1. Filter all brands that have total sales >500 units in the entire year

*Data present for 2019 only*

|  |  |  |
| --- | --- | --- |
| Brand | Date | Unit Sold |
| B1 | 20190101 | 10 |
| B1 | 20190102 | 5 |
| B1 | 20190103 | 100 |
| B2 | 20190101 | 50 |
| B2 | 20190103 | 500 |

1. Table *customers* contains shipping address (column name - address) of each customer (column name - CustomerID). Find out customers whose shipping address contains *‘institute’* as a keyword anywhere in the address*.* Ignore cases.

Solution: Ilike keyword is used instead of like keyword to make the match case insensitive

Query: select \* from 'customers'

where address iLIKE '%institute%'

1. Table *employees* contain salary of each employee. Find out range of salaries being given in each department

|  |  |  |
| --- | --- | --- |
| Department | EmpID | Salary |
| D1 | E1 | 10 |
| D1 | E2 | 5 |
| D1 | E3 | 100 |
| D2 | E4 | 50 |
| D2 | E5 | 500 |

Query: select 'depid', max(salary) as MAX\_SALARY, min(salary) as MIN\_SALARY from "Employee" group by Department

1. Objective is to clean noise from data (column name - gender) : If gender is M then keep as M, if F then keep as F. If gender is marked as ‘m’ change to ‘M’, ‘f’ to F. All other values should be labeled ‘Unknown’

Query: select deparment, emp\_id,

case when gender = 'M' then 'M'

when gender = 'F' then 'F'

when gender = 'f' then 'F'

when gender = 'm' then 'M'

else 'Unknown'

end as gender\_new

from "table\_name";

1. Output of the following query:

*select case when null = null then 'True' else 'False' end as Result;*

Explain your answer.

This will always return false as this is the incorrect way to compare null with null. The correct way to compare is (null is null).

1. SQL query used to fetch unique values from a field?
   * SELECT UNIQUE column\_name FROM table\_name;
   * SELECT DISTINCT column\_name FROM table\_name
   * SELECT column\_name FROM table\_name WHERE COUNT(column\_name) = 1;
   * SELECT UNIQUE column\_name FROM table\_name WHERE COUNT(column\_name) = 1;

Answer – It’s the second choice – SELECT DISTINCT column\_name from table\_name

In postrgres- SELECT DISTINCT(column\_name) from ”table\_name”

1. How to change the value of field ‘Salary’ as 7500 for an Employee\_Name ‘John’ in a table Employee\_Details?

Answer: UPDATE Employee\_details

SET Salary = 7500

WHERE Employee\_Name = ‘John’;

1. What is the difference between delete, truncate and drop ?

Drop command (DDL): Drop command is used to delete the existing database objects such databases, tables, etc. Objects deleted using the drop command are deleted permanently and cannot be reobtained.

Truncate command (DDL): Unlike the drop command Truncate command does not drop the entire table but it only drops all its records based on the query. The table structure is not deleted. Similar to the Drop command the records Truncated using the command cannot be reobtained.

Delete command (DML): Delete command is used delete a specific record or multiple records from a table with respect to the where condition. If no condition is specified then it will delete all the records.

1. What is the difference between union and union all ? Which of them will run faster ?

Answer: UNION and UNION ALL both the statements in SQL are used to concatenate tables.

Union statement when performed on two tables will return distinct elements of both the tables

if a record appears multiple times, it will be only taken once. UNION ALL statement will return all the records in both the tables if a record is repeated multiple times then it will be taken multiple times. UNION ALL is faster than UNION because UNION is equivalent of taking UNION ALL on the tables and then doing a DISTINCT command to remove duplicates.

10. Write a query to find out "CustomerName" who did highest "purchase" along with "purchase" value

Query: select Customer.cus\_name, max(orders.purchase)

from "Customer"

left join "orders" on Customer.cus\_id = orders.cus\_id;

|  |  |
| --- | --- |
| TABLE1 |  |
| CustomerID | CustomerName |
| 6 | A |
| 5 | F |
| 3 | C |
| 9 | R |
| 8 | G |
| 11 | H |
| 13 | D |
| 12 | P |
| 10 | S |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | TABLE2 |  |
| Order date | CustomerID | purchase |
| 20190101 | 3 | 1456 |
| 20190101 | 5 | 1190 |
| 20190101 | 6 | 1158 |
| 20190101 | 8 | 1973 |
| 20190101 | 9 | 1449 |
| 20190102 | 10 | 1897 |
| 20190102 | 11 | 1356 |
| 20190102 | 12 | 1003 |
| 20190102 | 5 | 1897 |
| 20190102 | 6 | 1356 |
| 20190102 | 10 | 1003 |
| 20190102 | 13 | 1003 |
| 20190103 | 6 | 1850 |
| 20190103 | 10 | 1135 |

11) Write the output in the below-required table

Table1 Table2

|  |
| --- |
| Col1 |
| 1 |
| 1 |
| 1 |
| 2 |
| 3 |
| 3 |
|  |

|  |
| --- |
| Col1 |
| 1 |
| 1 |
| 2 |
| 2 |
| 0 |
| 0 |
| 5 |

on clause

table1.col1=table2.col1

Output

|  |  |
| --- | --- |
|  | Number of Rows |
| Left join | 10 Rows affected |
| Right Join | 11 Rows affected |
| Inner Join | 8 Rows affected |
| Full Outer Join | 13 rows affected |