MORE ON SQL

Union operator in select command

To combine the result of two or more select statements union operator is used.

The number of columns selected and their data type from each table should be the same.

```
mysql> select * from employee;
+-------+--------+--------+
| empid | name   | salary |
+-------+--------+--------+
|  1001 | Rajesh |  45000 |
|  1002 | Rakesh |  34500 |
|  1003 | Yuva   |  56000 |
+-------+--------+--------+
```

```
mysql> select * from department;
+--------+--------+-------------------+
| deptid | emp_id | years_of_service  |
+--------+--------+-------------------+
|    101 |   1001 |                12 |
|    102 |   1002 |                 8 |
|    103 |   1003 |                 9 |
+--------+--------+-------------------+
```

```
mysql> select name,salary from employee where salary<50000 union
    -> select deptid,years_of_service from department
    -> where years_of_service>8;
+--------+--------+
| name   | salary |
+--------+--------+
| Rajesh |  45000 |
| Rakesh |  34500 |
| 101    |     12 |
| 103    |      9 |
+--------+--------+
```

Drop database command

Drop table <table_name> is used to drop or delete the table, whereas Drop Database command removes a database along with all its tables.

Syntax :

DROP DATABASE <database_name>;

Example:

DROP DATABASE XIF;

Adding NULL values in a table using insert command

Using Insert command we can add NULL values to any column other than being assigned as PRIMARY KEY or NOT NULL constraints.

When actual values are not defined it is advisable to set NULL values which are treated differently as they represent missing unknown data. NULL is not equivalent to 0, it acts a placeholder for unknown or inapplicable values.

```
mysql> desc department;
+------------------+---------+------+-----+---------+-------+
| Field            | Type    | Null | Key | Default | Extra |
+------------------+---------+------+-----+---------+-------+
| deptid           | int(11) | NO   | PRI | NULL    |       |
| emp_id           | int(11) | YES  |     | NULL    |       |
| years_of_service | int(11) | YES  |     | NULL    |       |
+------------------+---------+------+-----+---------+-------+
```

```
mysql> insert into department
    -> values(NULL,null,Null);
ERROR 1048 (23000): Column 'deptid' cannot be null
```

Adding NULL values in a table using insert command

```
mysql> desc department;
+------------------+---------+------+-----+---------+-------+
| Field            | Type    | Null | Key | Default | Extra |
+------------------+---------+------+-----+---------+-------+
| deptid           | int(11) | NO   | PRI | NULL    |       |
| emp_id           | int(11) | YES  |     | NULL    |       |
| years_of_service | int(11) | YES  |     | NULL    |       |
+------------------+---------+------+-----+---------+-------+

mysql> insert into department
    -> values(108,NULL,NULL);
Query OK, 1 row affected (0.00 sec)

mysql> select * from department;
+----------+----------+------------------+
| deptid   | emp_id   | years_of_service |
+----------+----------+------------------+
|      101 |     1001 |               12 |
|      102 |     1002 |                8 |
|      103 |     1003 |                9 |
|      108 |     NULL |             NULL |
+----------+----------+------------------+
```

Update command on multiple columns :

To modify the values in more than one column in the update command one can use more than one column to be updated by separating them by comma in SET clause.

```
mysql> select * from department;
+--------+--------+--------------------+
| deptid | emp_id | years_of_service   |
+--------+--------+--------------------+
|    101 |   1001 |                 12 |
|    102 |   1002 |                  8 |
|    103 |   1003 |                  9 |
|    108 |   NULL |               NULL |
+--------+--------+--------------------+
```

```
mysql> select * from department;
+--------+--------+--------------------+
| deptid | emp_id | years_of_service   |
+--------+--------+--------------------+
|    101 |   1001 |                 12 |
|    102 |    102 |                  8 |
|    103 |    103 |                  9 |
|    108 |   1008 |                 10 |
+--------+--------+--------------------+
```

```
mysql> update department
    -> set emp_id=1008,years_of_service=10 where deptid=108;
```

Update command on multiple columns :

One could also set NULL value to column (provided it is not with Primary key, NOT NULL constraint)

```
mysql> update department
    -> set emp_id=null where deptid=108;
```

```
ysql> select * from department;
+--------+--------+--------------------+
| deptid | emp_id | years_of_service   |
+--------+--------+--------------------+
|    101 |   1001 |                 12 |
|    102 |    102 |                  8 |
|    103 |    103 |                  9 |
|    108 |   NULL |                 10 |
+--------+--------+--------------------+
```

Update command on multiple columns :

Example : Updating column(s) in SQL using an expression or formula

```
mysql> update employee
    -> set salary=salary+salary*10/100
    -> where name like 'R%';
```

```
mysql> select * from employee;
+--------+---------+---------+
| empid  | name    | salary  |
+--------+---------+---------+
|   1001 | Rajesh  |   49500 |
|   1002 | Rakesh  |   37950 |
|   1003 | Yuva    |   56000 |
+--------+---------+---------+
```

Truncate statement in SQL

Truncate command is used to delete all the rows from the table and free the space containing the table.

