Practical 4 - Perform

* Simple Queries
* Simple Queries with Aggregate functions

|  |  |
| --- | --- |
| Q.1 | Creating Tables (With Constraints)-  Q.) Create an employee table having columns as employee number , employee name , date of joining and salary with the column values not excepting null values and employee number as primary key.  A.)  Mysql>create table Emp2 (EmpNo integer(5) NOT NULL, EmpName varchar(20) NOT NULL, JoinDate date NOT NULL, EmpSal integer(8) NOT NULL, PRIMARY KEY(EmpNo));  Mysql>show tables;  Mysql> desc Emp2; |
| Q.2 | Inserting Records in a Table  Insert at least 10 records to emp1 table, with at least 2 employees having salary less than 10000 and one employee having salary as 10000 . Example    C:\Users\RDF420~1.NAT\AppData\Local\Temp\ksohtml5392\wps1.png     * Insert One record at a time –     Mysql> insert into Emp1 (EmpNo , EmpName , JoinDate , EmpSal )  Values (101,’Rajesh’,’2021-10-23’,10000);  Mysql>select \* from emp1;     * Insert multiple records at a time –     Mysql> insert into Emp1 values (101,’Rajesh’,’2021-10-23’,10000), (102,’Kamlesh’,’2020-01-01’,19000), (103,’Ramesh’,’1998-03-07’,20000);  Mysql>select \* from emp1; |
| Q.3 | Updating Records in a Table  A: Change the salary for employee Rajesh as 50000  Mysql> update emp1 set empsal = 50000 where empname=’Rajesh’;  Mysql>select \* from emp1;  B: Change the date of joining to 2019-04-15 for employee 107  Mysql> update emp1 set joindate=’2019-04-15’ where empno=107;  Mysql>select \* from emp1;    C: Increase the salaries to 15000 for all the employees whose salary is less than or equal to 10000  Mysql> update emp1 set EmpSal = 15000 where empsal <= 10000;  Mysql>select \* from emp1; |
| Q.4 | Deleting Records in a Table  Q.) write a query to remove the employee record for employee no 109 from employee table  A.)  Mysql>select \* from emp1;  Mysql> delete from emp1 where empno=107;  Mysql>select \* from emp1; |
| Q.5 | Create a customers table with the following columns and data.    +----+----------+-----+-----------+----------+  | ID | NAME | AGE | ADDRESS | SALARY |  +----+----------+-----+-----------+----------+  | 1 | Ramesh | 32 | Ahmedabad | 2000.00 |  | 2 | Khilan | 25 | Delhi | 1500.00 |  | 3 | kaushik | 23 | Kota | 2000.00 |  | 4 | Chaitali | 25 | Mumbai | 6500.00 |  | 5 | Hardik | 27 | Bhopal | 8500.00 |  | 6 | Komal | 22 | MP | 4500.00 |  | 7 | Muffy | 24 | Indore | 10000.00 |  +----+----------+-----+-----------+----------+ |
|  | Perform Simple Queries with Where Operators     1. fetch the ID, Name and Salary fields from the CUSTOMERS table, where the salary is greater than 2000     SQL> SELECT ID, NAME, SALARY  FROM CUSTOMERS  WHERE SALARY > 2000;     1. fetch the ID, Name and Salary fields from the CUSTOMERS table for a customer with the name **Hardik**.     SQL> SELECT ID, NAME, SALARY  FROM CUSTOMERS  WHERE NAME = 'Hardik';     1. fetch id,name and salary form customers whose’s name starts with k     SQL> SELECT ID, NAME, SALARY  FROM CUSTOMERS  WHERE NAME LIKE ‘K%’; |
| Q.6 | Perform Where with Keywords and Logical Operators     1. (AND) fetch the ID, Name and Salary fields from the CUSTOMERS table, where the salary is greater than 2000 and the age is less than 25 years –   SQL> SELECT ID, NAME, SALARY  FROM CUSTOMERS  WHERE SALARY > 2000 AND age < 25;     1. (OR) fetch the ID, Name and Salary fields from the CUSTOMERS table, where the salary is greater than 2000 OR the age is less than 25 years.   SQL> SELECT ID, NAME, SALARY  FROM CUSTOMERS  WHERE SALARY > 2000 OR age < 25;     1. (IN ) Fetch customer name and address from CUSTOMERS table , where address IN Bhopal or MUMBAI   SQL> SELECT name, address  FROM customers  WHERE Address IN ('BHOPAL', 'MUMBAI');     1. (NOT ) Fetch customer name and address from CUSTOMERS table , where address is not indore or MUMBAI   SQL> SELECT name, address  FROM customers  WHERE Address NOT IN ('INDORE', 'MUMBAI');   1. (BETWEEN) fetch the customer details where customer id between 4 and 7.   SQL> SELECT \*  FROM customers  WHERE id BETWEEN 4 AND 7; |
