TERADATA CREATE DATABASE and USER

Um banco de dados Teradata é um repositório local para vários tipos de objetos de banco de dados, como Tabelas (requer espaço permanente), Exibições (não utiliza espaço permanente), Procedimentos (requer espaço permanente), funções, etc.

Sintaxe Teradata CREATE DATABASE

Abaixo está a sintaxe do Teradata create Database:

CREATE DATABASE database\_name

AS

[PERMANENT|PERM] = n BYTES,

SPOOL = n BYTES,

TEMPORARY = n BYTES;

O limite de espaço no banco de dados Teradata pode ser definido como abaixo :

* Perm Space – Quantidade máxima de espaço disponível para tabelas
* Spool Space – Quantidade máxima de espaço disponível para processamento de consultas
* Temp Space – Usado para criar uma tabela temporária

Um banco de dados recém-criado pertence a um banco de dados existente até que quaisquer objetos sejam criados no banco de dados, ele estará vazio.

Teradata CREATE DATABASE Exemplo

O exemplo a seguir cria um banco de dados UNIVERSITY:

CREATE DATABASE UNIVERSITY FROM SYSDBA

AS

PERM = 5368709120, -- 5GB

SPOOL = 5368709120, -- 5GB

TEMPORARY = 2147483648; -- 2GB

NO FALLBACK

NO BEFORE JOURNAL

NO AFTER JOURNAL;

Teradata CRIAR USUÁRIO

No Teradata, usuários e bancos de dados são exatamente os mesmos, exceto que o usuário recebe uma senha.

Igual ao banco de dados Teradata, os usuários do Teradata também podem manter diferentes objetos de banco de dados, como tabelas, exibições, macros e gatilhos. No entanto, um usuário não pode manter tabelas até que o espaço perm seja atribuído a ela.

Um usuário pode fazer logon no sistema Teradata usando senha e objeto de acesso dentro de:

* Em si
* Outros bancos de dados ou usuários para os quais tenha direito de acesso.

Teradata CREATE USER Sintaxe

A seguir está a sintaxe de CREATE USER no Teradata:

CREATE USER username

AS

[PERMANENT|PERM] = n BYTES,

PASSWORD = password,

TEMPORARY = n BYTES,

SPOOL = n BYTES;

Aqui tudo é igual a criação do DATABASE exceto o parâmetro PASSWORD onde você tem que fornecer uma senha para o usuário fazer o login.

Teradata CREATE USER Exemplo:

CREATE USER SYSDBA

AS

PERM = 5368709120,

PASSWORD = password1

SPOOL= 5368709120,

TEMPORARY = 2147483648;

No exemplo acima, criamos um usuário SYSDBA com espaço PERM 5368709120 ou 5 GB, espaço SPOOL 5368709120 ou 5 GB e espaço TEMPORÁRIO 2147483648 ou 2 GB.

Um usuário recém-criado também permanece vazio até que quaisquer objetos sejam criados nele.

A próxima etapa é fornecer todos os privilégios ao usuário SYSDBA para executar operações no usuário SYSDBA. Esta etapa deve ser executada após o login do usuário dbc.

GRANT ALL ON SYSDBA to SYSDBA WITH GRANT OPTION;

**Teradata Drop Database**

A instrução Teradata Drop Database é usada para descartar um banco de dados vazio do sistema Teradata.

Antes de descartar um banco de dados no Teradata, o ponto abaixo deve ser observado:

* O usuário que você está usando para descartar um banco de dados deve ter os privilégios apropriados (banco de dados DROP) para descartar um banco de dados.
* O banco de dados que você está descartando deve estar vazio antes de emitir a instrução drop database. Isso significa que o banco de dados não deve conter nenhum objeto como tabelas, visualizações, macros, procedimentos definidos pelo usuário, etc.
* Para descartar um banco de dados contendo outros objetos de banco de dados, você deve usar delete uma instrução de banco de dados para torná-lo vazio.
* Depois que um banco de dados é descartado, o espaço PERM e TEMP do banco de dados deve ser devolvido ao banco de dados ou usuário do proprietário imediato.

Sintaxe do banco de dados Teradata Drop

A sintaxe do Teradata Drop Database é a seguinte.

DROP DATABASE database\_name;

Exemplo de banco de dados Teradata Drop

A instrução a seguir descarta um banco de dados vazio chamado UNIVERSITY.

DROP DATABASE UNIVERSITY;

**Teradata DELETE DATABASE**

A instrução DELETE DATABASE no Teradata é usada para excluir todos os objetos do banco de dados, como tabelas de dados, exibições, macros, gatilhos, procedimentos de armazenamento e funções definidas pelo usuário de um banco de dados.

Para excluir um banco de dados no Teradata, você deve ter os privilégios DROP no banco de dados especificado.

Sintaxe de banco de dados de exclusão do Teradata

DELETE DATABASE database\_name [ALL];

Aqui,

* database\_name – O nome do banco de dados do qual você deseja remover todos os objetos.
* ALL – Opcional. All é especificado para excluir todos os objetos, incluindo tabelas temporárias globais materializadas.

Exemplo de banco de dados de exclusão do Teradata

A instrução a seguir exclui todos os objetos do banco de dados UNIVERSITY.

DELETE DATABASE UNIVERSITY;

Para remover completamente uma estrutura de banco de dados do dicionário de dados, você deve usar a instrução Teradata Drop Database.

**Create Table in Teradata**

A criação de uma tabela no Teradata é quase semelhante a outros bancos de dados relacionais. CREATE TABLE ou CT é o comando usado para criar uma tabela no Teradata.

CREATE TABLE Teradata Sintaxe:

CREATE SET/MULTISET TABLE  database\_name.table\_name, [NO] FALLBACK

   (column\_name   data\_type attribute,

    column\_name   data\_type attribute,

    column\_name   data\_type attribute)

[UNIQUE][NO] PRIMARY INDEX (primary\_index\_column);

A sintaxe SET/MULTISET é usada para definir a exclusividade dos registros na tabela. A tabela SET não permite nenhum registro duplicado enquanto a tabela MULTISET permite registros duplicados. Se você não especificar nada, o Teradata por padrão cria SET.

FALLBACK define que a cópia duplicada da tabela será mantida em caso de falha. O padrão é NO fallback. Defina FALLBACK apenas nas tabelas críticas do banco de dados, pois consome espaço extra.

Database\_Name: Onde a tabela será criada.

Table name: O nome da tabela.

Column\_name: Nome da coluna que você deseja na tabela.

Data\_type: especifica um único tipo de dados para uma coluna.

Atributo: define atributos para uma coluna. Exemplo – NOT NULL, NOT CASE SPECIFIC, MAIÚSCULAS, FORMATO, etc.

[UNIQUE/NO] PRIMARY INDEX: Você deve definir pelo menos um índice primário no Teradata que pode ser exclusivo ou não exclusivo por natureza. Se você não quiser nenhum índice primário em sua tabela, existe uma opção chamada NO PRIMARY INDEX no Teradata.

A coluna de índice primário define a(s) coluna(s) que funcionarão como um índice primário no Teradata. A coluna ou combinação de colunas pode ser exclusiva ou não exclusiva.

CREATE TABLE Teradata Exemplo:

O exemplo a seguir cria uma tabela definida chamada Employee em Employee\_database com a opção FALLBACK.

CREATE SET TABLE Employee\_Database.Employee, FALLBACK

   (Employee\_Id     INTEGER,

    Employee\_Name   CHAR(25),

    Salary           DECIMAL(8,2),

    DOB              DATE FORMAT 'YYYY-MM-DD',

    Job\_Title        VARCHAR(25),

    Dept\_No          SMALLINT,

    Marital\_Status   CHAR,

    No\_Of\_Dependents BYTEINT)

UNIQUE PRIMARY INDEX (Employee\_Id);

Depois que a tabela é criada, você pode executar o comando SHOW TABLE para ver a definição da tabela no Teradata.

**show** **table** Employee\_Database.Employee**;**

**CREATE** **SET** **TABLE** Employee\_Database.Employee ,**FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** MERGEBLOCKRATIO

     (

      Employee\_Id INTEGER,

      Employee\_Name CHAR(25) CHARACTER **SET** LATIN **NOT** **CASESPECIFIC**,

      Salary DECIMAL(8,2),

      DOB DATE FORMAT 'YYYY-MM-DD',

      Job\_Title VARCHAR(25) CHARACTER **SET** LATIN **NOT** **CASESPECIFIC**,

      Dept\_No SMALLINT,

      Marital\_Status CHAR(1) CHARACTER **SET** LATIN **NOT** **CASESPECIFIC**,

      No\_Of\_Dependents BYTEINT)

**UNIQUE** **PRIMARY** **INDEX** ( Employee\_Id )**;**

Outras versões de criação de tabelas no Teradata:

Até agora você aprendeu como criar uma tabela no Teradata. Existem outros vários métodos pelos quais você pode obter a tabela desejada rapidamente.

Teradata - criar tabela as select:

O exemplo abaixo mostra como criar uma tabela de funcionários com base em seu estado civil. Suponha que você queira apenas os funcionários cujo estado civil é SOLTEIRO. Isso pode ser feito facilmente usando nossa tabela existente.

**CREATE** **TABLE** Employee\_Database.bachelor\_employee **AS** (

**SELECT**

      \*

**FROM** Employee\_Database.Employee e

**WHERE**

      e.Marital\_Status = 'S'

   )

**WITH** **DATA;**

Teradata - criar tabela a partir de outra tabela com dados completos:

O exemplo abaixo apenas criará outra cópia da tabela de funcionários com dados.

**CREATE** **TABLE** Employee\_Database.new\_employee **AS** (

**SELECT**

      \*

**FROM** Employee\_Database.Employee

   )

**WITH** **DATA;**

**Teradata RENAME TABLE**

A instrução RENAME TABLE no Teradata renomeia uma tabela existente no Teradata.

Para renomear uma tabela existente no Teradata, você deve ter privilégios DROP na tabela e privilégios CREATE TABLE no banco de dados ou usuário que a contém.

Teradata RENAME TABLE Sintaxe

A sintaxe para Teradata RENAME TABLE é a seguinte:

RENAME TABLE database\_name.old\_tbl\_name to database\_name.new\_tbl\_name;

Aqui,

* database\_name – O nome do banco de dados que contém a tabela.
* old\_tbl\_name – O nome da tabela existente.
* new\_tbl\_name – O novo nome da tabela que você deseja manter.

Teradata RENAME TABLE Exemplo

A instrução a seguir renomeia um aluno de nome de tabela existente para student\_details no banco de dados Teradatapoint.

RENAME TABLE Teradatapoint.student to Teradatapoint.student\_details;

**Teradata DROP TABLE**

A instrução DROP TABLE remove uma tabela existente junto com os dados e a estrutura da tabela do banco de dados Teradata. Ele remove a estrutura da tabela do dicionário de dados. Como resultado da eliminação da tabela, a estrutura da tabela junto com os dados serão perdidos permanentemente.

Depois que uma tabela é descartada, ela não pode ser recuperada. Portanto, certifique-se de estar no banco de dados correto antes de descartar uma tabela.

Sintaxe Teradata DROP TABLE

A sintaxe para Teradata DROP TABLE é a seguinte:

DROP TABLE database\_name.tbl\_name;

Aqui,

database\_name – o nome do banco de dados pai onde a tabela reside.

tbl\_name – O nome da tabela que você deseja descartar.

Nota:- Você deve ter privilégios DROP na tabela para descartá-la.

Exemplo Teradata DROP TABLE

Você pode descartar uma tabela do Teradata de duas maneiras.

Opção 1:

A instrução a seguir remove a tabela do aluno do banco de dados Teradatapoint. Aqui, estamos fornecendo um nome de tabela totalmente qualificado, ou seja, nome da tabela junto com o nome do banco de dados.

DROP TABLE Teradatapoint.student;

Opção 2:

Aqui, primeiro selecionamos o banco de dados proprietário e, em seguida, emitimos a instrução DROP DATABASE.

DATABASE Teradatapoint;

DROP TABLE student;

O mesmo nome de tabela pode estar presente em diferentes bancos de dados. Portanto, precisamos ter certeza de que selecionamos o banco de dados correto antes de executar a instrução DROP TABLE.

**Teradata TRUNCATE TABLE**

Na maioria dos RDBMS populares, como SQL Server, Oracle, MySQL, etc, usamos a instrução de tabela truncada para excluir todos os registros de uma tabela existente. A diferença entre DELETE TABLE e TRUNCATE TABLE é que aqui no caso de TRUNCATE TABLE nenhum log transacional é mantido para cada linha deletada. Portanto, TRUNCATE TABLE é mais rápido em comparação com a instrução DELETE TABLE.

Infelizmente, não há nenhum comando TRUNCATE no Teradata. Como alternativa, você pode usar a instrução DELETE ALL.

Sintaxe

Abaixo está a sintaxe para excluir todas as linhas de uma tabela existente no Teradata.

DELETE database\_name.tbl\_name ALL;

Exemplo

A instrução a seguir exclui todos os registros da tabela de funcionários no banco de dados Teradatapoint.

DELETE Teradatapoint.employee ALL;

Teradata DELETE TABLE

A instrução Teradata DELETE TABLE remove todos os registros de uma tabela Teradata. A instrução DELETE TABLE excluirá apenas registros, a estrutura da tabela permanecerá a mesma.

Teradata DELETE TABLE Sintaxe

A sintaxe para excluir todos os registros:

DELETE FROM database\_name.tbl\_name;

Aqui,

database\_name – o nome do banco de dados proprietário da tabela.

tbl\_name – O nome da tabela de onde você deseja excluir todos os registros.

A sintaxe para excluir registros selecionados:

DELETE FROM database\_name.tbl\_name where condition;

Aqui,

where condition – A condição para excluir linha(s) específica(s).

Teradata DELETE TABLE exemplo

A instrução a seguir exclui todos os registros do banco de dados do aluno no banco de dados Teradatapoint.

DELETE FROM Teradatapoint.student;

O exemplo abaixo exclui apenas o funcionário específico da tabela de funcionários.

DELETE FROM Teradatapoint.employee where empid = 123;

Teradata Alter Table Add Column

A instrução ALTER TABLE ADD COLUMN do Teradata adiciona uma nova coluna a uma tabela existente do Teradata sem eliminá-la. Além de adicionar uma nova coluna, a função ADD tem a capacidade de adicionar tipos de dados, tamanho, se a nova coluna deve ou não permitir valores NULL e se a nova coluna tem ou não um valor padrão associado a ela.

Teradata Alter Table Adicionar Coluna Sintaxe

A sintaxe do Teradata Alter Table Add Column é a seguinte:

ALTER TABLE database\_name.tbl\_name ADD column\_name datatype;

Exemplo de adição de coluna de alteração de tabela do Teradata

Considere uma tabela de funcionários com a definição abaixo:

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC

)

UNIQUE PRIMARY INDEX ( empid );

Agora, vamos adicionar 2 novos hobbies de coluna e native\_place com os seguintes detalhes, um por um.

ALTER TABLE Teradatapoint.employee ADD hobbies varchar(50);

ALTER TABLE Teradatapoint.employee ADD hobbies varchar(50) DEFAULT 'Listen to Music';

ALTER TABLE Teradatapoint.employee ADD native\_place varchar(50) NOT NULL;

Na primeira instrução, adicionamos uma nova coluna hobbies na tabela de funcionários com varchar(50). Em seguida, adicionamos o valor DEFAULT para a coluna como 'Ouvir música'. E, finalmente, adicionei outra nova coluna native\_place com restrição NOT NULL.