Syntax :

Truncate table <table_name>;

```
mysql> truncate table department;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from department;
Empty set (0.00 sec)
```

Delete and Truncate statements - difference

Delete statement	Truncate statement
Deletes the row(s) in the table based on condition or even without specifying the condition the row(s) can be deleted. This statement does not free space containing the table.	Truncate delete all rows from the table and free the space containing the table.

Alter command with drop

To remove a column in table, drop keyword of Alter command can be used:

Syntax :

Alter table <table-name> DROP <column_name>;

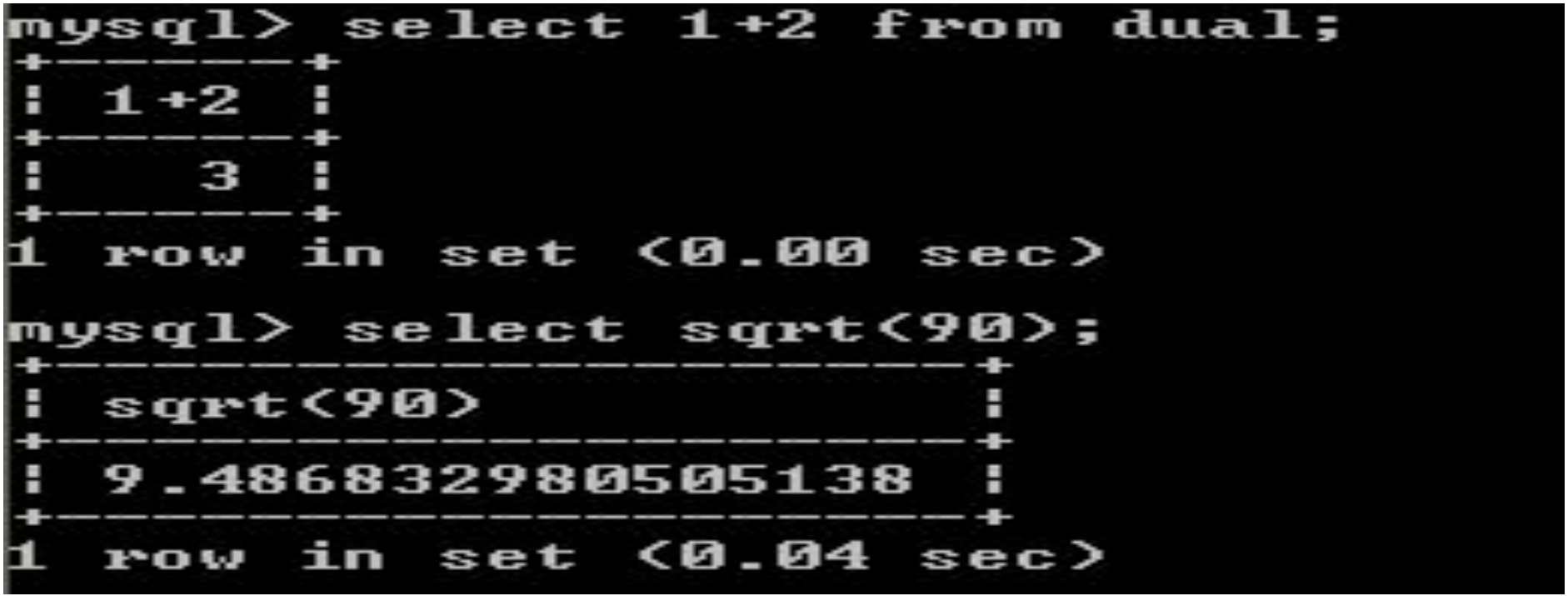
```
mysql> desc employee;
+--------+-----------+------+-----+---------+-------+
| Field  | Type      | Null | Key | Default | Extra |
+--------+-----------+------+-----+---------+-------+
| empid  | int(11)   | NO   | PRI | NULL    |       |
| name   | char(25)  | YES  |     | NULL    |       |
| salary | float     | YES  |     | NULL    |       |
+--------+-----------+------+-----+---------+-------+

mysql> alter table employee
    -> drop salary;

mysql> desc employee;
+--------+-----------+------+-----+---------+-------+
| Field  | Type      | Null | Key | Default | Extra |
+--------+-----------+------+-----+---------+-------+
| empid  | int(11)   | NO   | PRI | NULL    |       |
| name   | char(25)  | YES  |     | NULL    |       |
+--------+-----------+------+-----+---------+-------+
```

Dual table

MySQL has a default table called DUAL, it is one-row,one-column dummy table which can be used for any arithmetic or relational or logical or even date calculations without creating any table structure.



Select statement with scalar expression

Select statement permits calculations on the column contents and display the result without affecting or changing the data in the table content. We can use scalar expression and constant values for the selected columns. If the column(s) has NULL values then the expression will return NULL.

```
mysql> select empid,name,salary+1000 from employee;
+--------+---------+----------------+
| empid  | name    | salary+1000    |
+--------+---------+----------------+
|   1001 | Rajesh  |          50500 |
|   1002 | Rakesh  |          38950 |
|   1003 | Yuva    |          57000 |
+--------+---------+----------------+
3 rows in set (0.00 sec)

mysql> select * from employee;
+--------+---------+----------+
| empid  | name    | salary   |
+--------+---------+----------+
|   1001 | Rajesh  |   49500  |
|   1002 | Rakesh  |   37950  |
|   1003 | Yuva    |   56000  |
+--------+---------+----------+
3 rows in set (0.00 sec)
```

Comments in SQL

For single line or inline comment begin with -- the text after the -- cannot exceed to a new line and ends with a line break.

