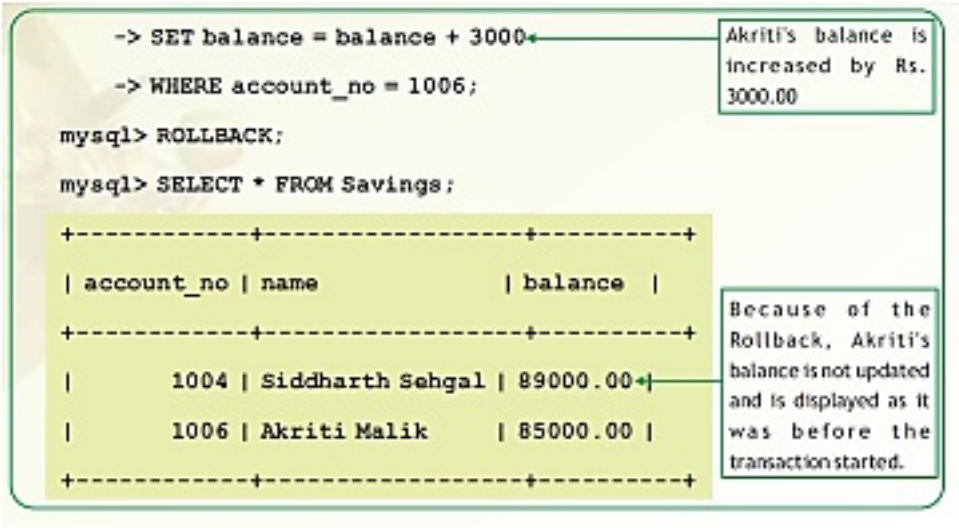
Or

ROLLBACK WORK;
Here WORK is a keyword and is optional.
If in Example 1 shown above ROLLBACK was used instead of COMMIT, the updation of incrementing Siddharth's account by ` 2000.00 and decrementing Akriti's account by 2000 wouldn't have taken place. Let us now initiate a transaction, increase Akriti's account by ` 3000.00, then Rollback the transaction and see what happens to the updation done on Akriti's account.



mysql> START TRANSACTION;

mysql> UPDATE Savings



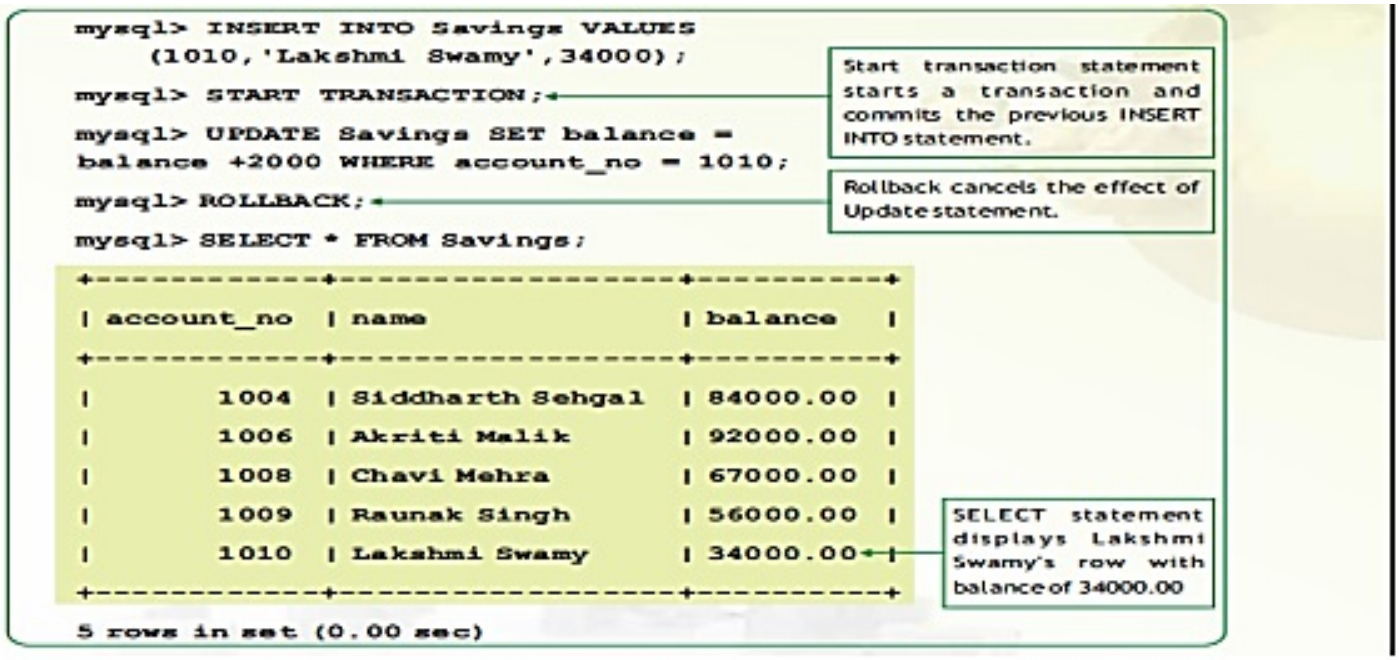
• After the ROLLBACK command is issued to the database, the database itself starts a new transaction; though no explicit command of starting a transaction like START TRANSACTION is issued.