Agora, se você verificar a definição da tabela de funcionário, verá como abaixo:

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC,

hobbies VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC DEFAULT 'Listen to Music',

native\_place VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)

)

UNIQUE PRIMARY INDEX ( empid );

A instrução Teradata Alter Table Drop Column é usada para remover uma coluna existente de uma tabela Teradata. Você também pode descartar várias colunas em uma instrução usando Alter Table Drop Column.

Sintaxe ALTER TABLE DROP COLUMN do Teradata

A sintaxe de Teradata ALTER TABLE DROP COLUMN é a seguinte.

ALTER TABLE database\_name.tbl\_name DROP column\_name;

A sintaxe para descartar várias colunas é a seguinte.

ALTER TABLE database\_name.tbl\_name

DROP column\_name1,

DROP column\_name2

....

DROP column\_name(n);

Teradata ALTER TABLE DROP COLUMN Exemplo

Considere a tabela de funcionários com a definição abaixo:

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC,

hobbies VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC DEFAULT 'Listen to Music',

native\_place VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)

)

UNIQUE PRIMARY INDEX ( empid );

Dropando uma coluna no Teradata

O exemplo abaixo descarta a coluna empname da tabela de employee.

ALTER TABLE Teradatapoint.employee

DROP empname;

Exemplo de eliminação de várias colunas no Teradata

O exemplo abaixo remove as colunas hobbies e native\_place da tabela de funcionários.

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER

)

UNIQUE PRIMARY INDEX ( empid );

**Teradata ALTER TABLE MODIFY COLUMN**

A coluna de modificação ALTER TABLE do Teradata é usada para modificar o tipo de dados, comprimento ou formato da coluna. A opção de alterar o comprimento de um tipo de dados é limitada no Teradata. No entanto, podemos usar a sintaxe abaixo para alterar o comprimento do tipo de dados apenas para varchar.

Sintaxe:

ALTER TABLE database\_name.tbl\_name

ADD existing\_column\_name new\_data\_type (New Length)

Exemplo

Suponha que temos uma tabela de funcionários com a definição abaixo:

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC,

hobbies VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC DEFAULT 'Listen to Music',

native\_place VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)

)

UNIQUE PRIMARY INDEX ( empid );

Agora, se você quiser alterar o comprimento de empname para varchar(100), poderá usar a seguinte instrução:

ALTER TABLE Teradatapoint.employee ADD empname varchar(100);

Agora, você pode verificar a definição da tabela conforme abaixo:

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee , NO FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(100) CHARACTER SET LATIN NOT CASESPECIFIC FORMAT 'X(50)',hobbies VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC DEFAULT 'Listen to Music',

native\_place VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)

)

UNIQUE PRIMARY INDEX ( empid );

Para modificar o tipo de dados de char para int ou para diminuir o tamanho da coluna varchar, a tabela precisa ser eliminada e recriada.

Vamos tentar diminuir o tamanho do empname agora novamente para varchar(50).

ALTER TABLE Teradatapoint.employee ADD empname varchar(50);

-- Syntax error 3558

A seguir estão os casos sem sucesso.

CHAR (De CHAR(20) para CHAR(25)) — Erro de sintaxe 3558

TIMESTAMP (De TIMESTAMP(0) para TIMESTAMP(1)) — Erro de sintaxe 3558

DECIMAL (De DECIMAL(10,0) para DECIMAL(10,1)) — Erro de sintaxe 3558

INTEGER (de INTEGER para DECIMAL(10,0)) — Erro de sintaxe 3558

**Teradata Alter Table Rename Column**

A opção Teradata Alter Table com Rename é usada para renomear um nome de coluna na tabela Teradata.

A sintaxe do Teradata Alter Table Rename Column é a seguinte:

Alter Table database\_name.tbl\_name

Rename column\_name to new\_column\_name;

Exemplo de coluna de renomeação de tabela de alteração do Teradata

Considere a definição da tabela de funcionários da seguinte maneira:

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC,

hobbies VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC DEFAULT 'Listen to Music',

native\_place VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)

)

UNIQUE PRIMARY INDEX ( empid );

O exemplo a seguir renomeia a coluna hobbies para extra\_curricular.

Alter Table database\_name.tbl\_name

Rename hobbies to extra\_curricular;

Você pode verificar o mesmo na definição da tabela de funcionário agora.

show table Teradatapoint.employee;

CREATE SET TABLE Teradatapoint.employee ,FALLBACK ,

NO BEFORE JOURNAL,

NO AFTER JOURNAL,

CHECKSUM = DEFAULT,

DEFAULT MERGEBLOCKRATIO,

MAP = TD\_MAP1

(

empid INTEGER,

empname VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC,

hobbies VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC DEFAULT 'Listen to Music',

extra\_curricular VARCHAR(50) CHARACTER SET LATIN NOT CASESPECIFIC NOT NULL)

)

UNIQUE PRIMARY INDEX ( empid );

**Teradata SELECT Statement**

SELECT é a mais simples entre todas as outras instruções e é usada para recuperar os registros da tabela.

Sintaxe Teradata SELECT

A seguir está a sintaxe básica de uma instrução SELECT.

SELECT column1, column2, ...

FROM tablename;

Teradata SELECT Exemplo:

Considere a tabela de clientes abaixo:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| cust\_id | income | years\_with\_bank | nbr\_children | gender | marital\_status |
| 1,362,487 | 6,605 | 1 | 0 | M | 2 |
| 1,362,956 | 0 | 4 | 0 | F | 1 |
| 1,362,752 | 47,668 | 3 | 0 | F | 1 |
| 1,363,221 | 0 | 4 | 0 | F | 1 |
| 1,362,813 | 15,684 | 3 | 2 | F | 2 |
| 1,362,609 | 1,929 | 8 | 0 | F | 2 |

A seguir está o exemplo de uma consulta SELECT:

**SELECT**    cust\_id, income, age, years\_with\_bank

**FROM** teradatapoint.customer;

Se você executar a consulta acima, ela recuperará as colunas cust\_id, income, age e years\_with\_bank da tabela customer.

   cust\_id       income     age  years\_with\_bank

-----------  -----------  ------  ---------------

    1362487         6605      71                1

    1362956            0      10                4

    1362752        47668      54                3

    1363221            0      10                4

    1362813        15684      53                3

    1362609         1929      79                8

Se você deseja recuperar todas as colunas da tabela de clientes, pode usar a consulta abaixo.

**SELECT**    \* **FROM** teradatapoint.customer**;**

Teradata Order by

A cláusula order by pode ser utilizada para colocar os registros em ordem de acordo com a sua necessidade. Pode ser ascendente ou descendente. O padrão é crescente.

Teradata Ordem por Sintaxe

A seguir está a sintaxe da instrução SELECT com a cláusula ORDER BY:

SELECT column1, column2, ....

FROM tablename

ORDER BY 1 DESC;

Order by Teradata exemplo

A seguir está um exemplo de uma instrução SELECT com cláusula ORDER BY.

**SELECT**    \* **FROM** teradatapoint.customer**order** **by** 1**;**

Ao executar a consulta acima, ela fornecerá o conjunto de resultados com cust\_id em ordem crescente.

cust\_id       income     age  years\_with\_bank  nbr\_children  gender  marital\_status

-----------  -----------  ------  ---------------  ------------  ------  --------------

1362487         6605      71                1             0  M       2

1362609         1929      79                8             0  F       2

1362752        47668      54                3             0  F       1

1362813        15684      53                3             2  F       2

1362956            0      10                4             0  F       1

1363221            0      10                4             0  F       1

Teradata Group by

O GROUP BY é usado para obter os registros semelhantes em grupos.

Grupo Teradata por sintaxe

A seguir está a sintaxe da instrução SELECT com a cláusula GROUP BY

SELECT coulmn1, column2, ...

from tablename

group by 1;

Grupo By Teradata por exemplo

O exemplo a seguir agrupa os registros por years\_with\_bank e identifica o número de clientes para cada grupo:

**SELECT**    years\_with\_bank,count(\*) **FROM** teradatapoint.customer

**group** **by** years\_with\_bank

**order** **by** years\_with\_bank**;**

Abaixo está a saída da consulta acima.

years\_with\_bank     Count(\*)

---------------  -----------

              1            1

              3            2

              4            2

              8            1

Teradata cláusula WHERE

A cláusula WHERE é usada para filtrar os registros retornados pela instrução SELECT. Somente os registros que atendem à condição na cláusula WHERE são retornados.

Sintaxe da cláusula Where do Teradata

A seguir está a sintaxe da instrução SELECT com a cláusula WHERE:

SELECT \* do nome da tabela

ONDE [condição];

Cláusula Teradata WHERE Exemplo

A consulta a seguir busca registros em que cust\_id é 1362487:

**SELECT**    \* **FROM** teradatapoint.customer **WHERE** cust\_id=1362487**;**

Abaixo está a saída da consulta acima.

cust\_id       income     age  years\_with\_bank  nbr\_children  gender  marital\_status

-----------  -----------  ------  ---------------  ------------  ------  --------------

    1362487         6605      71                1             0  M       2

**Insert/Select in Teradata**

A maioria de nós está familiarizada com a Instrução Insert pela qual inserimos dados em uma tabela. Aqui, estamos usando a palavra-chave “values” para especificar dados para as colunas.

Inserir/Selecionar fornece flexibilidade para inserir dados em uma tabela selecionando dados de uma ou mais tabelas. A instrução SELECT lê os dados de uma ou mais tabelas e os usa como valores para inserir em outra tabela. Basta colocar a consulta SELECT no lugar da parte de valores da instrução Insert.

Inserir/Selecionar na Sintaxe do Teradata

Abaixo está a sintaxe para inserir todos os registros de todas as colunas em outra tabela.

INSERT INTO database\_name.tbl\_2

SELECT \* FROM database\_name.tbl\_1;

Aqui,

* database\_name – o nome do banco de dados proprietário da tabela.
* tbl\_2 – O nome da tabela onde você deseja inserir os dados.
* tbl\_1 – O nome da tabela de onde você deseja selecionar os dados.

Abaixo está a sintaxe para selecionar e inserir colunas selecionadas de uma ou mais tabelas para outra tabela.

INSERT INTO database\_name.tbl\_2

SELECT (column\_1,column\_2,..,column\_n) from database\_name.tbl\_2;

OU

INSERT INTO database\_name.tbl\_2 (column\_1,column\_2,..,column\_n)

SELECT (column\_1,column\_2,..,column\_n) from database\_name.tbl\_2;

Você pode usar funções agregadas, subconsultas, expressões também na instrução SELECT, mas lembre-se de que os tipos de dados das colunas de origem e destino devem ser os mesmos.

**Teradata Insert Multiple Rows**

O Teradata não oferece suporte a valores tradicionais com várias linhas ao inserir dados na tabela. Em vez disso, você pode usar o Multi Statement Request (MSR).

Sintaxe

Insert into dbname.tblname values (value1, vlaue2)

;Insert into dbname.tblname values (value3, vlaue4);

Exemplo

Vamos CRIAR uma tabela Volátil conforme abaixo.

CREATE VOLATILE TABLE PRODUCT

(

ProductID Integer,

ProductName varchar(20),

Price Decimal (15,2)

) ON COMMIT PRESERVE ROWS;

Podemos usar instruções Teradata MSR para inserir os dados na tabela volátil acima.

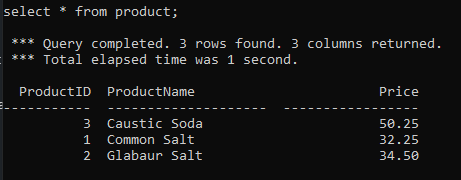
INSERT INTO PRODUCT VALUES (1,'Common Salt', 32.25)

;INSERT INTO PRODUCT VALUES (2,'Glabaur Salt',34.50)

;INSERT INTO PRODUCT VALUES (3,'Caustic Soda',50.25);

Se você estiver usando o Teradata SQL Assistant, poderá executar as instruções de inserção acima usando o botão executar paralelo.

Agora, você pode ver os dados usando a Instrução Teradata SELECT.



**Teradata UPDATE Statement**

A instrução Teradata UPDATE atualiza as colunas de uma tabela existente no Teradata. Você pode atualizar uma ou mais colunas usando uma única instrução UPDATE.

Sintaxe da instrução Teradata UPDATE

A sintaxe geral para a instrução Teradata UPDATE é a seguinte.

Para atualizar todos os registros em uma coluna

UPDATE DatabaseName.TableName

SET column\_name = value;

**Para atualizar registros específicos na coluna**

UPDATE DatabaseName.TableName

SET column\_name = value

WHERE condition;

**To update multiple columns in a table**

UPDATE DatabaseName.TableName

SET column1 = value1,

column2 = value2,

...

columnN = valueN

WHERE condition;

Exemplo de declaração Teradata UPDATE

Considere a seguinte tabela employee:

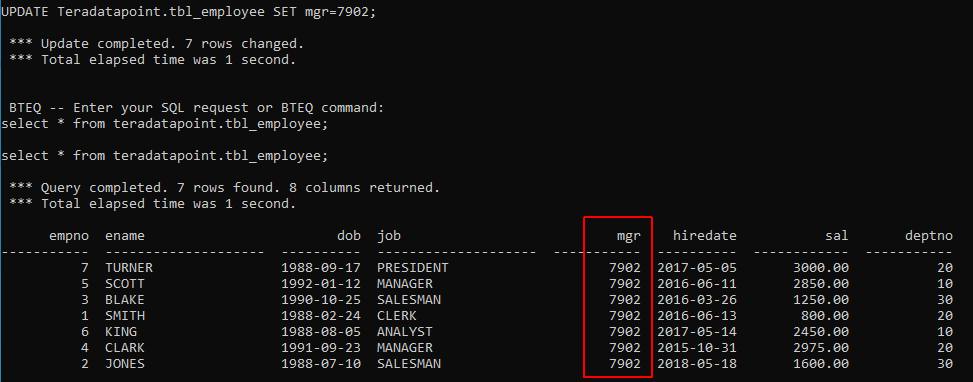
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| empno | ename | dob | job | mgr | hiredate | sal | deptno |
| 1 | SMITH | 24-02-1988 | CLERK | 7902 | 13-06-2016 | 800 | 20 |
| 2 | JONES | 10-07-1988 | SALESMAN | 7698 | 18-05-2018 | 1600 | 30 |
| 3 | BLAKE | 25-10-1990 | SALESMAN | 7698 | 26-03-2016 | 1250 | 30 |
| 4 | CLARK | 23-09-1991 | MANAGER | 7839 | 31-10-2015 | 2975 | 20 |
| 5 | SCOTT | 12-01-1992 | MANAGER | 7839 | 11-06-2016 | 2850 | 10 |
| 6 | KING | 05-08-1988 | ANALYST | 7839 | 14-05-2017 | 2450 | 10 |
| 7 | TURNER | 17-09-1988 | PRESIDENT | 7566 | 05-05-2017 | 3000 | 20 |

A instrução abaixo atualiza as colunas mgr para todos os funcionários.