Alternatively # can also be used to annotate user's information.

For multi-line comments /* comment */ can be used

```
mysql> select * from employee;  -- to display all records
+-------+---------+
| empid | name    |
+-------+---------+
|  1001 | Rajesh  |
|  1002 | Rakesh  |
|  1003 | Yuva    |
+-------+---------+
3 rows in set (0.00 sec)

mysql> select * from employee;  # to display all records
+-------+---------+
| empid | name    |
+-------+---------+
|  1001 | Rajesh  |
|  1002 | Rakesh  |
|  1003 | Yuva    |
+-------+---------+

mysql> /* select command is used to display
    /*> multiple rows
    /*> also to retrieve rows based on condition */
mysql>
```

Exercises

SQL - QUERIES

Unsolved Questions :

13. Write SQL queries to perform the following based on the table PRODUCT having fields as (prod_id, prod_name, quantity, unit_rate, price, city)

(i) Display those records from table PRODUCT where prod_id is more than 100.

(ii) List records from table PRODUCT where prod_name is 'Almirah'.

(iii) List all those records whose price is between 200 and 500.

(iv) Display the product names whose price is less than the average of price.

(v) Show the total number of records in the table PRODUCT.

13. Solution

1. Select * from product where prod_id>100;
2. Select * from product where prod_name='Almirah';
3. Select * from product where price between 200 and 500;
4. Select Price from product where price < (Select avg(price) from product);
5. Select count(*) 'Total Records' from product;

15. Consider the following tables STORE and SUPPLIERS. Write SQL commands for the statements (i) to (iv) and give outputs for SQL queries (v) to (viii).

Table: Store

ItemNo	Item	Scode	Qty	Rate	LastBuy
2005	Sharpener Classic	23	60	8	31-Jun-09
2003	Ball Pen 0.25	22	50	25	01-Feb-10
2002	Gel Pen Premium	21	150	12	24-Feb-10
2006	Gel Pen Classic	21	250	20	11-Mar-09
2001	Eraser Small	22	220	6	19-Jan-09
2004	Eraser Big	22	110	8	02-Dec-09
2009	Ball Pen 0.5	21	180	18	03-Nov-09

Table: SUPPLIERS

Scode	Sname
21	Premium Stationery
23	Soft Plastics
22	Tetra Supply

(i) To display details of all the items in the Store table in ascending order of LastBuy.

(ii) To display Itemno and item name of those items from Store table whose rate is more than 15 rupees.

(iii) To display the details of those items whose supplier code is 22 or quantity in store is more than 110 from the table Store.

(iv) To display minimum rate of items for each Supplier individually as per Scode from the table Store.

(v) SELECT COUNT(DISTINCT Scode) FROM STORE;

(vi) SELECT Rate*Qty FROM STORE WHERE Itemno=2004;

(vii) SELECT Item, Sname FROM STORE S, SUPPLIER P WHERE S.Scode=P.Scode AND ItemNo=2006;

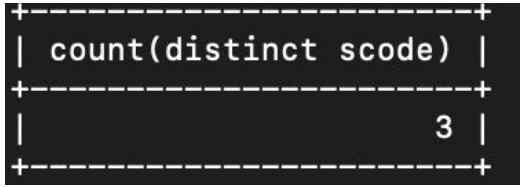
(viii) SELECT MAX(LastBuy)FROM STORE;

Solution : 15

1) Select * from store order by lastbuy;
2) Select itemname,itemno from store where rate > 15;
3) select * from store where scode=22 or qty>110;
4) Select scode,min(rate) ‘Minimum Rate’ from store s,suppliers p where s.scode = p.scode group by s.scode;

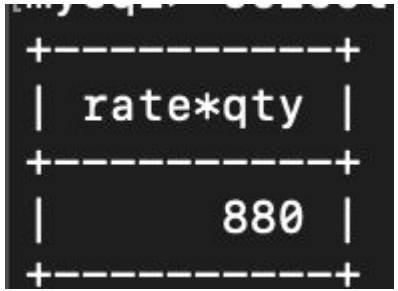
Solution : 15

5)



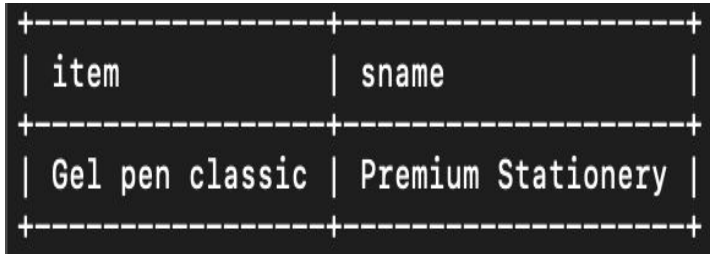
count(distinct scode)
3

6)