Example 2:

Let us try out some more SQL statements on Savings table to understand transactions well.

mysql> SELECT * FROM savings;

account_no	Name	Balance
1004	Siddharth Sehgal	84000.00
1006	Akriti Malik	92000.00
1008	Chavi Mehra	67000.00
1009	Raunak Singh	56000.00



Q4. What are SavePoints. What is benefit for inserting save points in a transaction ? Give Examples.

Ans The SAVEPOINT statement defines a marker in a transaction. These markers are useful in rolling

back a transaction till the marker.

We can add a savepoint anywhere in a transaction. When you roll back to that savepoint, any

changes made to the database after the savepoint are discarded, and any changes made prior to the savepoint are saved. It is like semicomitting a transaction.

To define a savepoint, we enter the SAVEPOINT statement like this:

SAVEPOINT <savepoint-name>;

Example : SAVEPOINT Mark1;

In the above statement a marker (savepoint) with the name Mark1 is defined. It becomes a bookmark in the transaction. Now we can write the following statement:

Q5. How we can rollback any transaction upto a save point ?

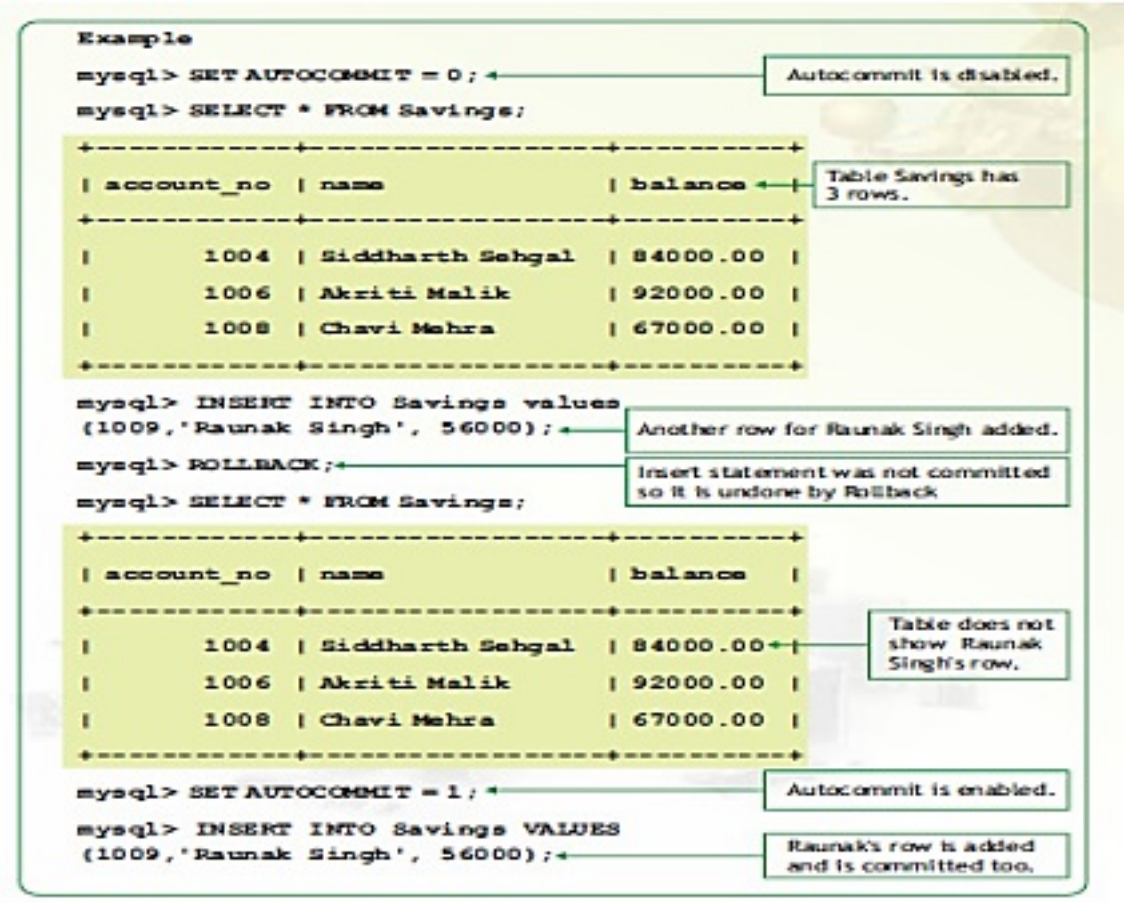
Ans : to rollback the transaction till the bookmark named Mark1.