UPDATE Teradatapoint.tbl\_employee

SET mgr = 7902;

Quando a consulta acima é executada, ela retorna os seguintes registros:

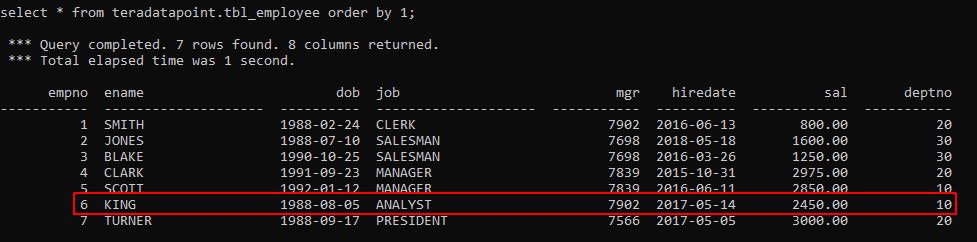


O exemplo abaixo atualiza mgr para o funcionário cujo empno é 6:

UPDATE Teradatapoint.tbl\_employee

SET mgr=7902

WHERE empno=6;



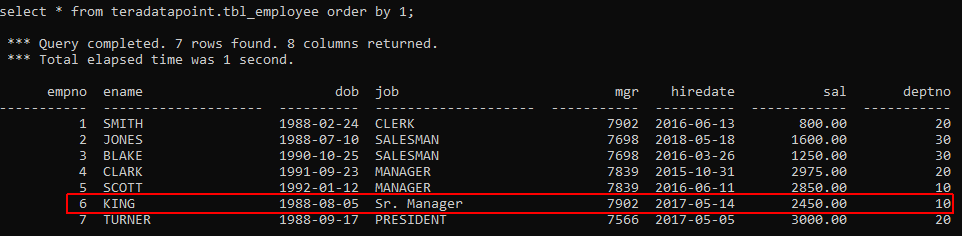
E a instrução SQL abaixo atualiza as colunas mgr e job:

UPDATE Teradatapoint.tbl\_employee

SET mgr=7902,

job='Sr. Manager'

WHERE empno=6;



**Teradata Delete Statement**

A instrução Teradata Delete exclui os registros de uma tabela. Você pode excluir todos os registros ou alguns registros específicos de uma tabela existente usando a instrução delete.

Sintaxe da declaração de exclusão do Teradata

A sintaxe para excluir todos os registros:

DELETE FROM DatabaseName.TableName;

A sintaxe para excluir registros específicos

DELETE FROM DatabaseName.TableName

WHERE condition;

Exemplo de instrução Teradata Delete

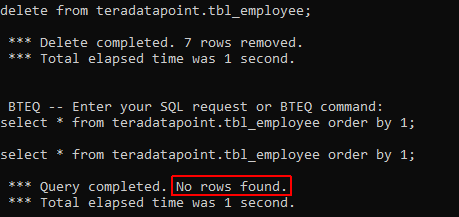
Considere a seguinte tabela employee:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| empno | ename | dob | job | mgr | hiredate | sal | deptno |
| 1 | SMITH | 24-02-1988 | CLERK | 7902 | 13-06-2016 | 800 | 20 |
| 2 | JONES | 10-07-1988 | SALESMAN | 7698 | 18-05-2018 | 1600 | 30 |
| 3 | BLAKE | 25-10-1990 | SALESMAN | 7698 | 26-03-2016 | 1250 | 30 |
| 4 | CLARK | 23-09-1991 | MANAGER | 7839 | 31-10-2015 | 2975 | 20 |
| 5 | SCOTT | 12-01-1992 | MANAGER | 7839 | 11-06-2016 | 2850 | 10 |
| 6 | KING | 05-08-1988 | ANALYST | 7839 | 14-05-2017 | 2450 | 10 |
| 7 | TURNER | 17-09-1988 | PRESIDENT | 7566 | 05-05-2017 | 3000 | 20 |

A instrução SQL abaixo exclui todos os registros da tabela de funcionários

DELETE FROM Teradatapoint.tbl\_employee;

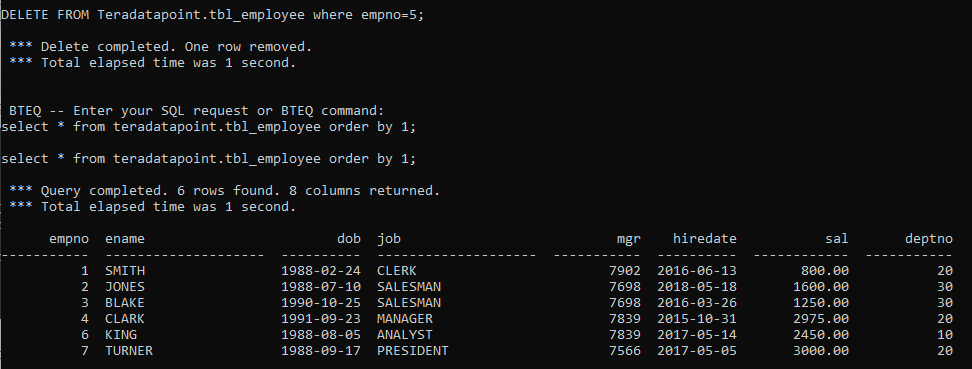
Você pode usar a instrução select para verificar se todos os registros foram excluídos.



A instrução abaixo exclui apenas o registro do funcionário cujo empno é 5.

DELETE FROM Teradatapoint.tbl\_employee

where empno=5;



**Teradata Order By Clause**

A cláusula Teradata Order By classifica o conjunto de resultados com base na expressão fornecida. É usado com a instrução SELECT para determinar como o conjunto de resultados será selecionado. Se a cláusula order by não for usada, o conjunto de resultados permanecerá não classificado.

Sintaxe da cláusula ORDER BY do Teradata

SELECT column1, column2, ......,columnN

FROM DatabaseName.TableName

ORDER BY expression [ NULL First | NULL Last ];

Aqui,

expressão pode ser de três tipos. Eles são:

* expressão como nome\_da\_coluna – O nome da coluna pela qual você deseja classificar o conjunto de resultados.
* expression as column\_name\_alias – o nome alternativo da coluna pela qual você deseja classificar o conjunto de resultados.
* expressão como posição\_da\_coluna – Você pode especificar a posição da coluna na lista de seleção.
* ASC|DESC – O conjunto de resultados pode ser ordenado em ordem crescente ou decrescente.
* NULL First – Os resultados nulos devem ser listados primeiro.
* NULL Last – Os resultados nulos devem ser listados por último.

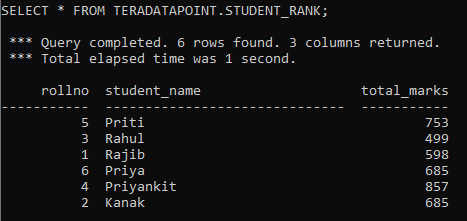
Teradata ORDER BY exemplo

Suponha que tenhamos uma tabela de alunos armazenada em nosso banco de dados Teradata.

Agora, se você executar a instrução select na tabela, obterá a seguinte saída:

SELECT \* FROM TERADATAPOINT.STUDENT\_RANK;

**Output:**

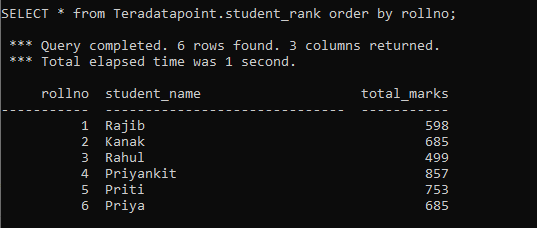


Aqui você pode ver que o conjunto de resultados não está classificado.

Se você deseja classificar o conjunto de resultados em rollno, pode usar a cláusula ORDER BY conforme abaixo.

SELECT \* FROM TERADATAPOINT.STUDENT\_RANK order by rollno;

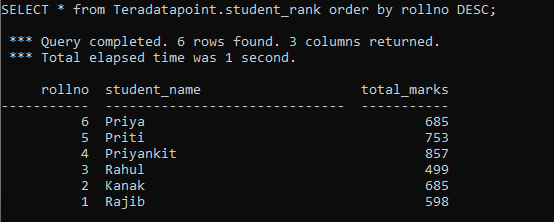
**Output:**



Por padrão, o Teradata classificará as linhas em ordem crescente.

Você também pode classificar o conjunto de resultados em ordem decrescente usando a palavra-chave DESC.

SELECT \* FROM TERADATAPOINT.STUDENT\_RANK order by rollno DESC;



**Teradata GROUP BY Clause**

A cláusula Teradata Group By cria o grupo de linhas relacionadas. A cláusula group by identifica as linhas que têm o mesmo valor para o atributo especificado (ou seja, valor duplicado) e retorna uma única linha de informações em vez de todas as linhas em que o atributo tem o mesmo valor.

Isso pode ser feito especificando uma ou mais colunas em uma tabela como coluna(s) de agrupamento.

Sintaxe Teradata GROUP BY:

Select column\_name

from DatabaseName.TableName

Group By column\_name;

Com Função Agregada

Select column\_name , sum(column\_2)

from DatabaseName.TableName

Group By column\_name;

Observe que column\_name pode ser usado com a função agregada somente se aparecer na cláusula GROUP BY.

Exemplo de cláusula Teradata GROUP BY

Considere a seguinte tabela employee:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| emp\_id | emp\_name | emp\_phone | emp\_gender | department |
| 101 | Kalyan Roy | 9620139678 | M | HR |
| 102 | Rajesh Sharma | 9611895588 | M | ADMIN |
| 103 | Rupali Sharma | 8884692570 | F | SALES |
| 104 | Dipankar Sen | 9957889640 | M | HR |
| 105 | Sunitha Rai | 9742067708 | F | SALES |
| 106 | Parag Barman | 8254066054 | M | MARKETING |
| 107 | Vinitha Sharma | 9435746645 | F | ADMIN |
| 108 | Abhishek Saha | 9850157207 | M | SALES |
| 109 | Rushang Desai | 9850157207 | M | SALES |
| 110 | Arvin Kumar | 8892340054 | M | ADMIN |

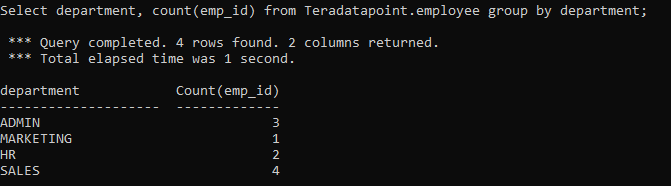
A consulta abaixo encontra o número de funcionários nos diferentes departamentos.

Select department, count(emp\_id)

from Teradatapoint.employee

group by department;

**Output:**



**Teradata HAVING Clause**

A cláusula HAVING geralmente é usada junto com a cláusula GROUP BY para aplicar mais filtros no conjunto de resultados.

Teradata HAVING exemplo

A instrução SQL a seguir lista o número de funcionários em cada departamento. Incluir apenas os departamentos com mais de 3 funcionários.

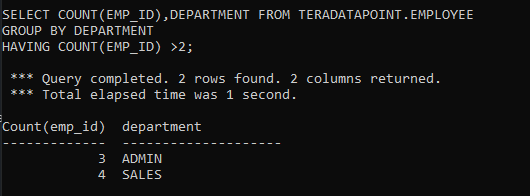
SELECT COUNT(EMP\_ID),DEPARTMENT

FROM TERADATAPOINT.EMPLOYEE

GROUP BY DEPARTMENT

HAVING COUNT(EMP\_ID) >2;

**Output:**



Observação:

A cláusula GROUP BY não pode ser usada na cláusula WHERE. Portanto, TER é útil.

Uma instrução SQL pode ter as cláusulas WHERE e HAVING.

WHERE – filtra os dados antes de agrupar.

HAVING – filtra os dados após o agrupamento.

**Alias in Teradata**

Na instrução select em geral, vimos selecionar as diferentes colunas de uma tabela. Como resultado da instrução select, a saída é gerada e a primeira linha do conjunto de resultados são os nomes das colunas que mencionamos na instrução select. Às vezes, o nome da coluna parece estranho e não é fácil de ler. Para torná-lo fácil de ler, podemos usar o Teradata Alias

Além do conjunto de resultados, um alias torna o SQL mais fácil de escrever. O novo nome de alias pode ser usado em qualquer lugar na instrução SQL. Podemos dar um nome de alias a qualquer outro objeto, como tabelas, visualizações para facilitar a gravação de consultas de junção.

Alias ​​na sintaxe do Teradata:

select column1 alias1, column2 alias2,...,columnN aliasN

from DatabaseName.TableName;

OU

select column1 as alias1, column2 as alias2,...,columnN as aliasN

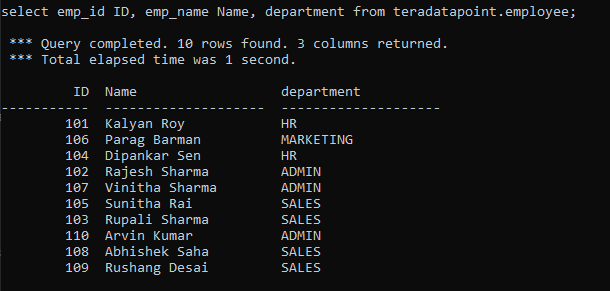
from DatabaseName.TableName;

**Alias no Teradata examplo:**

select emp\_id ID, emp\_name Name, department

from teradatapoint.employee;

**Output:**



**Teradata Logical Operators**

Abaixo estão os operadores lógicos usados ​​no Teradata.

|  |  |
| --- | --- |
| Operator | Meaning |
| = | Igual a |
| <> | Diferente de |
| > | Maior que |
| < | Menor que |
| > = | Maior ou igual a |
| < = | Menos ou igual a |
| AND | Combina várias condições. Avalia como verdadeiro somente se todas as condições forem atendidas. |
| OR | Combina várias condições. Avalia como verdadeiro se qualquer uma das condições for atendida. |
| NOT | Inverte o significado da condição |
| BETWEEN | Se os valores com intervalo entre |
| IN | Se os valores em <expressão> |
| NOT IN | Se os valores NÃO estão em <expressão> |
| IS NULL | Se o valor for NULO |
| IS NOT NULL | Se o valor NÃO for NULO |

Teradata condição AND

AND é usado para conectar diferentes operadores LÓGICOS. Todas as condições devem ser atendidas para que uma linha específica seja qualificada.

Exemplo:

Considere a tabela de alunos abaixo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| roll\_no | first\_name | last\_name | dept\_no | dept\_name | DOB |
| 101 | Amal | Roy | 1 | Computer Science | 10/12/1990 |
| 104 | Surajit | Saha | 3 | Physics | 8/17/1989 |
| 102 | Sagar | Sharma | 2 | Information Technology | 9/25/1991 |
| 105 | Bidyut | Dey | 5 | Electrical | 12/5/1985 |
| 103 | Aditya | Roy | 7 | Electronics | 7/12/1991 |

O exemplo a seguir busca os alunos cujo primeiro nome é Aditya e o sobrenome é Roy.

**sel** \* **from** teradatapoint.student

**where** first\_name='Aditya'

**AND** last\_name='Roy'**;**

Abaixo está a saída da consulta acima.

\*\*\* Query completed. One row found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

    roll\_no  first\_name  last\_name   dept\_no  dept\_name          DOB

        -----------  ----------  ----------  -------  ----------  ----------

        103  Aditya      Roy               6  Electronic  1991-07-12

Teradata OR Condição

A condição OR é usada para conectar diferentes operadores LÓGICOS. Somente uma das condições precisa ser verdadeira para que uma linha seja qualificada.

Exemplo:

Os exemplos a seguir recuperam os alunos cujo primeiro nome é Aditya OU o sobrenome é Roy.

**sel** \* **from** teradatapoint.student

**where** first\_name = 'Aditya'

**OR** last\_name='Roy'**;**

Abaixo está a saída da consulta acima

\*\*\* Query completed. 2 rows found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

    roll\_no  first\_name  last\_name   dept\_no  dept\_name          DOB

-----------  ----------  ----------  -------  ----------  ----------

        101  Amal        Roy               2  Computer Science   1990-10-12

        103  Aditya      Roy               6  Electronic  1991-07-12

Teradata BETWEEN Condição

A condição BETWEEN é usada para filtrar as linhas selecionadas para um intervalo específico.