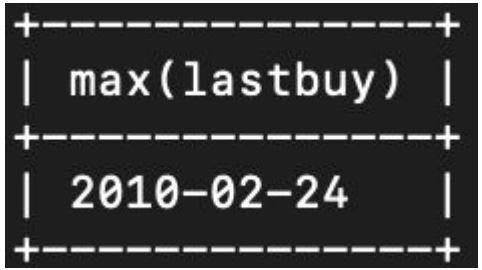
rate*qty
880

7)



item	sname
Gel pen classic	Premium Stationery

8)



max(lastbuy)
2010-02-24

16. Write SQL commands for (i) to (vi) on the basis of relations given below:

BOOKS

book_id	Book_name	author_name	Publishers	Price	Type	qty
k0001	Let us C	Sanjay Mukharjee	EPB	450	Comp	15
p0001	Genuine	J. Mukhi	FIRST PUBL.	755	Fiction	24
m0001	Mastering C++	Kantkar	EPB	165	Comp	60
n0002	VC++ advance	P. Purohit	TDH	250	Comp	45
k0002	Programming with Python	Sanjeev	FIRST PUBL.	350	Fiction	30

ISSUED

Book_ID	Qty_Issued
L02	13
L04	5
L05	21

(i) To show the books of FIRST PUBL. Publishers written by P. Purohit.
(ii) To display cost of all the books published for FIRST PUBL.
(iii) Depreciate the price of all books of EPB publishers by 5%.
(iv) To display the BOOK_NAME and price of the books, more than 3 copies of which have been issued.
(v) To show total cost of books of each type.
(vi) To show the details of the costliest book.

16. Solution

1. Select * from BOOKS where Publishers=’FIRST PUBl.’ and author_name=’P.Purohit’;
2. Select sum(Price) from BOOKS where Publishers=’FIRST PUBL.’;
3. Update Books set Price = Price -(Price*5/100) where Publishers=’EPB’;
4. Select Book_name,price,book_id, from BOOKS,ISSUED WHERE books.book_id=issued.book_id and qty_issued>3;
5. Select sum(Price*qty) ‘Amount’ from Books group by type;
6. Select book_name,max(Price) from Books;

17. Write SQL commands for (i) to (vi) and write output for (vii) on the basis of PRODUCTS relation given below:

PRODUCTS TABLE

PCODE	PNAME	COMPANY	PRICE	STOCK	MANUFACTURE	WARRANTY
P001	TV	BPL	10000	200	12-JAN-2018	3
P002	TV	SONY	12000	150	23-MAR-2017	4
P003	PC	LENOVO	39000	100	09-APR-2018	2
P004	PC	COMPAQ	38000	120	20-JUN-2019	2
P005	HANDYCAM	SONY	18000	250	23-MAR-2017	3

(i) To show details of all PCs with stock more than 110.

(ii) To list the company which gives warranty of more than 2 years.

(iii) To find stock value of the BPL company where stock value is the sum of the products of price and stock.

(iv) To show number of products from each company.

(v) To count the number of PRODUCTS which shall be out of warranty on 20-NOV-2020.

(vi) To show the PRODUCT name of the products which are within warranty as on date.

(vii) Give the output of the following statements:

(a) Select COUNT(distinct company) from PRODUCT;

(b) Select MAX(price)from PRODUCT where WARRANTY<=3;

Solution : 17

i) select * from Products where pname like ‘PC’ and stock>110;

ii) select company from products where warranty >2;

iii) select sum(price*stock) ‘Total’ from product where company=’BPL’;

iv) select count(Pcode) ‘Product count’ from products group by company;

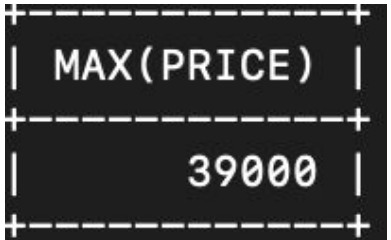
v) select count(Pcode) ‘Product’ from product where year(‘20-NOV-2020’)-YEAR(manufacture) >warranty;

vi) select pname from products where year(sysdate())-year(manufacture)>warranty;

vii) a)

COUNT(DISTINCT COMPANY)
4

b)



MAX(PRICE)
39000

22. Consider the given table and answer the questions.

Table: SchoolBus

Rtno	Area_Covered	Capacity	Noofstudents	Distance	Transporter	Charges
1	Vasant Kunj	100	120	10	Shivam travels	100000
2	Hauz Khas	80	80	10	Anand travels	85000
3	Pitampura	60	55	30	Anand travels	60000
4	Rohini	100	90	35	Anand travels	100000
5	Yamuna Vihar	50	60	20	Bhalla travels	55000
6	Krishna Nagar	70	80	30	Yadav travels	80000
7	Vasundhara	100	110	20	Yadav travels	100000
8	Paschim Vihar	40	40	20	Speed travels	55000
9	Saket	120	120	10	Speed travels	100000
10	Janakpuri	100	100	20	Kisan Tours	95000

(i) To show all information of students where capacity is more than the no. of students in order of rtno.