| Q.7 | Simple Queries with Aggregate functions  Execute the below statement to create an Works table:  **CREATE** **TABLE** Works(  name varchar(45) **NOT NULL**,  gender varchar(6) **NOT NULL**,  occupation varchar(35) **NOT NULL**,  working\_date date,  working\_hours varchar(10)  );  Insert data as:  **INSERT** **INTO** Works **VALUES**  ('Robin', ‘Male’, 'Scientist', '2020-10-04',12),  ('Warner', ‘Male’, 'Engineer', '2020-10-04',10),  ('Patricia', ‘Female’, 'Actor', '2020-10-04',13),  ('Marco', ‘Male’, 'Doctor', '2020-10-04',14),  ('Brayden', ‘Male’, 'Teacher', '2020-10-04',12),  ('Anita', ‘Female’, 'Business', '2020-10-04',11),  ('Roshani', ‘Female’, 'Analyst', '2021-12-14',10),  ('Wishu', ‘Male’, 'Engineer', '2022-11-24',8),  ('Preet', ’Male’, 'Anchor', '2020-04-01',9),  ('Maria', ‘Female’, 'Doctor', '2023-01-01',8),  ('Badri', ‘Male’, 'Teacher', '2021-06-01',16),  ('Ananda', ‘Male’, 'Steward', '2022-08-05',12),  (‘Roshan', ‘Male’, 'Engineer', '2020-03-23',10),  ('Preesha', ’Female’, 'Actor', '2021-04-09',13),  ('Madan', ‘Male’, 'Doctor', '2022-10-04',10),  ('Brajesh', ‘Male’, 'Teacher', '2023-09-01',08),  ('Anindita', ‘Female’, 'Business', '2020-09-01',10),  ('Roshani', ‘Female’, 'Analyst', '2021-12-14',10),  ('Watson',’Male’, 'Engineer', '2022-05-14',9),  ('Prachi',’Female’, 'Anchor', '2020-04-01',9),  ('Mehul',’Female’, 'Doctor', '2022-04-01',10);    Sql> SELECT \* FROM works;     1. (COUNT)get the total number of employee in the works table   sql> select count(\*) from works;   1. (SUM)calculate the total number of working hours of all employees in the works table   sql> select sum(working\_hours) as “Total Working Hours” from works;   1. (AVG)get the average working hours of all employees in the works table   sql> select avg(working\_hours) as “Average Working Hours” from works;   1. (MIN)get minimum working hours of an employee available in the table   sql>select min(working\_hours) as “Minimum Working hour” from Works;   1. (MAX)get maximum working hours of an employee available in the table   sql>select max(working\_hours) as “Maximum Working hour” from Works; |
| Q.8 | Queries with Aggregate functions (GROUP BY and HAVING clause)     1. Fetch the gender from works table.   Sql> SELECT gender FROM works GROUP BY gender;  Fetch total employee per occupation in reverse alphabetic   1. order.   Sql> SELECT occupation, count (\*) as “Total Employee” FROM works GROUP BY occupation ORDER BY occupation DESC   1. Fetch the average working hours of male and female employees.   Sql> SELECT gender, avg(working\_hours) FROM works GROUP BY gender ORDER BY gender   1. Fetch the number of male and female workers FROM works table.   Sql> SELECT gender, count (\*) as “Total” FROM works GROUP BY gender;   1. Fetch the total working hours, avg working hours per occupation, sum total working hours for total working hours greater than 18 hrs   Sql>SELECT occupation,  AVG(working\_hours) AS “Avg WH”,  SUM(working\_hours) AS “Total WH”,  FROM works  GROUP BY working\_hours  HAVING SUM(working\_hours) > 18  ORDER BY working\_hours DESC   1. to retrieve the total working hours of All the Male employees who are engineers and doctors   Sql> SELECT occupation, working\_hours  FROM works  WHERE gender =’Male’  GROUP BY occupation  HAVING occupation IN (‘Engineer’,’Doctor’);   1. using Aggregate function return the occupation and number of employees and apply having clause to filter the results to display occupation having more than 2 employees.   Sql> SELECT occupation,count(\*) as “Total Employee”  FROM works  GROUP BY occupation  HAVING count(\*) > 2; |