Exemplo:

O exemplo a seguir filtra os alunos se o Roll\_no estiver entre 102 e 104.

**sel** \* **from** teradatapoint.student **where** roll\_no between 102 and 104**;**

Abaixo está a saída da consulta acima:

\*\*\* Query completed. 3 rows found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

    roll\_no  first\_name  last\_name   dept\_no  dept\_name          DOB

-----------  ----------  ----------  -------  ----------  ----------

        104  Surajit     Saha              8  Physics     1989-08-17

        102  Sagar       Sharma            2  Information Technology  1991-09-25

        103  Aditya      Roy               6  Electronic  1991-07-12

Teradata IN Condição

A condição IN é usada para verificar o valor em uma determinada lista de valores.

Exemplo:

O exemplo a seguir busca registros com roll\_no 104 e 105.

**sel** \* **from** teradatapoint.student **where** roll\_no **IN** (104,105)**;**

Abaixo está o resultado da consulta acima:

\*\*\* Query completed. 2 rows found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

    roll\_no  first\_name  last\_name   dept\_no  dept\_name          DOB

-----------  ----------  ----------  -------  ----------  ----------

        104  Surajit     Saha              8  Physics     1989-08-17

        105  Bidyut      Dey               7  Electrical  1985-12-05

Teradata NOT IN Condição

O comando NOT IN reverte o resultado do comando IN. Ele retornará o resultado que NÃO ESTÁ NA lista de valores fornecida.

Exemplo:

O exemplo a seguir retorna os registros, exceto roll\_no 104 e 105:

**sel** \* **from** teradatapoint.student **where** roll\_no **NOT** **IN** (104,105)**;**

Abaixo está a saída da consulta acima:

 \*\*\* Query completed. 3 rows found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

    roll\_no  first\_name  last\_name   dept\_no  dept\_name          DOB

-----------  ----------  ----------  -------  ----------  ----------

        101  Amal        Roy               2  Computer S  1990-10-12

        102  Sagar       Sharma            2  Information Technology  1991-09-25

        103  Aditya      Roy               6  Electronic  1991-07-12

**Teradata SET operators**

Os operadores SET no Teradata manipulam os conjuntos de resultados de duas ou mais consultas combinando os resultados de cada consulta individual em um único conjunto de resultados.

Os operadores SET mais comuns são os seguintes:

|  |  |
| --- | --- |
| Operator | Description |
| UNION | Combina os conjuntos de resultados de duas ou mais instruções SELECT. Linhas duplicadas são eliminadas. |
| UNION ALL | Combina os conjuntos de resultados de duas ou mais instruções SELECT. Linhas duplicadas não são eliminadas. |
| INTERSECT | Selecione apenas as linhas que aparecem em todos os conjuntos de resultados gerados pela instrução SELECT individual. |
| MINUS/EXCEPT | Retorna apenas as linhas dos primeiros conjuntos de resultados, mas não do segundo conjunto de resultados. |

Regras do operador Teradata SET

* Os números da coluna em cada instrução SELECT devem ser os mesmos.
* O tipo de dados para cada coluna deve ser correspondido ou compatível.
* ORDER BY, se usado, deve ser incluído apenas na instrução SELECT final.
* Aliasing, Title, Format devem ser usados ​​na primeira instrução SELECT, outros serão ignorados.

Teradata UNION

Um operador UNION combina os conjuntos de resultados de duas ou mais instruções SELECT juntas como um único conjunto de resultados. Isso removerá as linhas duplicadas.

Exemplo:

Considere abaixo duas tabelas de clientes relacionadas a clientes de poupança e conta corrente.

Cliente de poupança:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| cust\_id | income | age | years\_with\_bank | nbr\_children | gender | marital\_status |
| 1363355 | 13087 | 22 | 1 | 0 | M | 2 |
| 1363329 | 0 | 6 | 0 | 0 | F | 1 |
| 1362672 | 16319 | 50 | 0 | 1 | F | 2 |
| 1362629 | 0 | 6 | 0 | 0 | M | 1 |
| 1362611 | 24115 | 48 | 8 | 1 | F | 2 |

Cliente de conta corrente:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| cust\_id | income | age | years\_with\_bank | nbr\_children | gender | marital\_status |
| 1363355 | 13087 | 22 | 1 | 0 | M | 2 |
| 1362629 | 0 | 6 | 0 | 0 | M | 1 |
| 1363056 | 2442 | 19 | 1 | 0 | F | 1 |

Agora, usando a consulta abaixo, podemos todos os clientes de poupança e conta corrente.

**sel** cust\_id **from** teradatapoint.customer\_current

**UNION**

**sel** cust\_id **from** teradatapoint.customer\_savings**;**

Quando executada, a consulta acima produz o resultado abaixo. Observe que apenas o cust\_id exclusivo foi retornado.

  cust\_id

-----------

    1362611

    1362629

    1362672

    1363056

    1363329

    1363355

Teradata UNION ALL

Este também funciona da mesma forma que UNION, exceto que não eliminará a duplicata do conjunto de resultados final.

Exemplo:

Retorna todas as contas de poupança e conta corrente do cliente.

**sel** cust\_id **from** teradatapoint.customer\_current

**UNION**  **ALL**

**sel** cust\_id **from** teradatapoint.customer\_savings**;**

Quando executada, a consulta acima produz o resultado abaixo.

\*\*\* Query completed. 8 rows found. One column returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id

-----------

    1363355

    1363355

    1362629

    1362672

    1363056

    1363329

    1362629

    1362611

Teradata INTERSECT

Ele retorna apenas as linhas que são comuns para todos os conjuntos de resultados retornados pela instrução SELECT individual.

Exemplo:

O exemplo a seguir retorna apenas o ID do cliente que possui tanto a poupança quanto a conta corrente no banco.

**sel** cust\_id **from** teradatapoint.customer\_current

**INTERSECT**

**sel** cust\_id **from** teradatapoint.customer\_savings**;**

Quando executada, a consulta acima produz o resultado abaixo:

\*\*\* Query completed. 2 rows found. One column returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id

-----------

    1362629

    1363355

Teradata MINUS/EXCEPT

O operador MINUS/EXCEPT retorna apenas as linhas que estão presentes no primeiro conjunto de resultados, mas não no segundo.

Exemplo:

O exemplo a seguir busca o ID do cliente que possui apenas uma conta poupança, mas não uma conta corrente.

**sel** cust\_id **from** teradatapoint.customer\_savings

**MINUS**

**sel** cust\_id **from** teradatapoint.customer\_current**;**

Quando executada, a consulta acima produz o resultado abaixo.

\*\*\* Query completed. 3 rows found. One column returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id

-----------

    1362672

    1363329

    1362611

**BETWEEN in Teradata**

O operador BETWEEN no Teradata seleciona um intervalo de dados entre dois valores. Isso geralmente é usado com a cláusula WHERE para filtrar os dados no conjunto de resultados.

Teradata BETWEEN Sintaxe

WHERE expression1 [NOT] BETWEEN expression2 AND expression3;

Aqui,

expression1 – O nome da coluna ou alias na instrução select para a qual a filtragem deve ser feita.

NOT – Se você não quiser selecionar os dados entre 2 intervalos.

expression2 – O intervalo inicial para o qual o teste precisa ser feito.

expression3 – O intervalo final para o qual o teste precisa ser feito.

Se o teste BETWEEN falhar, nenhuma linha retornará.

Exemplo:

Considere a seguinte tabela employee.

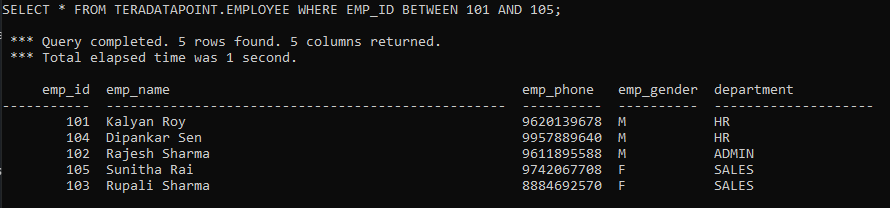
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| emp\_id | emp\_name | emp\_phone | emp\_gender | department |
| 101 | Kalyan Roy | 9620139678 | M | HR |
| 102 | Rajesh Sharma | 9611895588 | M | ADMIN |
| 103 | Rupali Sharma | 8884692570 | F | SALES |
| 104 | Dipankar Sen | 9957889640 | M | HR |
| 105 | Sunitha Rai | 9742067708 | F | SALES |
| 106 | Parag Barman | 8254066054 | M | MARKETING |
| 107 | Vinitha Sharma | 9435746645 | F | ADMIN |
| 108 | Abhishek Saha | 9850157207 | M | SALES |
| 109 | Rushang Desai | 9850157207 | M | SALES |
| 110 | Arvin Kumar | 8892340054 | M | ADMIN |

A consulta abaixo retorna os detalhes do funcionário cujo ID está entre 101 e 105.

SELECT \* FROM TERADATAPOINT.EMPLOYEE

WHERE EMP\_ID BETWEEN 101 AND 105;

**Output:**

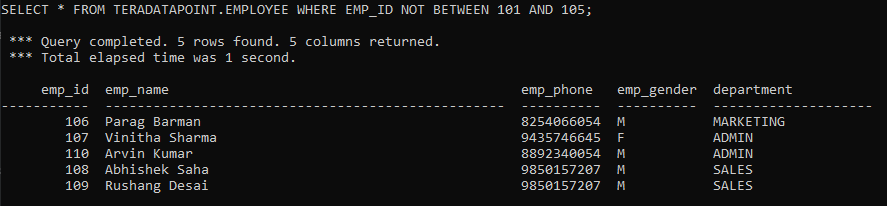


Teradata NOT BETWEEN Exemplo

O exemplo abaixo mostra o oposto do funcionário acima, ou seja, isso não selecionará employee\_id entre 101 e 105.

SELECT \* FROM TERADATAPOINT.EMPLOYEE

WHERE EMP\_ID NOT BETWEEN 101 AND 105;



**LIKE in Teradata**

O operador Teradata LIKE é usado para correspondência de padrão na coluna especificada. Isso é usado na cláusula WHERE e pode usar caracteres curinga e caracteres especiais para corresponder a um padrão de uma palavra. O operador Teradata LIKE pode ser usado em qualquer instrução SQL válida, como instrução SELECT, instrução INSERT, UPDATE ou DELETE no Teradata.

A sintaxe do operador Teradata LIKE é a seguinte:

SELECT \* FROM DatabaseNAME.TableName

WHERE columnName [NOT] LIKE pattern\_to\_match [Escape escape\_character]

Exemplo de operador LIKE do Teradata

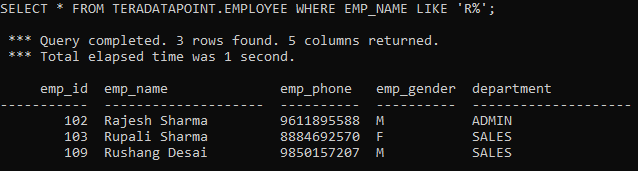
Considere a seguinte tabela de empregados.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| emp\_id | emp\_name | emp\_phone | emp\_gender | department |
| 101 | Kalyan Roy | 9620139678 | M | HR |
| 102 | Rajesh Sharma | 9611895588 | M | ADMIN |
| 103 | Rupali Sharma | 8884692570 | F | SALES |
| 104 | Dipankar Sen | 9957889640 | M | HR |
| 105 | Sunitha Rai | 9742067708 | F | SALES |
| 106 | Parag Barman | 8254066054 | M | MARKETING |
| 107 | Vinitha Sharma | 9435746645 | F | ADMIN |
| 108 | Abhishek Saha | 9850157207 | M | SALES |
| 109 | Rushang Desai | 9850157207 | M | SALES |
| 110 | Arvin Kumar | 8892340054 | M | ADMIN |

Agora vamos filtrar o nome do funcionário cujo nome começa com 'R'.

SELECT \* FROM TERADATAPOINT.EMPLOYEE

WHERE EMP\_NAME LIKE 'R%';

**OUTPUT:**

**Describe Table in Teradata**

DESCRIBE TABLE mostra as informações relacionadas às colunas na maioria dos RDBMS populares, como ORACLE, MySQL, etc. Não há nenhum comando DESCRIBE tabela disponível no Teradata.

O Teradata fornece duas formas alternativas de visualizar as informações relacionadas às colunas. Eles são como abaixo.

HELP TABLE - SHOW TABLE

Abaixo estão as sintaxes do Teradata HELP TABLE e SHOW TABLE:

HELP TABLE DatabaseName.TableName;

SHOW TABLE DatabaseName.TableName;

Onde,

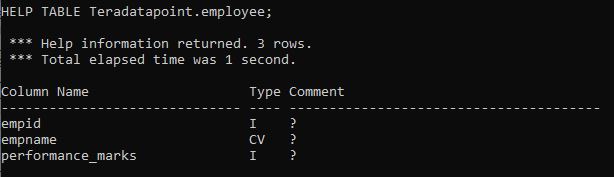
* DatabaseName – o nome do banco de dados ao qual a tabela pertence.
* TableName – o nome da tabela cujas informações você deseja verificar.

HELP TABLE retorna apenas as definições de coluna, enquanto o comando SHOW TABLE mostra a instrução de criação de tabela completa.

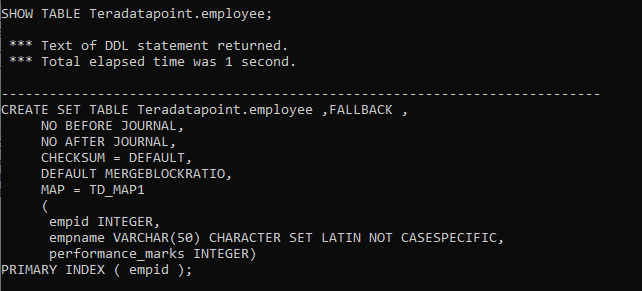
Os exemplos abaixo mostram as informações sobre a tabela de funcionários.

HELP TABLE Teradatapoint.employee;

**Output:**



**Output:**



**Teradata Count Distinct**

A função de contagem retorna o número total de linhas qualificadas na expressão de valor. Contagem Teradata distinta retorna o número total de linhas distintas.

Sintaxe Distinta de Contagem de Teradata

COUNT ( [TYPE] value\_expression );

Aqui,

* value\_expression – Uma expressão literal ou de coluna para a qual a contagem total é calculada. A expressão não pode conter funções analíticas ou agregadas ordenadas.
* TYPE as ALL – Todos os valores de value\_express que não são nulos, incluindo duplicados, são incluídos no cálculo.
* TYPE as DISTINCT – exclui duplicatas especificadas por value\_expression da expressão. Isso é usado para obter a contagem DISTINCT no Teradata.
* TYPE as \* – Conta todas as linhas no grupo de linhas em que a contagem opera.