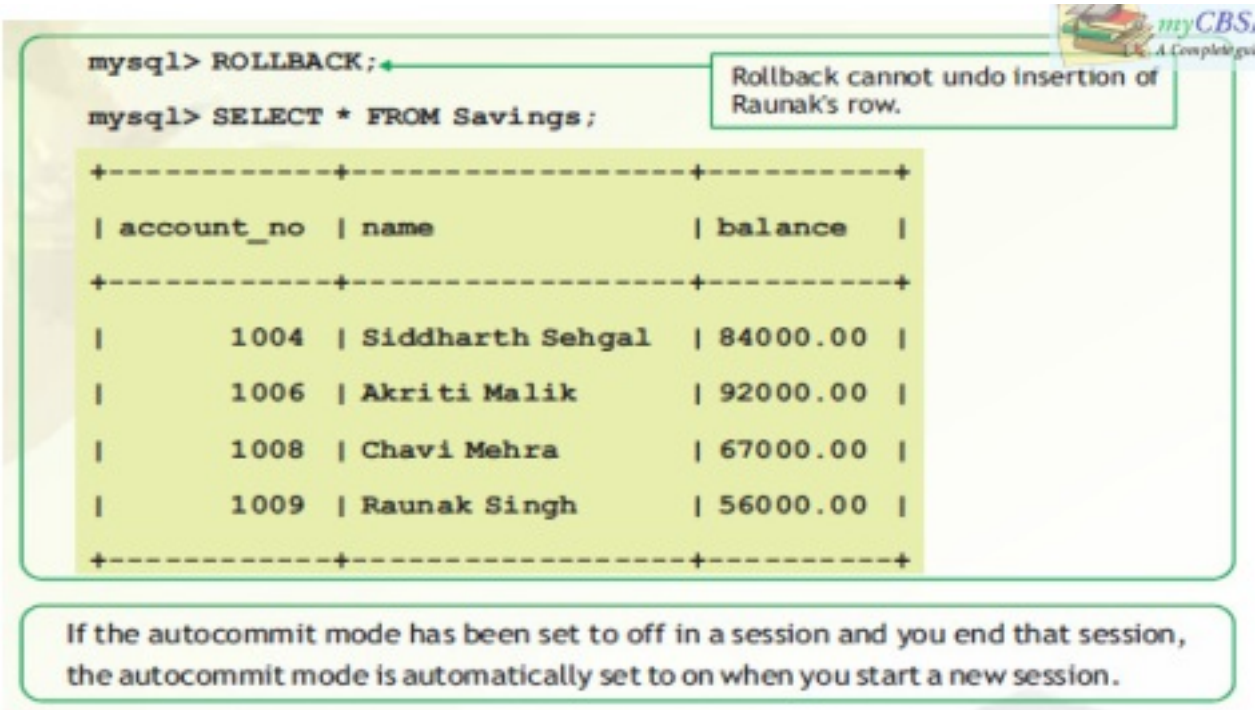
ROLLBACK TO SAVEPOINT Mark1;

Q6. What is Autocommit ?How can it be set ?