(ii) To show area_covered for buses covering more than 20 km., but charges less than 80000.

(iii) To show transporter-wise total no. of students travelling.

(iv) To show rtno, area_covered and average cost per student for all routes where average cost per student is—charges/noofstudents.

(v) Add a new record with the following data:
(11, "Motibagh",35,32,10, "kisan tours", 35000)

(vi) Give the output considering the original relation as given:

(a) select sum(distance) from schoolbus where transporter= "Yadav travels";

(b) select min(noofstudents) from schoolbus;

(c) select avg(charges) from schoolbus where transporter= "Anand travels";

(d) select distinct transporter from schoolbus;

Solution :22

1) Select * from schoolbus where noofstudent > capacity order by rtno;
2) Select area_covered from schoolbus where distance > 20 and charges <80000;
3) Select Transporter,sum(noofstudent) ‘Student Count’ from schoolbus group by tranporter;
4) Select rtno,area_covered,round(avg(charges/noofstudent),2) ‘Travel charge per student’ from schoolbus group by area_covered order by rtno;
5) Insert into schoolbus values (11,’Motibagh’,35,32,10,’Kisan Tours’,35000);
6)

sum(distance)
50

7)

min(noofstudent)
32

8)

avg(charges)
81666.66666666667

9)

transporter
Shivam Travels
Anand Travels
Bhalla Travels
Yadav Travels
Speed Travels
Kisan Tours

23. Write SQL Commands for (i) to (v) and write the outputs for (vi) to (viii) on the basis of the following table:

Table: FURNITURE

NO	ITEM	TYPE	DATEOFSTOCK	PRICE	DICOUNT
1	WhiteLotus	DoubleBed	2002-02-23	3000	25
2	Pinkfeathers	BabyCot	2002-01-29	7000	20
3	Dolphin	BabyCot	2002-02-19	9500	20
4	Decent	OfficeTable	2002-02-01	25000	30
5	Comfortzone	DoubleBed	2002-02-12	25000	30
6	Donald	BabyCot	2002-02-24	6500	15

(i) To list the details of furniture whose price is more than 10000.

(ii) To list the Item name and Price of furniture whose discount is between 10 and 20.

(iii) To delete the record of all items where discount is 30.

(iv) To display the price of 'Babycot'.

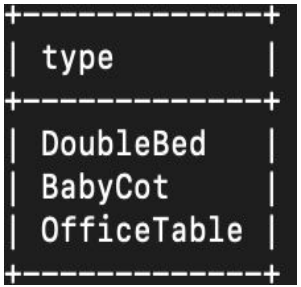
(v) To list item name, type and price of all items whose names start with 'D'.

(vi) Select Distinct Type from Furniture;

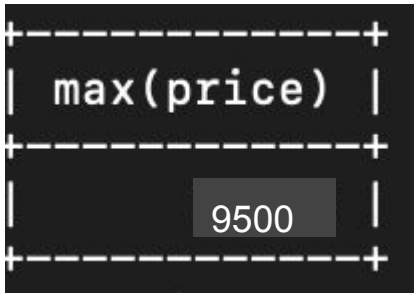
(vii) Select Max(Price) from Furniture where DateofStock>'2002-02-15';

(viii) Select Count(*) from Furniture where Discount<25;

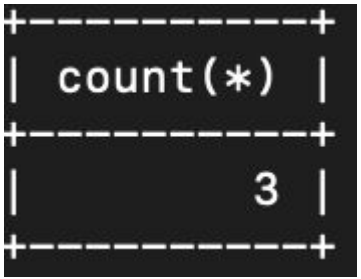
Solution : 23

1) Select * from furniture where price > 10000;
2) Select Item,Price from Furniture where discount between 10 and 20;
3) Delete * from Furniture where discount = 30;
4) Select Price from Furniture where Type = ‘BabyCot’;
5) Select Item,Type,Price from Furniture where Item like ‘D%’;
6) 

type
DoubleBed
BabyCot
OfficeTable

7) 

max(price)
9500

8) 

count(*)
3

	Function	Description	Examples
1.	CHAR( )	Returns the character for each integer passed	1. SELECT CHAR (70, 65, 67, 69) ; 2. SELECT CHAR (65, 67.3, '69.3) ;
2.	CONCAT( )	Returns concatenated string	SELECT CONCAT(name, aggregate) AS "Name Marks" FROM student WHERE age = 14 OR age = 16 ;
3.	LOWER( ) / LCASE( )	Returns the argument in lowercase	SELECT LOWER('MR. OBAMA') AS "LowerName1", LOWER('Ms. Gandhi') AS "LowerName2" ;
4.	SUBSTRING( ), SUBSTR()	Returns the substring as specified	1. SELECT SUBSTR('ABCDEFG', 3, 4) "Subs" ; 2. SELECT SUBSTR ('ABCDEFG', -5, 4) "Subs"
.	UPPER()/UCASE( )	Converts to uppercase	SELECT UPPER('Large') "Uppercase" ; or SELECT UCASE('Large') "Uppercase" ;
	TRIM( )	Removes leading and trailing spaces	SELECT TRIM(' Bar One ') ;
	LENGTH( )	Returns the length of a string in bytes	SELECT LENGTH('CANDIDE') "Length in characters" ;

required operation, return numeric values.