Exemplo distinto de contagem de Teradata

Considere a seguinte tabela de empregados:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| emp\_id | emp\_name | emp\_phone | emp\_gender | department |
| 101 | Kalyan Roy | 9620139678 | M | HR |
| 102 | Rajesh Sharma | 9611895588 | M | ADMIN |
| 103 | Rupali Sharma | 8884692570 | F | SALES |
| 104 | Dipankar Sen | 9957889640 | M | HR |
| 105 | Sunitha Rai | 9742067708 | F | SALES |
| 106 | Parag Barman | 8254066054 | M | MARKETING |
| 107 | Vinitha Sharma | 9435746645 | F | ADMIN |
| 108 | Abhishek Saha | 9850157207 | M | SALES |
| 109 | Rushang Desai | 9850157207 | M | SALES |
| 110 | Arvin Kumar | 8892340054 | M | ADMIN |

Você pode usar count(distinct) para descobrir o número de departamentos distintos na tabela de funcionários conforme abaixo.

SELECT COUNT(distinct department)

FROM Teradatapoint.employee;

\*\*\* Query completed. One row found. One column returned.

\*\*\* Total elapsed time was 1 second.

Count(Distinct(department))

---------------------------

4

Preparação Da Table

CREATE TABLE Teradatapoint.employee

(

emp\_id integer,

emp\_name varchar(50),

emp\_phone varchar(10),

emp\_gender char(1),

department varchar(20)

)primary index(emp\_id);

insert into teradatapoint.employee values (101,'Kalyan Roy','9620139678','M','HR');

insert into teradatapoint.employee values (102,'Rajesh Sharma','9611895588','M','ADMIN');

insert into teradatapoint.employee values (103,'Rupali Sharma','8884692570','F','SALES');

insert into teradatapoint.employee values (104,'Dipankar Sen','9957889640','M','HR');

insert into teradatapoint.employee values (105,'Sunitha Rai','9742067708','F','SALES');

insert into teradatapoint.employee values (106,'Parag Barman','8254066054','M','MAKETING');

insert into teradatapoint.employee values (107,'Vinitha Sharma','9435746645','F','ADMIN');

insert into teradatapoint.employee values (108,'Abhishek Saha','9850157207','M','SALES');

insert into teradatapoint.employee values (109,'Rushang Desai','9850157207','M','SALES');

insert into teradatapoint.employee values (110,'Arvin Kumar','8892340054','M','ADMIN');

**CASE and COALESCE Statement in Teradata**

Declaração CASE no Teradata

A instrução Teradata CASE fornece a flexibilidade para buscar valores alternativos para uma base de coluna na condição especificada na expressão. Não está conseguindo? Ok, deixe-me explicar com um exemplo simples.

Exemplo de Declaração CASE Teradata

Considere a tabela de clientes abaixo.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| cust\_id | income | age | years\_with\_bank | nbr\_children | gender | marital\_status |
| 1,362,487 | 6,605 | 71 | 1 | 0 | M | 2 |
| 1,362,956 | 0 | 10 | 4 | 0 | F | 1 |
| 1,362,752 | 47,668 | 54 | 3 | 0 | F | 1 |
| 1,363,221 | 0 | 10 | 4 | 0 | F | 1 |
| 1,362,813 | 15,684 | 53 | 3 | 2 | F | 2 |
| 1,362,609 | 1,929 | 79 | 8 | 0 | F | 2 |

Agora, você pode escrever alguma instrução CASE em marital\_status conforme abaixo.

**SELECT**

cust\_id,

income,

age,

years\_with\_bank,

nbr\_children, gender,

**CASE** **WHEN** marital\_status=1 **THEN** 'Single'

**WHEN** marital\_status=2 **THEN** 'Married'

**ELSE** 'Not Sure'

**END** **AS** marital\_status

**FROM** teradatapoint.customer;

Se o marital\_status for 1, retornará como Solteiro e se o marital\_status for 2, esta consulta retornará como Casado.

Abaixo está a saída da consulta acima

\*\*\* Query completed. 6 rows found. 7 columns returned.

 \*\*\* Total elapsed time was 1 second.

cust\_id       income     age  years\_with\_bank  nbr\_children  gender  marital\_status

-----------  ----------- ------  ------------  ------------  ------  --------------

1362487         6605      71             1             0  M       Married

1362956            0      10                4             0  F       Single

1362752        47668      54                3             0  F       Single

1363221            0      10                4             0  F       Single

1362813        15684      53                3             2  F       Married

1362609         1929      79                8             0  F       Married

Também podemos escrever a consulta acima de outra maneira.

**SELECT**    cust\_id,

income,

age,

years\_with\_bank,

nbr\_children,

gender,

**CASE** marital\_status **WHEN** 1 **THEN** 'Single'

**WHEN** 2 **THEN** 'Married

'**ELSE** 'Not Sure'**END** **AS** marital\_status

**FROM** teradatapoint.customer**;**

COALESCE em Teradata

COALESCE em Teradata é usado para manipulação de NULL. O COALESCE retorna o primeiro valor NOT NULL encontrado na lista fornecida. Caso contrário, ele retornará NULL se todos os argumentos na lista de expressões forem avaliados como NULL.

A sintaxe básica da função COALESCE é a seguinte:

COALESCE(expressão1, expressão2 [,lista de expressões])

Exemplo de Função COALESCE

Considere a tabela de funcionários abaixo.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| emp\_no | first\_name | last\_name | home\_no | ph\_no1 | ph\_no2 |
| 345 | Amal | Roy | ? | ? | ? |
| 123 | Alex | Martin | 45637887 | 7209756747 | ? |
| 134 | Sager | Sharma | 34567548 | ? | ? |

Agora, você pode priorizar qual número de telefone selecionar usando a função COALESCE conforme abaixo.

SELECT

emp\_no,

first\_name,

last\_name,

COALESCE(ph\_no1,home\_no,ph\_no2,'Phone# not available') AS Phone\_No

FROM teradatapoint.employee;

A consulta acima procurará ph\_no1 primeiro. Se for NULL , ele procurará home\_no e ph\_no2 respectivamente. Se nenhum dos argumentos estiver retornando um valor não nulo, ele retornará o valor padrão.

\*\*\* Query completed. 3 rows found. 4 columns returned.

 \*\*\* Total elapsed time was 1 second.

     emp\_no  first\_name  last\_name   Phone\_No

-----------  ----------  ----------  --------------------

        345  Amal        Roy         Phone# not available

        123  Alex        Martin                7209756747

        134  Sager       Sharma                  34567548

**Teradata Aggregate Function**

As funções Agregadas preparam o resumo dos valores de uma ou mais tabelas. O nível de detalhamento das informações será perdido após realizar a agregação em algumas tabelas específicas.

O Teradata oferece suporte à função agregada mais comum, conforme abaixo.

|  |  |
| --- | --- |
| Function Name | Detail |
| COUNT | Fornece o número total de linhas das tabelas. |
| SUM | Produz a soma de todos os valores para a(s) coluna(s) específica(s) |
| MIN | Retorna o valor mínimo de uma coluna específica |
| MAX | Retorna o valor máximo de uma coluna específica |
| AVG | Retorna o valor médio da coluna específica |

Exemplo de função agregada

Considere a tabela de clientes abaixo:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| cust\_id | income | age | years\_with\_bank | nbr\_children | gender | marital\_status |
| 1363355 | 13087 | 22 | 1 | 0 | M | 2 |
| 1363329 | 0 | 6 | 0 | 0 | F | 1 |
| 1362672 | 16319 | 50 | 0 | 1 | F | 2 |
| 1362629 | 0 | 6 | 0 | 0 | M | 1 |
| 1362611 | 24115 | 48 | 8 | 1 | F | 2 |

Função CONTAGEM

COUNT fornecerá o número total de registros presentes na tabela.

A consulta a seguir retornará o número de registros da tabela de clientes.

**sel** count(\*) **from** teradatapoint.customer;

Quando executado, produz o seguinte resultado:

\*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

   Count(\*)

-----------

          5

Função SOMA

SUM fornece o somatório para uma coluna específica.

Exemplo de função SOMA

Suponha que queremos obter a soma das receitas de todos os clientes.

sel sum(income) from teradatapoint.customer;

Quando executado, produz o seguinte resultado:

\*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Sum(income)

-----------

      53521

Função MIN

MIN retorna o valor mínimo de uma coluna específica.

Exemplo de função MIN

Suponha que queremos obter a idade mínima do cliente.

**SELECT** MIN(age) **FROM** teradatapoint.customer;

Quando executado, produz o seguinte resultado:

\*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Minimum(age)

------------

           6

Função MAX

MAX retorna o valor máximo de uma coluna específica.

Exemplo de função MAX

**SELECT** MAX(age) **FROM** teradatapoint.customer;

Quando executado, produz o seguinte resultado:

\*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Maximum(age)

------------

          50

Função AVERAGE/ AVG

AVG retorna o valor médio de uma coluna específica.

Exemplo de Função AVERAGE

Suponha que queremos obter a idade média do cliente.

**SELECT** AVG(age) **FROM** teradatapoint.customer;

Quando executado, produz o seguinte resultado:

\*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Average(age)

------------

          26

**Teradata String Function**

O Teradata oferece suporte a várias funções de string para manipular a string.

Função UPPER E LOWER

As funções UPPER e LOWER cobrem todos os valores da coluna de caracteres em maiúsculas e minúsculas, respectivamente. UPPER e LOWER são compatíveis com ANSI.

Sintaxe

UPPER (expressão) –> retorna a expressão em letras maiúsculas

LOWER (expressão) –> retorna a expressão em letras minúsculas

Exemplo:

O exemplo a seguir converterá a string “teradatapoint” em letras maiúsculas.

SELECT UPPER('teradatapoint');

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Upper('teradatapoint')

----------------------

TERADATAPOINT

O exemplo a seguir converterá a string “TERADATAPOINT” para letras minúsculas.

SELECT LOWER('TERADATAPOINT');

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Lower('TERADATAPOINT')

----------------------

teradatapoint

Função CHARACTER\_LENGTH

A função CHARACTER\_LENGTH retorna os números de caracteres da expressão string de caracteres.O resultado será um número inteiro que representa o comprimento.

O resultado será o mesmo para caracteres de comprimento fixo.

O resultado irá variar para caracteres de comprimento variável.

Os espaços são caracteres válidos, portanto, o comprimento será contado para o espaço.

Sintaxe

*CHARACTER\_LENGTH ( expression )*

O exemplo a seguir retornará o número de caracteres da string “TERADATAPOINT”.

SELECT CHARACTER\_LENGTH('TERADATAPOINT');

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Characters('TERADATAPOINT')

---------------------------

                         13

Função TRIM

A função TRIM é usada para remover o espaço de um conjunto específico de início ou fim ou ambos de uma expressão. Por padrão, ele remove espaço de ambos. TRIM é o padrão ANSI.

Sintaxe

TRIM ( [ LEADING | BOTH | TRAILING ] [ trim\_character] FROM expressão)

Exemplo

O exemplo a seguir remove o espaço de ambas as extremidades da string.

SELECT TRIM('TERADATAPOINT');

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Trim(BOTH FROM 'TERADATAPOINT')

-------------------------------

TERADATAPOINT

Função de POSIÇÃO

A função POSITION é usada para retornar a posição de uma string dentro de outra. Apenas a posição da primeira ocorrência da string é retornada.

Sintaxe

POSIÇÃO (expressão1 IN expressão2)

Exemplo

O exemplo a seguir retornará a primeira ocorrência de point na string “teradatapoint”.

SELECT POSITION('POINT' IN 'TERADATAPOINT');

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Position('POINT' in 'TERADATAPOINT')

------------------------------------

                                   9

Função SUBSTRING

O SUBSTRING é usado para extrair posicionalmente o texto de outro valor de dados. SUBSTRING é o padrão ANSI.

Sintaxe

SUBSTRING (expressão1 FROM n1 [para n2])

Ele retorna uma substring de expression1, começando na posição n1, por um comprimento de n2 (se presente) ou até o final da string (se não presente).

Exemplo

O exemplo a seguir retorna o caractere da 5ª posição para 4 caracteres.

SELECT SUBSTRING('TERADATAPOINT' FROM 5 FOR 4);

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Substring('TERADATAPOINT' From 5 For 4)

---------------------------------------

DATA

**Teradata Date/Time Function**

A função de data e hora é uma das funções mais importantes do Teradata. Portanto, precisamos entender o conceito com muita clareza.

Palavras-chave relacionadas à Data e Hora:

SELECT DATE AS "DATE"

,CURRENT\_DATE AS "ANSI DATE"

,TIME AS "TIME"

,CURRENT\_TIME AS "ANSI TIME"

,CURRENT\_TIMESTAMP(0) "TIMESTAMP"

,CURRENT\_TIMESTAMP(6) "TIMESTAMP WITH SIX PRECISION SECOND";

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DATE | ANSI DATE | TIME | ANSI TIME | TIMESTAMP | TIMESTAMP WITH SIX PRECISION SECOND |
| 9/21/2016 | 9/21/2016 | 07:06:35 | 07:06:35 | 9/21/2016 07:06:35 | 9/21/2016 07:06:35.950000 |

Como a Teradata armazena as datas internamente?

Teradata armazena datas como INTEGER usando a fórmula abaixo.

((Year – 1900)\*10000) + (Month\*100) + Day

Portanto, a data 2016-09-21 será armazenada internamente conforme abaixo.

(2016-1900)\*10000 + (9\*100) + 21 = 1160921

Você pode verificar isso usando a seguinte consulta:

**SELECT CAST**(DATE'2016-09-21' **AS**INTEGER)**;**

\*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

 2016-09-21

-----------

    1160921

Alterando o DATEFORM no BTEQ

Você pode alterar a exibição do formulário de data usando DATEFORM. Existem 2 tipos de formulário para exibir a data. Eles são INTEGERDATE e ANSIDATE. O padrão é INTEGERDATE.

INTEGERDATE -> AA/MM/DD

ANSIDATO -> AAAA-MM-DD

set session dateform=ANSIDATE;

 \*\*\* Set SESSION accepted.

 \*\*\* Total elapsed time was 1 second.

 BTEQ -- Enter your SQL request or BTEQ command:

SELECT DATE;

SELECT DATE;

 \*\*\* Query completed. One row found. One column returned.

 \*\*\* Total elapsed time was 1 second.

Current Date

------------

  2016-09-21

Como o Teradata armazena datas como INTEGER, podemos realizar algumas operações aritméticas na data. O Teradata fornece algumas funções para executar essas operações.

Adicionar ou subtrair dias de uma DATE

SELECT CURRENT\_DATE, CURRENT\_DATE-30 AS "OLD DATE", CURRENT\_DATE+30" AS FUTURE DATE";

\*\*\* Query completed. One row found. 3 columns returned.

 \*\*\* Total elapsed time was 1 second.

Current Date    OLD DATE  FUTURE DATE

------------  ----------  -----------

  2016-09-21  2016-08-22   2016-10-21

Comando ADD MONTHS

Você pode adicionar um mês ou muitos meses à sua coluna de data ou carimbo de data/hora.

SELECT CURRENT\_DATE, ADD\_MONTHS(CURRENT\_DATE,-2) AS "OLD MONTH", ADD\_MONTHS(CURRENT\_DATE,2) AS "FUTURE MONTH";

\*\*\* Query completed. One row found. 3 columns returned.

 \*\* Total elapsed time was 1 second.

Current Date   OLD MONTH  FUTURE MONTH

------------  ----------  ------------

  2016-09-21  2016-07-21    2016-11-21

O comando EXTRACT

O comando EXTRACT extrai parte da data, como ano, mês e dia, de data e hora, minuto e segundo do valor TIME/TIMESTAMP.

Exemplo

Seguem exemplos de como extrair ano, mês e data do valor de data e hora, minuto e segundo do valor TIMESTAMP.