Ans : By default, Autocommit mode is on in MySQL. It means that MySQL does a COMMIT

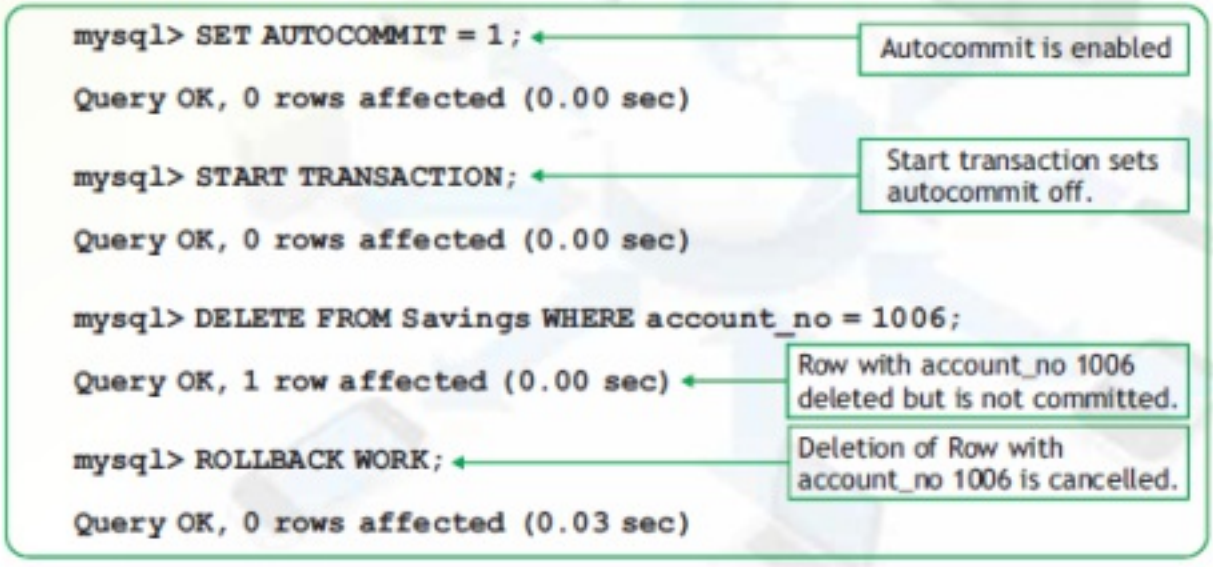
after every SQL statement that does not return an error. If it returns an error then either Rollback or Commit happens depending on the type of error. If we do not want individual statements of SQL to be automatically committed, we should set the autocommit mode to off.When Autocommit is off then
we have to issue COMMIT statement explicitly to save changes made to the database. The following
statement sets the autocommit mode to off. It also starts a new transaction
SET AUTOCOMMIT=0;
The following statement sets the autocommit mode to ON. It also commits and terminates the current transaction.
SET AUTOCOMMIT=1;
If autocommit is set to ON. we can still perform a multiple-statement transaction by starting it with an explicit START TRANSACTION statement and ending it with COMMIT or ROLLBACK.
Let us look at the following example to understand it:





Let us try out some more SQL statements :

Example



mysql> ROLLBACK WORK;

Query OK, 0 rows affected (0.03 sec)

Row with account_no 1006 deleted but is not committed. Deletion of Row with account_no 1006 is cancelled.An implicit COMMIT takes place, even if AUTOCOMMIT is set OFF, on the database when the user issues a Data Definition language command like CREATE TABLE, ALTER TABLE etc

Important Questions Part-5

MySQL

Q.1. What is MySQL?

Ans:- It is an Open Source RDBMS Software. It is available free of cost.