	Function	Description	Example
1.	MOD()	Returns the remainder of one expression by diving by another expression.	SELECT MOD(11, 4) "Modulus" ;
2.	POWER()/POW()	Returns the value of one expression raised to the power of another expression	SELECT POWER(3, 2) "Raised" ;
3.	ROUND()	Returns numeric expression rounded to an integer. Can be used to round an expression to a number of decimal points	SELECT ROUND(15.193, 1) "Round" ;
4.	SIGN()	This function returns sign of a given number	SELECT SIGN(-15) "Sign" ;
5.	SQRT()	Returns the non-negative square root of numeric expression.	SELECT SQRT(26) "Square root" ;
6.	TRUNCATE()	Returns numeric exp1 truncated to exp2 decimal places. If exp2 is 0, then the result will have no decimal point.	SELECT TRUNCATE(15.79, 1) "Truncate"

16.6.3 Date and Time Functions

Date functions operate on values of the DATE datatype.

	Function	Description	Example
1.	CURDATE( ) / CURRENT_DATE( ) / CURRENT_DATE	Returns the current date	SELECT CURDATE( ) ;
2.	DATE( )	Extracts the date part of a date or date-time expression	SELECT DATE('2020-12-31 01:02:03') ;
3.	MONTH( )	Returns the month from the date passed	SELECT MONTH('2020-02-03') ;
4.	YEAR( )	Returns the year	SELECT YEAR('2020-02-03') ;
5.	NOW( )	Returns the time at which the function executes	SELECT NOW( ) ;
6.	SYSDATE( )	Returns the current date and time	SELECT NOW(), SLEEP(2), NOW();

24. Write the outputs of the following:

(i) Select POWER(2,5);

(ii) Select ROUND(10.195,2);

(iii) Select TRUNCATE(10.195,2);

(iv) Select SUBSTR('COMPUTER SCIENCE',3,4);

(v) Select INSTR('Technical Teaching','ch');

24) Solution

i)

power(2,5)
32

ii)

round(10.195,2)
10.20

iii)

truncate(10.195,2)
10.19

iv)

substr('COMPUTER SCIENCE',3,4)
MPUT

v)

INSTR('Technical Teaching','ch')
3

25. Write SQL Commands for the following on the basis of the given table GRADUATE:

Table: GRADUATE

SL.No.	NAME	STIPEND	SUBJECT	AVERAGE	RANK
1	KARAN	400	PHYSICS	68	1
2	RAJ	450	CHEMISTRY	68	1
3	DEEP	300	MATHS	62	2
4	DIVYA	350	CHEMISTRY	63	1
5	GAURAV	500	PHYSICS	70	1
6	MANAV	400	CHEMISTRY	55	2
7	VARUN	250	MATHS	64	1
8	LIZA	450	COMPUTER	68	1
9	PUJA	500	PHYSICS	62	1
10	NISHA	300	COMPUTER	57	2

(i) List the names of those students who have obtained rank 1 sorted by NAME.

(ii) Display a list of all those names whose AVERAGE is greater than 65.

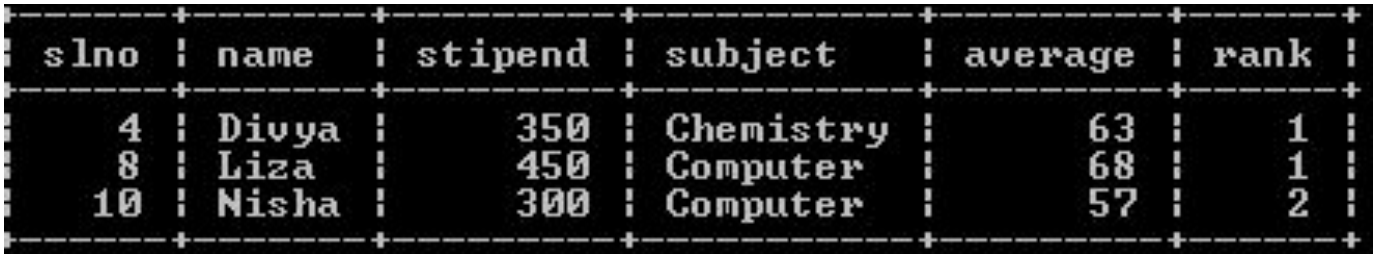
(iii) Display the names of those students who have opted COMPUTER as a SUBJECT with an AVERAGE of more than 60.

(iv) List the names of all the students in alphabetical order.