SELECT EXTRACT(YEAR from DATE);

EXTRACT(YEAR FROM Current Date)

-------------------------------

                          2016

SELECT EXTRACT(MONTH from DATE);

EXTRACT(MONTH FROM Current Date)

--------------------------------

                               9

SELECT EXTRACT(DAY from DATE);

EXTRACT(DAY FROM Current Date)

------------------------------

                            21

SELECT EXTRACT(HOUR from CURRENT\_TIMESTAMP(6));

EXTRACT(HOUR FROM Current TimeStamp(6))

---------------------------------------

                                      9

SELECT EXTRACT(MINUTE from CURRENT\_TIMESTAMP(6));

EXTRACT(MINUTE FROM Current TimeStamp(6))

-----------------------------------------

                                        7

SELECT EXTRACT(SECOND from CURRENT\_TIMESTAMP(6));

EXTRACT(SECOND FROM Current TimeStamp(6))

-----------------------------------------

                                52.260000

O comando INTERVAL

A função INTERVAL é usada para realizar a operação aritmética e de conversão nos valores DATA e HORA.

A Teradata fornece as funções INTERVAL abaixo.

Intervalos de 1 Rótulo

* ANO
* MÊS
* DIA
* HORA
* MINUTO
* SEGUNDO

Intervalos de 2 rótulos

* ANO A MÊS
* DIA A HORA
* DIA A MINUTO
* DIA A SEGUNDO
* HORA A MINUTO
* HORA PARA SEGUNDO
* MINUTO A SEGUNDO

Exemplo

O exemplo a seguir adiciona 2 anos à data atual.

SELECT CURRENT\_DATE, CURRENT\_DATE + INTERVAL '02' YEAR;

 Current Date  (Current Date+ 2)

------------  -----------------

  2016-09-21         2018-09-21

 O exemplo a seguir adiciona 3 anos e 3 meses à data atual.

SELECT CURRENT\_DATE, CURRENT\_DATE + INTERVAL '03-03' YEAR TO MONTH;

 Current Date  (Current Date+ 3-03)

------------  --------------------

  2016-09-21            2019-12-21

O exemplo a seguir adiciona 2 horas, 5 minutos e 10 segundos ao current\_timestamp.

SELECT CURRENT\_TIMESTAMP, CURRENT\_TIMESTAMP + INTERVAL '02:05:10' HOUR TO SECOND;

            Current TimeStamp(6)   (Current TimeStamp(6)+ 2:05:10)

--------------------------------  --------------------------------

2016-09-21 09:57:48.710000-04:00  2016-09-21 12:02:58.710000-04:00

O exemplo a seguir adiciona 2 dias e 10 minutos ao current\_timestamp.

SELECT CURRENT\_TIMESTAMP, CURRENT\_TIMESTAMP + INTERVAL '02 00:10' DAY TO MINUTE;

Current TimeStamp(6)   (Current TimeStamp(6)+ 2 00:10)

--------------------------------  --------------------------------

2016-09-21 10:02:05.220000-04:00  2016-09-23 10:12:05.220000-04:00

**Teradata DATE Format**

A função de formato Teradata DATE converte o formato de data padrão de acordo com os requisitos do usuário. Para o formato de data ANSI, por padrão, o Teradata segue o formato DATE como 'YYYY-MM-DD' e para o IntegerDate Teradata segue o formato de data como 'YY/MM/DD'.

A função de formato DATE pode substituir o formato de data padrão do sistema Teradata. Como alternativa, o administrador do sistema Teradata pode alterar o formato de data padrão quando o formulário de data é definido como tipos IntegerDate.

Alterar o formato DATE padrão envolve alterar o valor dos elementos de data em uma especificação personalizada para o arquivo de formatação DATE e usar o utilitário tdlocaledef para converter as informações em um formato interno utilizável pelo sistema Teradata.

Sintaxe do formato Teradata DATE

A sintaxe da função de formato DATE no Teradata é a seguinte.

SELECT date\_column FORMAT '<format\_type>'

FROM TABLE;

Aqui,

* FORMAT – A palavra-chave para formatar o valor da data que deve ser seguido pela coluna de dados.
* format\_type – O formato requerido que você deseja mostrar no conjunto de resultados.

Existem vários formatos DATE disponíveis no Teradata. Alguns deles são mencionados abaixo.

Formatos DATE do Teradata Exemplo

|  |  |
| --- | --- |
| FORMAT Phrase | Result |
| FORMAT ‘YY/MM/DD’ | 19/05/25 |
| FORMAT ‘DD-MM-YY’ | 28-01-2021 |
| FORMAT ‘YYYY/MM/DD’ | 28-01-2021 |
| FORMAT ‘YYYY-MM-DD’ | 28-01-2021 |
| FORMAT ‘YYYY.DDD’ | 1985.225 |
| FORMAT ‘YYBDDD’ | 85 225 |
| FORMAT ‘DDBMMMBYYYY’ | 12-Sep-85 |
| FORMAT ‘MMMBDD,BYYYY’ | Aug 12, 1999 |
| FORMAT ‘YYYYBMMMBDD’ | 1985 JUL 14 |
| FORMAT ‘MMM’ | Mar |
| FORMAT ‘EEE,BM4BDD,BYYYY’ | Thu, September 12, 1985 |
| FORMAT ‘E4,BMMMMBDD,BYYYY’ | Thursday, September 12, 1985 |
| FORMAT ‘E4BDDBM4BYYYY’ | Jeudi 12 Septembre 1985 |
| (Jeudi is French for Thursday and Septembre is French for September.) |
| FORMAT ‘999999’ | 850912 |

Formatos de TEMPO do Teradata Exemplo

|  |  |
| --- | --- |
| FORMAT Phrase | Result |
| FORMAT ‘HH:MIBT’ | 01:20 PM |
| FORMAT ‘HH:MI’ | 13:20 |
| FORMAT ‘HH.MI.SS’ | 13.20.53 |
| FORMAT ‘HH:MI:SSBT’ | 01:20:53 Nachm (Nachm is German for PM.) |
| FORMAT ‘HH:MI:SSDS(F)’ | 13:20:53.64 |
| FORMAT ‘HH:MI:SSDS(F)Z’ | 13:20:53.64+03:00 |
| FORMAT ‘HHhMImSSs’ | 13h20m53s |

Formatos do Teradata TIMESTAMP Exemplo

|  |  |
| --- | --- |
| FORMAT Phrase | Result |
| FORMAT ‘MM/DD/YYBHH:MIBT’ | 09/12/85 01:20 PM |
| FORMAT ‘MMMBDD,BYYBHH:MI:SS’ | Sep 12, 85 13:20:53 |
| FORMAT ‘E3,BM4BDD,BY4BHH:MI:SSDS(F)’ | Thu, September 12, 1985 13:20:53.64 |
| FORMAT ‘YYYY-MM-DDBHH:MI:SSDS(F)Z’ | 1985-09-12 13:20:53.64+03:00 |

**Teradata BETWEEN Date**

O operador Between no Teradata também pode ser usado para filtrar o conjunto de resultados para as colunas de tipo de data. A coluna de data pode ter datas em vários formatos.

Sintaxe Teradata Between DATE

A sintaxe geral para Teradata Between date é a seguinte.

select column1, column2,.., columnN

from DatabaseName.TableName

where date\_column between date1 and date2;

Teradata entre DATE exemplo

Considere a seguinte tabela Employees\_details.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| empno | ename | dob | job | mgr | hiredate | sal | deptno |
| 1 | SMITH | 1988-02-24 | CLERK | 7902 | 2016-06-13 | 800 | 20 |
| 2 | JONES | 1988-07-10 | SALESMAN | 7698 | 2018-05-18 | 1600 | 30 |
| 3 | BLAKE | 1990-10-25 | SALESMAN | 7698 | 2016-03-26 | 1250 | 30 |
| 4 | CLARK | 1991-09-23 | MANAGER | 7839 | 2015-10-31 | 2975 | 20 |
| 5 | SCOTT | 1992-01-12 | MANAGER | 7839 | 2016-06-11 | 2850 | 10 |
| 6 | KING | 1988-08-05 | ANALYST | 7839 | 2017-05-14 | 2450 | 10 |
| 7 | TURNER | 1988-09-17 | PRESIDENT | 7566 | 2017-05-05 | 3000 | 20 |

Agora vamos selecionar o nome e dob do funcionário cujo dob está entre 1-Jan-1988 a 31-Dez-1988 conforme abaixo.

select ename, dob

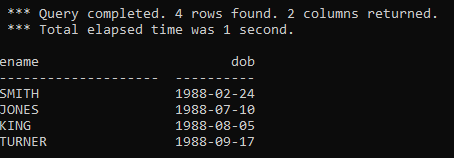
from teradatapoint.employee\_details

where dob between DATE '1988-01-01'

and DATE '1988-12-31'

order by dob;

**Output:**



No exemplo acima, o formato de data é AAAA-MM-DD.

Você pode alterar o formato de data acima para o formato inteiro.

select ename, dob (FORMAT '99-99-99')

from teradatapoint.employee\_details

where dob between 32143

and DATE '1988-12-31'

order by dob;

OU

select ename, dob (INTEGER)

from teradatapoint.employee\_details

where dob between 32143 and 32508

order by dob;

**Teradata Sample Function**

A função SAMPLE do Teradata retorna uma quantidade específica de dados de uma tabela ou exibição. Pode ser usado quando um dado menor ou mais gerenciável é desejável do que todo o conjunto de dados da tabela.

Sintaxe de função de exemplo do Teradata

SELECT \* FROM table sample n;

Aqui n pode ser um dos dois abaixo.

* Um número real de linhas.
* Uma porcentagem de tabelas. (n pode ser qualquer valor entre 1,00 e 0,00 )

Se o número n for maior que o número de linhas na tabela, a amostra n retornará todas as linhas da tabela.

Considere a seguinte tabela de empregados.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Empid** | **First\_Name** | **Last\_Name** | **Joining\_Date** | **salary** | **dept\_id** |
| 107 | Alexis | Bull | 1/12/2015 | 36000 | 18 |
| 109 | Adam | Vollman | 5/17/2013 | 19000 | 10 |
| 103 | Sarath | Sewall | 1/12/2013 | 23000 | 15 |
| 108 | Adam | Landry | 12/11/2013 | 42000 | 18 |
| 104 | Anthony | Gietz | 6/12/2014 | 13000 | 12 |
| 100 | Gerald | Cambault | 3/16/2001 | 25000 | 12 |
| 101 | Renske | Ladwig | 5/25/2003 | 10000 | 12 |
| 106 | Jennifer | Cabrio | 6/16/2012 | 29000 | 10 |
| 102 | Janette | King | 12/10/2010 | 26000 | 15 |
| 105 | Britney | Everett | 5/5/2013 | 45000 | 14 |

Exemplo

Selecione 5 registros de amostra da tabela Employee.

SELECT \* FROM TERADATAPOINT.EMPLOYEE Sample 5;

\*\*\* Query completed. 5 rows found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

Empid First\_Name Last\_Name Joining\_Date       salary      dept\_id

-----------  ----------  ----------  ------------  -----------  -----------

        107  Alexis      Bull          2015-01-12        36000           18

        109  Adam        Vollman       2013-05-17        19000           10

        103  Sarath      Sewall        2013-01-12        23000           15

        100  Gerald      Cambault      2001-03-16        25000           12

        105  Britney     Everett       2013-05-05        45000           14

 Selecione registros de amostra de 30% na tabela Funcionários.

**SELECT** \* **FROM** TERADATAPOINT.EMPLOYEE **Sample** .30**;**

\*\*\* Query completed. 3 rows found. 6 columns returned.

 \*\*\* Total elapsed time was 1 second.

Empid First\_Name Last\_Name Joining\_Date       salary      dept\_id

-----------  ----------  ----------  ------------  -----------  -----------

    107  Alexis      Bull          2015-01-12        36000           18

        103  Sarath      Sewall        2013-01-12        23000           15

        105  Britney     Everett       2013-05-05        45000           14

Temos um total de 10 linhas na tabela Employee. 30% da linha total é 3,33. Portanto, a saída é de 3 linhas.

O resultado fracionário inferior a 0,4999 não adicionará nenhuma linha.

SAMPLE usando SAMPLEID

SAMPLEID é uma extensão para SAMPLE para obter vários conjuntos de amostras em uma única consulta.

Exemplo

**SELECT** Empid, **SAMPLEID**

**FROM** TERADATAPOINT.EMPLOYEE

**SAMPLE** 4,4,4

**ORDER BY** 2,1**;**

\*\*\* Query completed. 10 rows found. 2 columns returned.

 \*\*\* Warning: 7473 Requested sample is larger than table rows. All rows returned

 \*\*\* Total elapsed time was 1 second.

Empid SampleId

-----------  -----------

        103            1

        105            1

        107            1

        109            1

        101            2

        102            2

        104            2

        106            2

        100            3

        108            3

Como você pode ver no exemplo acima, selecionamos 3 amostras. Na tabela Employee, temos 10 linhas, portanto, para a terceira amostra, não há linha suficiente para retornar. Portanto, recebemos a mensagem de aviso no ambiente BTEQ.

Exemplo :

**SELECT** Empid, **SAMPLEID**

**FROM** TERADATAPOINT.EMPLOYEE

**SAMPLE** .5,.2,.3

**ORDERBY** 2**;**

\*\*\* Query completed. 10 rows found. 2 columns returned.

 \*\*\* Total elapsed time was 1 second.

Empid SampleId

-----------  -----------

        107            1

        109            1

        100            1

        106            1

        102            1

        103            2

        105            2

        108            3

        104            3

        101            3

**Teradata TOP Function**

A função TOP é usada para produzir ‘N’ linhas da tabela com base em critérios específicos. Também produz

* A porcentagem TOP N da linha da tabela.
* Todos os valores idênticos se a opção WITH TIES for usada com valores de ordem de classificação.
* Linhas de maneira ordenada ou não ordenada.

Sintaxe da Função Teradata TOP

TOP {[ INTEGER | DECIMAL ]} [PERCENTAGE] [WITH TIES]

Nota – TOP N onde N é um número inteiro de até 18 dígitos de comprimento.

Considere a seguinte tabela de clientes.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| cust\_id | income | age | years\_with\_bank | nbr\_children | gender | marital\_status |
| 1363355 | 13087 | 22 | 1 | 0 | M | 2 |
| 1363329 | 0 | 6 | 0 | 0 | F | 1 |
| 1362672 | 16319 | 50 | 0 | 1 | F | 2 |
| 1362629 | 0 | 6 | 0 | 0 | M | 1 |
| 1362611 | 24115 | 48 | 8 | 1 | F | 2 |

Exemplo 1

A consulta abaixo retorna apenas os dois registros da tabela de clientes:

**SELECT** **TOP** 2 \* **FROM** teradatapoint.cutomer;

\*\*\* Query completed. 2 rows found. 7 columns returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id      income    age years\_with\_bank nbr\_children gender marital\_status

----------- ----------- ------ --------------- ------------ ------ --------

    1363329           0      6               0            0 F      1

    1362629           0      6               0            0 M      1

Exemplo 2

A consulta abaixo retorna 2 linhas, mas classificadas por idade.

**SELECT** **TOP** 2 cust\_id,age**FROM** teradatapoint.cutomer\_savings**ORDER** **BY** 2 **DESC;**

\*\*\* Query completed. 2 rows found. 2 columns returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id     age

-----------  ------

    1362672      50

    1362611      48

**TOP N WITH TIES**

**SELECT** **TOP** 3 **WITH** **TIES**cust\_id,years\_with\_bank **FROM** teradatapoint.cutomer\_savings **ORDER** **BY** 2 **DESC;**

\*\*\* Query completed. 5 rows found. 2 columns returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id  years\_with\_bank

-----------  ---------------

    1362611                8

    1363355                1

    1362629                0

    1363329                0

    1362672                0

Podemos ver no exemplo acima que, embora tenhamos especificado apenas TOP 3, ele retornou 5 linhas. Por causa da opção WITH TIES, ele escolheu todos os registros duplicados para a terceira posição.