Q.2. What is SQL?

Ans . SQL is Non-procedural universal data access language used to access and manipulate data stored in nearly all the data bases available currently. SQL standards are defined by ANSI (American National Standards Institute). SQL statements are used to retrieve and update data in a database. SQL works with database programs like MySQL, MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc.

Q.3. Differentiate between DDL and DML?

Ans Data Definition Language (DDL): This is a category of SQL commands. All the commands which are used to create, destroy, or restructure databases and tables come under this category. Examples of DDL commands are - CREATE, DROP, ALTER. Data Manipulation Language (DML): This is a category of SQL commands. All the commands which are used to manipulate data within tables come under this category. Examples of DML commands are - INSERT, UPDATE, DELETE.

Q.4 What is a constraint?

Ans : A constraints is a condition or check application on a field or set of fields.

Example: NOT NULL (ensure that column con not have null value), CHECK (make sure that all value satisfy certain criteria), UNIQUE (ensure that all values in a column are different) etc.

Q5 What are single row functions?

Ans: Single Row Function work with a single row at a time. A single row function returns a result for every row of a queried table.

Examples of Single row functions are Sqrt(), Concat(), Lcase(), Upper(), Day(), etc.

Q. 6 Compare CHAR and VARCHAR data types.

Ans. The CHAR data-type stores fixed length strings such that strings having length smaller than the field size are padded on the right with spaces before being stored.

The VARCHAR on the other hand supports variable length strings and therefore stores strings smaller than the field size without modification.

Q.7 What are the differences between DELETE and DROP commands of SQL?

Ans: DELETE is DML command while DROP is a DDL command. Delete is used to delete

rows from a table while DROP is used to remove the entire table from the database.

Q8 What do you understand by MySQL Server?

Ans: MySQL server listens for clients requests coming in over the network and accesses database contents according to those requests and provides that to the client.

Q9 What do you understand by MySQL Client?

Ans: MySQL Clients are programs that connect to MySQL Server and issue queries in predefined format.

Q.10 Explain with the help of an example that why should a transaction be executed as a whole or it should be not executed at all.

Ans: Suppose Raunak's account number is 3246 and his aunt's account number is 5135. In order to process the cheque presented by Raunak, the following two SQL commands need to be executed on the database maintained by the bank:

UPDATE Savings SET balance = balance - 2000

WHERE account_no = 5135;

UPDATE Savings SET balance = balance + 2000

WHERE account_no = 3246;

The above two Updates should both take place. If the first Update takes place and there is

a system failure, the first updation should be undone. Either both the updations should be done and if it is not possible for both the updations to be done, then no updation should be done.

Query Based question & answers

1.The Pincode column of table 'Post' is given below-

.Pincode
10001
120012
300048
281001

i. SELECT Pincode from Post where Pincode LIKE " %1" ;

ii. SELECT Pincode from Post where Pincode LIKE " 0%" ;

Ans: i) 110001 ii) No Output

2. A table "Animals" in a database has 3 columns and 10 records. What is the degree and cardinality of this table?

Ans: Degree 3 and Cardinality=10

3. Answer the question based on the table VOTER given below:

Table: voter

Column Name	Data type	Size	Constraints	Description
V_id	BIGINT	8	Primary key	Voter identification
Vname	VARCHAR	25	Not null	Name of the voter
Age	INT	3	Check>17	Age should not less than equal to 17

Address	VARCHAR2	30		Address of voter
Phone	VARCHAR	10		Phone number of the voter

(i) Write the command to delete all the rows of particular voter from the table voter where voter ID between

10 and 20.

Ans: Delete from VOTER where V_id between 10 and 20;

(ii) Delete the table physically.

Ans: Drop table VOTER;

4. . Write MySql command to create a furniture table including all constraint.