(v) SELECT * FROM GRADUATE WHERE NAME LIKE "% I %";

(vi) SELECT DISTINCT RANK FROM GRADUATE;

Solution: 25

1) Select name from Graduate where rank=1 order by name;
2) Select name from Graduate where average > 65;
3) Select name from Graduate where Subject like ‘CO%’ and average >60;
4) Select name from Graduate order by name;
5) 

slno	name	stipend	subject	average	rank
4	Divya	350	Chemistry	63	1
8	Liza	450	Computer	68	1
10	Nisha	300	Computer	57	2

6) 

rank
1
2

(c) For the given table, do as directed:

Table: STUDENT

Column name	Data type	size	Constraint
ROLLNO	Integer	4	Primary Key
SNAME	Varchar	25	Not Null
GENDER	Char	1	Not Null
DOB	Date		Not Null
FEES	Integer	4	Not Null
HOBBY	Varchar	15	Null

(i) Write SQL query to create the table.

(ii) Write SQL query to increase the size of SNAME to hold 30 characters.

(iii) Write SQL query to remove the column HOBBY.

(iv) Write SQL query to insert a row in the table with any values of your choice that can be accommodated there.

Solution c)

i) create table student (rollno integer(4) primary key,

Sname varchar(25) not null,gender char(1) not null,dob date not null,fees integer(4) not null,hobby varchar(15) null);

2) alter table student modify sname varchar(30);

3) alter table student drop hobby;

4) insert into student values(1,’Syed’,’M’,’2000-01-01’,12000);

27. Write SQL queries based on the following tables:

PRODUCT:

P_ID	ProductName	Manufacturer	Price	Discount
TP01	Talcum Powder	LAK	40	
FW05	Face Wash	ABC	45	5
BS01	Bath Soap	ABC	55	
SH06	Shampoo	XYZ	120	10
FW12	Face Wash	XYZ	95	

CLIENT:

C_ID	ClientName	City	P_ID
01	Cosmetic Shop	Delhi	TP01
02	Total Health	Mumbai	FW05
03	Live Life	Delhi	BS01
04	Pretty Woman	Delhi	SH06
05	Dreams	Delhi	FW12

(i) Write SQL Query to display ProductName and Price for all products whose Price is in the range 50 to 150.

(ii) Write SQL Query to display details of products whose manufacturer is either XYZ or ABC.

(iii) Write SQL query to display ProductName, Manufacturer and Price for all products that are not giving any discount.

(iv) Write SQL query to display ProductName and price for all products whose ProductName ends with 'h'.

(v) Write SQL query to display ClientName, City, P_ID and ProductName for all clients whose city is Delhi.

(vi) Which column is used as Foreign Key and name the table where it has been used as Foreign key.

Solution : 27

1) Select ProductName,Price From Product where price between 50 and 150;
2) Select * from Product where manufacturer in (‘XYZ’,’ABC’);
3) Select ProductName,Manufacturer,Price from Product where
   Discount is NULL;
4) Select ProductName,Price from Product where ProductName like ‘%h’;
5) Select ClientName,City,Product.P_ID,ProductName from Product,Client
   Where Product.P_ID = Client.P_ID and city like ‘Delhi’;
   (or)
   Select ClientName,City,P.P_ID,ProductName from Product P,Client C
   Where P.P_ID = C.P_ID and city like ‘Delhi’;

6) P_ID in Client table is used as foreign key as it acts as reference between
   Product and Client Table.

28. Answer the question based on the table given below:

Table: HOSPITAL

SL.No.	Name	Age	Department	Dateofadm	Charges	Sex
1	Arpit	62	Surgery	21-01-98	300	M
2	Zareena	22	ENT	12-12-97	250	F
3	Kareem	32	Orthopedic	19-02-98	200	M
4	Arun	12	Surgery	11-01-98	300	M
5	Zubin	30	ENT	12-01-98	250	M

(a) To list the names all the patients admitted after 15/01/98.
(b) To list the names of female patients who are in ENT department.
(c) To list the names of all patients with their date of admission in ascending order.
(d) To display Patient's Name, Charges, Age for only female patients.
(e) Find out the output of the following SQL commands:-
  (i) Select COUNT(DISTINCT charges) from HOSPITAL;
  (ii) Select MIN(Age) from HOSPITAL where Sex="F";

Solution:28

1) Select Name from Hospital where Dateofadm >’1998-01-15’;
2) Select Name from Hospital where Sex = ‘F’ and Department=’ENT’;
3) Select Name from Hospital order by Dateofadm;
4) Select Name,Charges,Age from Hospital where Sex = ‘F’;
5) i)

```
----------------------
DISTINCT charges
----------------------
250
300
----------------------
```

ii)

```
-----------------
MIN(Age)
-----------------
22
-----------------
```

Joins - examples

Join clause combines rows in two or more tables.

Equi - join

Id	Name	Salary	desig	dept
101	Sri	45000	Manager	10
102	Helen	55000	Manager	10
103	Karan	85000	President	20
104	Yogesh	78000	President	10
105	Krish	25000	Clerk	20

ordid	emp_id	amount
1001	102	3000
1002	101	560
1004	108	7800
1008	103	5600

```
mysql> select * from employee,orders where employee.id = orders.emp_id;
```

Id	Name	Salary	desig	dept	ordid	emp_id	amount
102	Helen	55000	Manager	10	1001	102	3000
101	Sri	45000	Manager	10	1002	101	560
103	Karan	85000	President	20	1008	103	5600

Joins - examples

Left join, joins two tables and fetches rows based on a condition, which are matching in both the tables. The unmatched rows will also be available from the left table before the join clause.