A opção PORCENTAGEM

A função TOP N também pode produzir uma porcentagem de linhas, além de números absolutos de linhas.

**SELECT** **TOP** 30 **PERCENT**cust\_id,income **FROM** teradatapoint.cutomer\_savings **ORDER** **BY** 2 **DESC;**

\*\*\* Query completed. 2 rows found. 2 columns returned.

 \*\*\* Total elapsed time was 1 second.

    cust\_id       income

-----------  -----------

    1362611        24115

    1362672        16319

**Concatenation Operator || in Teradata**

A operação de concatenação no Teradata une duas strings fornecidas entre aspas simples. O operador de concatenação funciona de forma semelhante à função CONCAT. Este operador é usado principalmente na instrução SELECT.

A concatenação é o processo de combinar duas ou mais colunas em uma única coluna. Esta função tem compatibilidade com versões anteriores, então use || operador sempre que precisar concatenar qualquer string.

Exemplo:

SELECT 'Teradata ' ||'Point'

**Output**

Teradata Point

**Teradata Subquery**

A subconsulta Teradata é basicamente uma consulta SELET dentro de uma consulta. A consulta SELECT chamada como consulta interna é executada primeiro e a consulta externa usa o resultado da subconsulta.

Abaixo estão alguns dos recursos da subconsulta.

* Uma consulta pode ter várias subconsultas ou uma subconsulta pode ter outra subconsulta.
* As subconsultas não retornam registros duplicados.
* A subconsulta pode retornar valores individuais ou lista de valores.
* As subconsultas devem ser colocadas entre parênteses.

Abaixo está a sintaxe das subconsultas.

SELECT col1, col2, col3,…

FROM

Outer Table

WHERE col1 OPERATOR ( Inner SELECT Query);

Exemplo de subconsulta

Considere a tabela Estudante e Departamento abaixo.

Table Aluno

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Roll\_No | First\_Name | Last\_Name | City | dept\_no |
| 101 | Akash | Sharma | Mumbai | 204 |
| 106 | Sucheta | Mandal | Delhi | 201 |
| 104 | Raju | Saha | Mumbai | 204 |
| 102 | Sagar | Sharma | Kolkata | 205 |
| 100 | Robert | PY | Mumbai | 205 |
| 107 | Sourav | Das | Kanpur | 204 |
| 105 | Avijit | Pal | Bangalore | 201 |
| 103 | Milon | Mukherjee | Kolkata | 205 |

Table Departamento

|  |  |
| --- | --- |
| dept\_no | dept\_name |
| 205 | EEE |
| 201 | CSE |
| 204 | ECE |

A consulta a seguir identifica que o aluno pertence ao departamento CSE. O SELECT interno retorna o dept\_no do departamento CSE e a consulta externa usa esse resultado para identificar os detalhes do aluno.

**SELECT** ROLL\_NO, FIRST\_NAME, LAST\_NAME

**FROM** TERADATAPOINT.STUDENT

**WHERE** DEPT\_NO = (**SELECT** DEPT\_NO **FROM** TERADATAPOINT.DEPARTMENT

**WHERE** DEPT\_NAME='CSE'

)**;**

Quando a consulta acima é executada, ela produz o resultado abaixo.

\*\*\* Query completed. 2 rows found. 3 columns returned.

 \*\*\* Total elapsed time was 1 second.

    Roll\_No First\_Name            Last\_Name

-----------   ----------------           ---------------

        106   Sucheta                    Mandal

        105   Avijit                     Pal

A consulta a seguir identifica que o aluno pertence aos departamentos ECE e EEE. O SELECT interno retorna o dept\_no dos departamentos ECE e EEE e a consulta externa usa esse resultado para identificar os detalhes do aluno.

SELECT ROLL\_NO, FIRST\_NAME, LAST\_NAME

FROM TERADATAPOINT.STUDENT

WHERE DEPT\_NO IN (SELECT DEPT\_NO FROM TERADATAPOINT.DEPARTMENT

WHERE DEPT\_NAME IN ('ECE','EEE')

);

Quando a consulta acima é executada, ela produz o resultado abaixo.

\*\*\* Query completed. 6 rows found. 3 columns returned.

 \*\*\* Total elapsed time was 1 second.

    Roll\_No First\_Name                                         Last\_Name

----------- -------------------------------------------------- ------------

        100 Robert                                             PY

        103 Milon                                              Mukherjee

        102 Sagar                                              Sharma

        107 Sourav                                             Das

        104 Raju                                               Saha

        101 Akash                                              Sharma

**SUBSTR in Teradata**

SUBSTR or SUBSTRING function works same in Teradata. Only the syntax is different.

Sintaxe:

SUBSTR ( <column-name>, <start-location> [ , <length> ] )

Exemplo:

SELECT

SUBSTR('TeradataPoint', 4 , 4),

SUBSTR('TeradataPoint', 2 , 6),

SUBSTR('TeradataPoint', 8 , 4),

SUBSTR('TeradataPoint', 4 , 8);

Output:

|  |  |  |  |
| --- | --- | --- | --- |
| **Substr(‘TeradataPoint’,4,4)** | **Substr(‘TeradataPoint’,2,6)** | **Substr(‘TeradataPoint’,8,4)** | **Substr(‘TeradataPoint’,4,8)** |
| adat | eradat | aPoi | adataPoi |

Comprimento é opcional aqui. Se não for especificada, esta função retornará todos os caracteres restantes do local inicial.

SELECT SUBSTR('TeradataPoint', 4 );

Output:

\*\*\* Query completed. One row found. One column returned.

\*\*\* Total elapsed time was 1 second.

Substr('TeradataPoint',4)

-------------------------

adataPoint

**Teradata Identity Column**

A coluna de identidade em uma tabela define uma coluna que gera o valor automaticamente quando você define essa coluna como Identidade. Você não precisa inserir nenhum valor explicitamente para a coluna Identidade. O próprio Teradata cuida do valor depois que você o define.

No Teradata, você não pode garantir que o número gerado pela coluna Identity esteja sempre em sequência. Isso ocorre porque os números são gerados em uma base AMP-Local devido a motivos de desempenho.

Sintaxe da coluna de identidade do Teradata

A sintaxe para definir a coluna Identity é a seguinte.

column\_name INTEGER GENERATED always as IDENTITY

(START WITH value1

INCREMENT BY value2

MINVALUE value3

MAXVALUE value4

NO CYCLE)

Exemplo de coluna de identidade do Teradata

Vamos primeiro criar uma tabela de alunos onde roll\_no será Identity.

CREATE TABLE Teradatapoint.students

(

roll\_no INT GENERATED ALWAYS AS IDENTITY

(START WITH 1

INCREMENT BY 1

MINVALUE 0

MAXVALUE 99999

NO CYCLE),

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

gender CHAR(1) NOT NULL

);

Agora, insira alguns registros na tabela conforme abaixo.

INSERT INTO Teradatapoint.students (first\_name, last\_name,gender) VALUES ('Deepan','Das','M');

INSERT INTO Teradatapoint.students (first\_name, last\_name,gender) VALUES ('Sagar','Sharma','M');

INSERT INTO Teradatapoint.students (first\_name, last\_name,gender) VALUES ('Rupa','Das','F');

Se você verificar os dados da tabela, verá que roll\_no é gerado automaticamente. Não inserimos nenhum valor para a coluna roll\_no.

SELECT \* FROM Teradatapoint.students ORDER BY roll\_no;

\*\*\* Query completed. 3 rows found. 4 columns returned.

\*\*\* Total elapsed time was 1 second.

roll\_no first\_name last\_name gender

----------- -------------------------------------------------- ------------

1 Deepan Das  M

2 Sagar Sharma  M

3 Rupa Das  Y

**ABS Function in Teradata**

A função ABS calcula o valor absoluto de um determinado valor.

Função ABS na sintaxe do Teradata

ABS(number\_value)

Função ABS no Exemplo de Teradata

SELECT ABS(10);

Output: 10

SELECT ABS(-10);

Output: 10

SELECT ABS(+10)

Output: 10

A função Teradata ABS converte o valor negativo em positivo e, quando um valor positivo é fornecido, retorna o mesmo valor sem modificação.

Se o valor de entrada for uma string, o valor será convertido em valor flutuante.

SELECT ABS('+76');

Output: 76.00

**Qualify Clause in Teradata**

A cláusula Qualify filtra os resultados de uma função analítica ordenada calculada anteriormente de acordo com as condições de pesquisa especificadas pelo usuário.

A cláusula Qualify funciona de forma semelhante à cláusula HAVING. Mas a principal diferença entre Qualify e HAVING é que, com Qualify, a filtragem é baseada no resultado da execução de várias funções analíticas de ordem nos dados.

Sintaxe de cláusula de qualificação Teradata

A sintaxe da Teradata Qaulify Clause é a seguinte.

Qualify search\_condition;

Exemplo de sintaxe de cláusula de qualificação Teradata

Considere a seguinte tabela de desempenho de Empregado.

|  |  |  |
| --- | --- | --- |
| empid | empname | performance\_marks |
| 1 | Austin | 89 |
| 2 | Robinson | 84 |
| 3 | Taylor | 82 |
| 4 | Mark | 79 |
| 5 | Peter | 75 |

Se você quiser selecionar os detalhes do funcionário que obtiveram as notas de desempenho mais altas, você pode usar a cláusula Qualify junto com a função RANK() conforme abaixo.

SELECT \* FROM Teradatapoint.Employee

QUALIFY RANK(performance\_marks)=1;

|  |  |  |
| --- | --- | --- |
| empid | Name | Performance Marks |
| 1 | Austin | 89 |

Se você quiser selecionar a terceira nota mais alta, poderá obter o resultado desejado conforme abaixo.

SELECT \* FROM Teradatapoint.Employee

QUALIFY RANK(performance\_marks)=3;

|  |  |  |
| --- | --- | --- |
| empid | Name | Performance Marks |
| 3 | Taylor | 82 |

**Teradata RANK Function**

A função Teradata RANK atribui a classificação(1…n) de linhas no conjunto de resultados com base no valor da lista sort\_expression. As linhas com o mesmo valor recebem a mesma classificação no Teradata.

Você também pode especificar a ordem de classificação na função RANK. A ordem decrescente é a ordem padrão e atribui a classificação um ao maior valor na lista classificada. A ordem crescente atribui a classificação um ao menor valor da lista.

Sintaxe da função de classificação do Teradata

A sintaxe da função de classificação é a seguinte.

RANK ( sort\_expression [ASC | DESC] );

Onde,

* sort\_expression – pode ser uma expressão literal ou de coluna ou uma lista separada por vírgulas de expressões literais ou de coluna.
* ASC – ordem de classificação crescente.
* DESC – Ordem de classificação decrescente. A ordem de classificação padrão é DESC.

Exemplo de Função Teradata RANK

Considere a seguinte tabela de alunos com as notas obtidas por cada aluno.

|  |  |  |
| --- | --- | --- |
| rollno | student\_name | total\_marks |
| 1 | Rajib | 598 |
| 2 | Kanak | 685 |
| 3 | Rahul | 499 |
| 4 | Priyankit | 857 |
| 5 | Priti | 753 |
| 6 | Priya | 685 |

Agora podemos fornecer classificação para cada aluno usando a função RANK conforme abaixo.

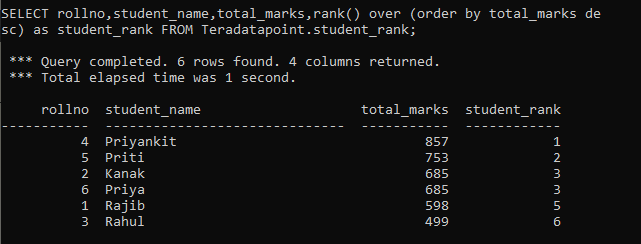
SELECT rollno,student\_name,total\_marks,rank() over (order by total\_marks de

sc) as student\_rank FROM Teradatapoint.student\_rank;

\*\*\* Query completed. 6 rows found. 4 columns returned.

\*\*\* Total elapsed time was 1 second.

Output:



Você também pode atribuir a classificação um ao aluno que obteve as notas mais baixas conforme abaixo.

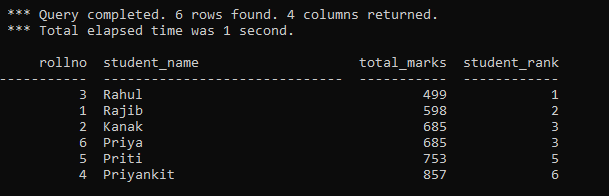
SELECT rollno,student\_name,total\_marks,rank(total\_marks asc) as student\_ran

k FROM Teradatapoint.student\_rank;

\*\*\* Query completed. 6 rows found. 4 columns returned.

\*\*\* Total elapsed time was 1 second.

Output:



**Teradata Joins**

Joins significa juntar mais de uma tabela com base nas colunas/valores comuns dessas tabelas. As junções são úteis para obter dados significativos para consultas de negócios.

O Teradata oferece suporte a diferentes tipos de junção, conforme abaixo.

* Inner Join.
* Left Outer Join.
* Right Outer Join.
* Full Outer Join.
* Self Join.
* Cross Join.
* Cartesian Production Join.

INNER JOIN

Inner Join combina registros de mais de uma tabela e retorna os valores que existem em ambas as tabelas.

Sintaxe

Select column1, column2, ...., columnN

from DatabaseName.TableName1

INNER JOIN

Databasename.TableName2

on column1=column2

where condition;

Exemplo

**Employee Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| empno | ename | dob | job | mgr | hiredate | sal | deptno |
| 1 | SMITH | 24-02-1988 | CLERK | 7902 | 13-06-2016 | 800 | 20 |
| 2 | JONES | 10-07-1988 | SALESMAN | 7698 | 18-05-2018 | 1600 | 30 |
| 3 | BLAKE | 25-10-1990 | SALESMAN | 7698 | 26-03-2016 | 1250 | 30 |
| 4 | CLARK | 23-09-1991 | MANAGER | 7839 | 31-10-2015 | 2975 | 20 |
| 5 | SCOTT | 12-01-1992 | MANAGER | 7839 | 11-06-2016 | 2850 | 10 |
| 6 | KING | 05-08-1988 | ANALYST | 7839 | 14-05-2017 | 2450 | 10 |
| 7 | TURNER | 17-09-1988 | PRESIDENT | 7566 | 05-05-2017 | 3000 | 20 |

**Department Table**

|  |  |  |
| --- | --- | --- |
| deptno | deptname | location |
| 10 | RESEARCH | DALLAS |
| 20 | SALES | CHICAGO |
| 40 | ACCOUNTING | NEW YORK |

A consulta a seguir junta a tabela de funcionários e a tabela de departamentos na coluna comum deptno.

select empno, ename, depname, job, hiredate, location

from teradatapoint.employee

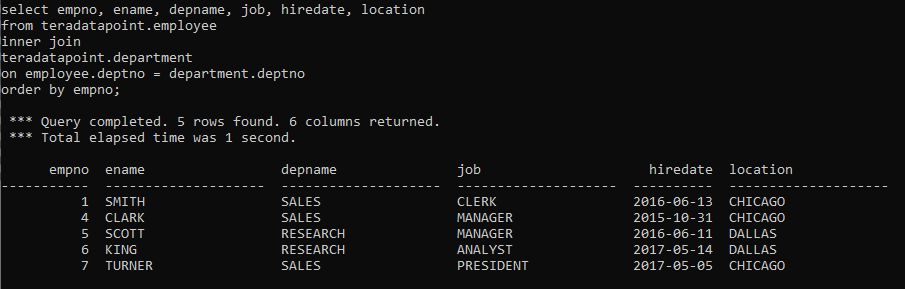
inner join

teradatapoint.department

on employee.deptno = department.deptno

order by empno;

Quando a consulta acima é executada, ela retorna os seguintes registros.