Table: Furniture

ITEMNO	ITEMNAME	TYPE	DATEOFSTOCK	PRICE	DISCOUNT
INT	VARCHAR	VARCHAR		INT	INT
5	20	20	DATE	6	6
PRIMARY KEY	NOT NULL		DEFAULT ‘19/03/2010’		

CREATE TABLE FURNITURE

(ITEMNO INT(5) PRIMARY KEY, ITEMNAME VARCHAR(20) NOT NULL,
TYPE VARCHAR (20),DATE_STOCK DATE DEFAULT '2012/03/19', PRICE INT(6), DISCOUNT INT(2));

5. Consider a database LOANS with the following table:
Table: Loan_Accounts

AccNo	Cust_Name	Loan_Amount	Instalments	Int_Rate	Start_Date
1	R.K. Gupta	300000	36	12.00	19-07-2009
2	S.P. Sharma	500000	48	10.00	22-03-2008
3	K.P. Jain	300000	36	NULL	08-03-2007

4	M.P. Yadav	800000	60	10.00	06-12-2008
5	S.P. Sinha	200000	36	12.50	03-01-2010
6	P. Sharma	700000	60	12.50	05-06-2008
7	K.S. Dhall	500000	48	NULL	05-03-2008

Answer the following questions.

Create Database and use it

1. Create the database LOANS.

Mysql> Create Database LOANS;

2. Use the database LOANS.

Mysql> Use LOANS;

Create Table / Insert Into

3. Create the table Loan_Accounts and insert tuples in it.

Mysql>Create table Loan_Acc (AccNo int primary key,

Cust_Name varchar(30), Loan_Amount int, Installment int, Int_Rate number(5,3),

Start_Date date, Interest number(7,2));

Mysql> Insert into Loan_Acc values(1,'R.K. GUPTA',300000,36,12.0.'2009-07-19');

Simple Select

4. Display the details of all the loans.

Mysql> Select * from Loan_Acc;

5. Display the AccNo, Cust_Name, and Loan_Amount of all the loans.

Mysql> Select Acc_No,Cust_Name,Loan_Amount from Loan_Acc;

Conditional Select using Where Clause

6 Display the details of all the loans with less than 40 instalments.

Mysql> Select * from Loan_Acc where Instalment <40;

7. Display the AccNo and Loan_Amount of all the loans started before 01-04-2009.

Mysql> Select AccNo, Loan_Amount from Loan_Acc where Start_Date <'2009-04-01'; 8. Display the

Int_Rate of all the loans started after 01-04-2009.

Mysql> Select Int_Rate from Loan_Acc where Start_date>'2009-04-01';

Using NULL

8. Display the details of all the loans whose rate of interest is NULL.

Mysql> Select * from Loan_Acc where Int_rate is NULL;

9. Display the details of all the loans whose rate of interest is not NULL.

Mysql> Select * from Loan_Acc where Int_rate is not NULL;

Using DISTINCT Clause

10. Display the amounts of various loans from the table Loan_Accounts. A loan amount should appear only once.

Mysql> Select DISTINCT Loan_Amount from Loan_Acc;

11. Display the number of instalments of various loans from the table Loan_Accounts. An instalment should appear only once..

Mysql> Select DISTINCT Instalment from Loan_Acc;

Using Logical Operators (NOT, AND, OR)

12. Display the details of all the loans started after 31-12-2008 for which the number of instalments are more than 36.

Mysql> Select * from Loan_Acc where Start_Date>'2008-12-31' and Instalment>36;

13. Display the Cust_Name and Loan_Amount for all the loans which do not have number of instalments 36.

Mysql> Select Cust_Name, Loan_Amount from Loan_Acc where Instalment <>36;

14. Display the Cust_Name and Loan_Amount for all the loans for which the loan amount is less than 500000 or int_rate is more than 12.

Mysql> Select Cust_Name, Loan_Amount from Loan_Acc where Loan_Amount <500000 or Int_rate>12;

15. Display the details of all the loans which started in the year 2009.

Mysql> Select * from Loan_Acc where Year(Start_Date)=2009;

16. Display the details of all the loans whose Loan_Amount is in the range 400000 to 500000.

Mysql> Select * from Loan_Acc where Loan Amount between 400000 and 50000;

17. Display the details of all the loans whose rate of interest is in the range 11% to 12%.

Mysql> Select * from Loan_Acc where Int_Rate between 11 and 12;

Using IN Operator

19. Display the Cust_Name and Loan_Amount for all the loans for which the number of instalments are 24, 36, or 48. (Using IN operator)

Mysql> Select Cust_Name, Loan_Amount from Loan_Acc where Instalment IN(24,36,48); UR

Using LIKE Operator

20. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the Cust_Name ends with 'Sharma'.