Id	Name	Salary	desig	dept
101	Sri	45000	Manager	10
102	Helen	55000	Manager	10
103	Karan	85000	President	20
104	Yogesh	78000	President	10
105	Krish	25000	Clerk	20

ordid	emp_id	amount
1001	102	3000
1002	101	560
1004	108	7800
1008	103	5600

```
mysql> select * from employee left join orders on employee.id=orders.emp_id;
```

Id	Name	Salary	desig	dept	ordid	emp_id	amount
101	Sri	45000	Manager	10	1002	101	560
102	Helen	55000	Manager	10	1001	102	3000
103	Karan	85000	President	20	1008	103	5600
104	Yogesh	78000	President	10	NULL	NULL	NULL
105	Krish	25000	Clerk	20	NULL	NULL	NULL

Joins - examples

Right join, joins two tables and fetches rows based on a condition, which are matching in both the tables. The unmatched rows will also be available from the right table before the join clause.

Id	Name	Salary	desig	dept
101	Sri	45000	Manager	10
102	Helen	55000	Manager	10
103	Karan	85000	President	20
104	Yogesh	78000	President	10
105	Krish	25000	Clerk	20

ordid	emp_id	amount
1001	102	3000
1002	101	560
1004	108	7800
1008	103	5600

```
mysql> select * from employee right join orders on employee.id = orders.emp_id;
```

Id	Name	Salary	desig	dept	ordid	emp_id	amount
102	Helen	55000	Manager	10	1001	102	3000
101	Sri	45000	Manager	10	1002	101	560
NULL	NULL	NULL	NULL	NULL	1004	108	7800
103	Karan	85000	President	20	1008	103	5600

Joins - examples

inner join, joins both participating tables as long as there is a match between the columns.A SQL inner join is same as join clause.

E

Id	Name	Salary	desig	dept
101	Sri	45000	Manager	10
102	Helen	55000	Manager	10
103	Karan	85000	President	20
104	Yogesh	78000	President	10
105	Krish	25000	Clerk	20

ordid	emp_id	amount
1001	102	3000
1002	101	560
1004	108	7800
1008	103	5600

```
mysql> select * from employee  inner join orders on employee.id = orders.emp_id;
+-----+-------+--------+-----------+------+-------+--------+--------+
| Id  | Name  | Salary | desig     | dept | ordid | emp_id | amount |
+-----+-------+--------+-----------+------+-------+--------+--------+
| 102 | Helen |  55000 | Manager   |   10 |  1001 |    102 |   3000 |
| 101 | Sri   |  45000 | Manager   |   10 |  1002 |    101 |    560 |
| 103 | Karan |  85000 | President |   20 |  1008 |    103 |   5600 |
+-----+-------+--------+-----------+------+-------+--------+--------+
3 rows in set (0.00 sec)

mysql> select * from employee join orders on employee.id=orders.emp_id;
+-----+-------+--------+-----------+------+-------+--------+--------+
| Id  | Name  | Salary | desig     | dept | ordid | emp_id | amount |
+-----+-------+--------+-----------+------+-------+--------+--------+
| 102 | Helen |  55000 | Manager   |   10 |  1001 |    102 |   3000 |
| 101 | Sri   |  45000 | Manager   |   10 |  1002 |    101 |    560 |
| 103 | Karan |  85000 | President |   20 |  1008 |    103 |   5600 |
+-----+-------+--------+-----------+------+-------+--------+--------+
```

Joins - examples

self join, joins the table to itself (unary operation) self join is join of two copies of same table.

Note: Aliases play an important role here.

E

Id	Name	Salary	desig	dept
101	Sri	45000	Manager	10
102	Helen	55000	Manager	10
103	Karan	85000	President	20
104	Yogesh	78000	President	10
105	Krish	25000	Clerk	20

ordid	emp_id	amount
1001	102	3000
1002	101	560
1004	108	7800
1008	103	5600

```
mysql> select * from employee T1,employee T2 where T1.id = T2.id;
+-----+--------+--------+-----------+------+-----+--------+--------+-----------+------+
| Id  | Name   | Salary | desig     | dept | Id  | Name   | Salary | desig     | dept |
+-----+--------+--------+-----------+------+-----+--------+--------+-----------+------+
| 101 | Sri    |  45000 | Manager   |   10 | 101 | Sri    |  45000 | Manager   |   10 |
| 102 | Helen  |  55000 | Manager   |   10 | 102 | Helen  |  55000 | Manager   |   10 |
| 103 | Karan  |  85000 | President |   20 | 103 | Karan  |  85000 | President |   20 |
| 104 | Yogesh |  78000 | President |   10 | 104 | Yogesh |  78000 | President |   10 |
| 105 | Krish  |  25000 | Clerk     |   20 | 105 | Krish  |  25000 | Clerk     |   20 |
+-----+--------+--------+-----------+------+-----+--------+--------+-----------+------+
```