Os funcionários 2 e 3 não são incluídos no conjunto de resultados, pois não possuem os registros correspondentes na tabela de departamentos.

**OUTER JOIN**

A junção externa pode ser de três tipos.

* Left Outer Join – retorna todos os registros da tabela à esquerda e apenas os registros correspondentes da tabela à direita.
* Right Outer Join – retorna todos os registros da tabela da direita e apenas os registros correspondentes da tabela da esquerda.
* Full Outer Join – combina os resultados do lado esquerdo e do lado direito. Ele retorna linhas correspondentes e não correspondentes das tabelas de junção.

Outer Join Sintaxe:

Select column1, column2, ..., columnN

from DatabaseName.TableName1

LEFT OUTER JOIN/RIGHT OUTER JOIN/FULL OUTER JOIN

DatabaseName.Tablename2

on column1=column2

where condition;

Exemplo

A consulta a seguir retorna todos os registros da tabela da esquerda, ou seja, da tabela de funcionários, mesmo os registros correspondentes não estão presentes na tabela da direita, ou seja, da tabela de departamentos.

select empno, ename, depname, job, hiredate, location

from teradatapoint.employee

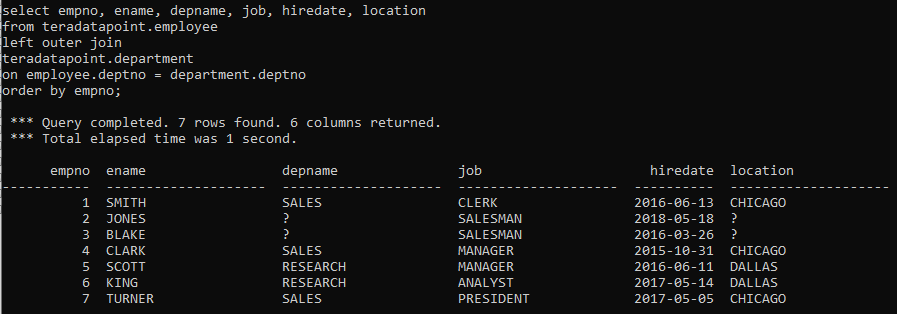
left outer join

teradatapoint.department

on employee.deptno = department.deptno

order by empno;

Depois de executar a consulta acima, ele retorna o resultado abaixo.



**RIGHT OUTER JOIN**

A consulta a seguir retorna todos os registros da tabela da direita, ou seja, da tabela de departamentos, mesmo que os registros correspondentes não estejam presentes na tabela da esquerda, ou seja, da tabela de funcionários.

select empno, ename, depname, job, hiredate, location

from teradatapoint.employee

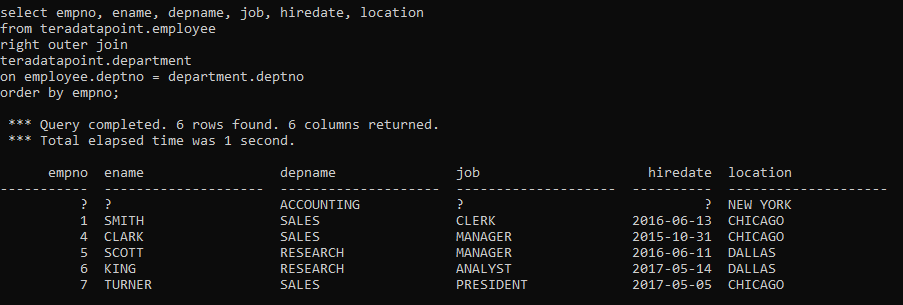
right outer join

teradatapoint.department

on employee.deptno = department.deptno

order by empno;

Depois de executar a consulta acima, ele retorna o resultado abaixo.



FULL OUTER JOIN

A consulta de junção a seguir retorna os registros correspondentes e não correspondentes da tabela de funcionários e departamentos.

select empno, ename, depname, job, hiredate, location

from teradatapoint.employee

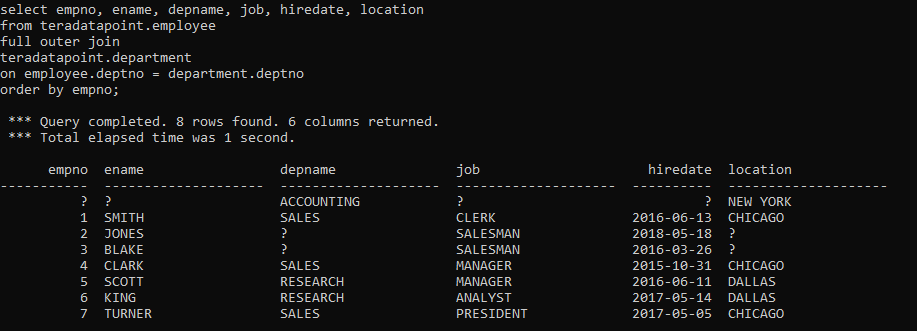
full outer join

teradatapoint.department

on employee.deptno = department.deptno

order by empno;

Quando a consulta acima é executada, ela retorna o seguinte resultado.



**CROSS JOIN**

A junção cruzada une cada linha da tabela à esquerda a cada linha da tabela à direita.

Sintaxe:

Select column1, column2, ..., columnN

from DatabaseName.TableName1

CROSS JOIN

DatabaseName.Tablename2

where condition;

Exemplo:

select empno, ename, depname, job, hiredate, location

from teradatapoint.employee

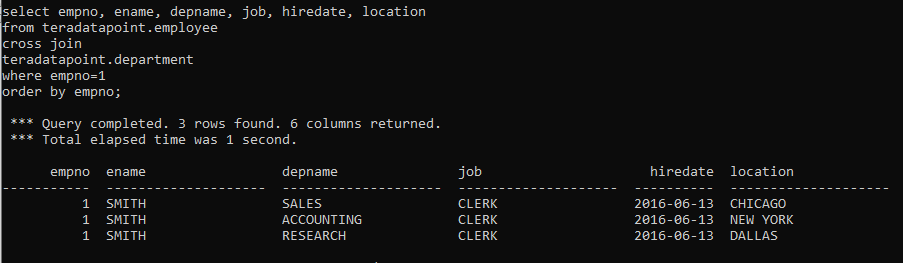
cross join

teradatapoint.department

where empno=1

order by empno;

Quando a consulta acima é executada, ela produz a seguinte saída. O funcionário 1 da tabela de funcionários é associado a todos os registros da tabela de departamentos.



**Teradata database space query**

Como administrador ou desenvolvedor do Teradata, você pode querer saber a utilização do espaço do banco de dados no sistema Teradata. Isso fornecerá uma imagem clara sobre como os espaços são alocados nos diferentes bancos de dados.

Seguir a consulta de espaço do banco de dados Teradata ajudará você a ter uma ideia sobre a distribuição do espaço.

Consulta de espaço de banco de dados Teradata:

SELECT

DatabaseName

,SUM(CurrentPerm)/1024/1024/1024 AS USEDSPACE\_IN\_GB

,SUM(MaxPerm)/1024/1024/1024 AS MAXSPACE\_IN\_GB

,SUM(CurrentPerm)/ NULLIFZERO (SUM(MaxPerm)) \*100 (FORMAT 'zz9.99%') AS Percentage\_Used

,MAXSPACE\_IN\_GB- USEDSPACE\_IN\_GB AS REMAININGSPACE\_IN\_GB

FROM DBC.DiskSpace

--WHERE DatabaseName = 'dbc'

order by 3 desc

GROUP BY DatabaseName;

Explicação:

Aqui,

* USEDSPACE\_IN\_GB indica o espaço utilizado atual para cada banco de dados.
* MAXSPACE\_IN\_GB indica espaço permanente alocado para cada banco de dados.
* Percentage\_Used indica o espaço utilizado atual em porcentagem.
* REMAININGSPACE\_IN\_GB indica o espaço restante para cada banco de dados.

Se você quiser ver apenas a utilização de um espaço de banco de dados específico, poderá fazê-lo fornecendo o nome do banco de dados na cláusula WHERE.

Se a consulta acima for executada, ela produzirá o resultado abaixo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DatabaseName | USEDSPACE \_IN\_GB | MAXSPACE \_IN\_GB | Percentage\_Used | REMAININGSPACE \_IN\_GB |
| DBC | 0.15 | 28.47 | 0.52 | 28.32 |
| TERADATAPOINT | 0 | 5 | 0 | 5 |
| financial | 0.02 | 2.11 | 0.82 | 2.09 |
| SystemFe | 0 | 0.75 | 0.02 | 0.74 |
| tdwm | 0 | 0.09 | 0.7 | 0.09 |
| dbcmngr | 0 | 0.09 | 0 | 0.09 |
| twm\_md | 0.07 | 0.07 | 92.49 | 0.01 |
| Crashdumps | 0 | 0.07 | 0 | 0.07 |
| tpch | 0.05 | 0.05 | 89.18 | 0.01 |
| SYSLIB | 0 | 0.04 | 11.14 | 0.03 |
| SYSBAR | 0 | 0.04 | 0.01 | 0.04 |
| SYSUDTLIB | 0 | 0.03 | 1.1 | 0.03 |
| retail | 0.02 | 0.02 | 95.19 | 0 |
| SysAdmin | 0 | 0.02 | 12.01 | 0.02 |
| twm\_source | 0.01 | 0.01 | 79.05 | 0 |
| Samples | 0 | 0.01 | 0.02 | 0.01 |
| SYSSPATIAL | 0 | 0.01 | 61.37 | 0 |

Espaço total no sistema Teradata

Você pode descobrir o espaço total do sistema Teradata usando a seguinte consulta.

SELECT

--DatabaseName

SUM(CurrentPerm)/1024/1024/1024 AS USEDSPACE\_IN\_GB

,SUM(MaxPerm)/1024/1024/1024 AS MAXSPACE\_IN\_GB

,SUM(CurrentPerm)/ NULLIFZERO (SUM(MaxPerm)) \*100 (FORMAT 'zz9.99%') AS Percentage\_Used

,MAXSPACE\_IN\_GB- USEDSPACE\_IN\_GB AS REMAININGSPACE\_IN\_GB

FROM DBC.DiskSpace;

Se a consulta acima for executada, ela produzirá o resultado abaixo.

|  |  |  |  |
| --- | --- | --- | --- |
| USEDSPACE\_IN\_GB | MAXSPACE\_IN\_GB | Percentage\_Used | REMAININGSPACE\_IN\_GB |
| 0.33 | 36.89 | 0.88 | 36.57 |

**Oreplace in Teradata**

Oreplace no Teradata é usado para substituir cada ocorrência de string correspondente pela nova string no Teradata. Oreplace é usado principalmente para substituir ou remover partes de uma string.

Sintaxe:

SELECT oreplace('source\_string','string\_to\_replace','new\_string')

Exemplo:

SELECT oreplace('TeradataPoint','Point','Tutorial');

Output:

TeradataTutorial

Substituindo uma string em uma coluna usando oreplace no Teradata

Em uma coluna específica, você pode substituir uma determinada string ou caracteres usando a função oreplace.

Sintaxe:

SELECT oreplace(column\_name, 'string\_to\_replace','new\_string');

Exemplo:

Considere uma coluna de uma tabela como essa:

|  |
| --- |
| **Pincode** |
| Pincode- 700001 |
| Pincode- 400614 |
| Pincode- 110001 |
| Pincode- 440012 |

Você pode remover a string “Pincode-” usando o sql abaixo.

SELECT oreplace(Pincode,'Pincode-','');

Output: (Substituindo “Pincode-” usando string vazia)

|  |
| --- |
| **Pincode** |
| 700001 |
| 400614 |
| 110001 |
| 440012 |

**OTRANSLATE in Teradata**

OTRANSLATE em Teradata:

* Retorna source\_string com cada ocorrência de cada caractere em from\_string substituído pelo caractere correspondente em to\_string.
* OTRANSLATE é uma função do sistema de serviços integrados. Pode ser ativado invocando funções de serviços incorporados
* As expressões passadas para esta função devem ter um dos seguintes tipos de dados: CHAR ou VARCHAR

Sintaxe:

OTRANSLATE(source\_string,from\_string,to\_string)

Exemplo:

SELECT OTRANSLATE('teradata point','dpo','tti');

No exemplo acima, o caractere 'd' será substituído pelo caractere 't' e 'p', 'o' será substituído pelo caractere 't' e 'i', respectivamente.

**Output**

teratata piint

Exemplo 2:

**SELECT** OTRANSLATE('teradata point','dpo','tt')**;**

No exemplo acima, os caracteres ‘d’ e ‘p’ serão substituídos pelos caracteres ‘t’ e ‘t’, respectivamente. O caractere 'o' também será removido da string de origem.

**Output**

teratata tint

**INSTR in Teradata**

INSTR em Teradata é usado para obter a posição de uma string de pesquisa na string de origem.

Sintaxe:

select instr( source\_string, search\_string [,position[,occurrence]])

Exemplo:

SELECT INSTR('choose a chocolate chip cookie','ch',2,2);

A consulta acima retornará 20, indicando a posição da string ‘ch’ em ‘chip’. Esta é a segunda ocorrência de 'ch' com a pesquisa começando na segunda posição da string de origem.

**Output**

20

Recursos do INSTR em Teradata:

* Se a posição for especificada, a pesquisa começará nesta posição no source\_string.
* A pesquisa começará no início da string de origem se a posição não for especificada.
* No caso de valor negativo especificado em position, a pesquisa contará e começará de trás para frente a partir do final da source\_string.
* Se a ocorrência for especificada, a pesquisa começará nesta posição na source\_string.
* Caso a ocorrência não seja especificada, a pesquisa começa no início de source\_string.
* A ocorrência não pode ser um valor negativo ou zero.

Perguntas de entrevista do Teradata:

1. O que é Master Index and Cylinder Index?

Teradata usa sistema de endereço lógico, ao contrário de outros DBMS que são sistema de endereço físico. Isso permite que o Teradata localize os dados facilmente se eles forem realocados ou desfragmentados. Master Index e Cylinder Index são duas estruturas deste sistema lógico que ajudam o Teradata a localizar os dados no bloco de dados.

Ao ler a linha de dados, o Master Index é usado para identificar o Cylinder Index no qual os dados são armazenados. O índice de cilindro é usado para fornecer o bloco de dados correto para essa linha.

2. O que é Clique no Teradata?

A combinação de nós Teradata que compartilham conjuntos comuns de matrizes de disco é chamada de Clique. O clique fornece a proteção contra a falha do nó, ou seja, se algum nó cair de um clique, o AMP vprocs será migrado para os outros nós dentro do clique.

3. O que é Cluster no Teradata?

Um conjunto de AMP chamado Cluster em Teradata.

Este é um grupo lógico que fornece capacidade de fallback. Se um AMP ficar inativo, os dados poderão ser acessados ​​por meio do amplificador de fallback no mesmo cluster. O tamanho padrão do cluster é 2.