Mysql> Select AccNo, Cust_name from Loan_Acc where

Cust_Name like '%Sharma';

21. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the Cust_Name ends with 'a'.

Mysql> Select AccNo, Cust_name, Loan_Amount from Loan_Acc where Cust_Name like '%a';

22. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the

Cust_Name contains 'a'

Mysql> Select AccNo, Cust_name,Loan_Amount from Loan_Acc where

Cust_Name like '%a%';

23. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the

Cust_Name does not contain 'P'.

Mysql> Select AccNo, Cust_name,Loan_Amount from Loan_Acc where

NOT (Cust_Name like '%P%');

24. Display the AccNo, Cust_Name, and Loan_Amount for all the loans for which the

Cust_Name contains 'a' as the second last character.

Mysql> Select AccNo, Cust_name,Loan_Amount from Loan_Acc where

Cust_Name like '%a_';

Using ORDER BY clause

25. Display the details of all the loans in the ascending order of their Loan_Amount.

Mysql> Select * from Loan_Acc ORDER BY Loan_Amount;

28. Display the details of all the loans in the descending order of their Start_Date.

Mysql> Select * from Loan_Acc ORDER BY Start_date DESC;

29. Display the details of all the loans in the ascending order of their Loan_Amount and within Loan_Amount in the descending order of their Start_Date.

Mysql> Select * from Loan_Acc ORDER BY Loan_Amount, Start_Date DESC;

Using UPDATE, DELETE, ALTER TABLE

30. Put the interest rate 11.50% for all the loans for which interest rate is NULL.
Mysql> Update Loan_Acc SET Int_Rate =11.50 Where Int_Rate IS NULL:

31. Increase the interest rate by 0.5% for all the loans for which the loan amount is more than 400000.

Mysql> Update Loan_Acc SET Int_Rate= Int_Rate+0.5

Where Loan_Amount >400000;

32. For each loan replace Interest with (Loan_Amount*Int_Rate*Instalments) 12*100.
Mysql> Update Loan_Acc
SET Interest=(Loan_Amount*Int_Rate*Instalments) /12*100;

33. Delete the records of all the loans whose start date is before 2007.
Mysql> Delete From Loan_Acc Where Year(Start_Date)<2007;

34. Delete the records of all the loans of 'K.P. Jain'
Mysql> Delete From Loan_Acc Where Cust_Name='K.P.Jain';

35. Add another column Category of type CHAR(1) in the Loan table.

Mysql> Alter Table Loan_Acc ADD (Category CHAR(1));

Find the Output of the following queries

36.SELECT cust_name, LENGTH(Cust_Name), LCASE(Cust_Name), UCASE(Cust_Name) FROM Loan_Accounts WHERE Int_Rate < 11.00;

Cust_Name	Length(Cust_Name)	LCASE(Cust_Name)	UCASE(Cust_Name)
S.P. Sharma	11	s.p. sharma	S.P. SHARMA
M.P. Yadav	10	m.p. yadav	M.P. YADAV

37.SELECT LEFT(Cust_Name, 3), Right(Cust_Name, 3), SUBSTR(Cust_Name, 1, 3) FROM

Loan_Accounts WHERE Int_Rate > 10.00;

LEFT(Cust_Name,3)	RIGHT(Cust_Name,3)	SUBSTR(Cust_Name,1,3)
R.K	Pta	R.K
S.P	Nha	S.P
P.	Rma	P.

SELECT RIGHT(Cust_Name, 3), SUBSTR(Cust_Name, 5) FROM Loan_Accounts;

RIGHT(Cust_Name, 3)	SUBSTR(Cust_Name, 5)
pta	Gupta
rma	Sharma
ain	Jain
dav	Yadav
nha	Sinha
rma	harma
all	Dhal

39. SELECT DAYOFMONTH(Start_Date) FROM Loan_Accounts;

DAYOFMONTH(Start_Date)
19
22
08
06
03
05
05