T-SQL

Transact SQL

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

T-SQL (Transact-SQL) is an extension of SQL language. This tutorial covers the fundamental concepts of T-SQL such as its various functions, procedures, indexes, and transactions related to the topic. Each topic is explained using examples for easy understanding.

Audience

This tutorial is designed for those who want to learn the basics of T-SQL.

Prerequisites

To go ahead with this tutorial, familiarity with database concepts is preferred. It is good to have SQL Server installed on your computer, as it might assist you in executing the examples yourself and get to know how it works.

Disclaimer & Copyright

Table of Contents

1. T-SQL – Overview

In 1970's the product called 'SEQUEL', structured English query language, developed by IBM and later SEQUEL was renamed to 'SQL' which stands for Structured Query Language.

In 1986, SQL was approved by ANSI (American national Standards Institute) and in 1987, it was approved by ISO (International Standards Organization).

SQL is a structure query language which is a common database language for all RDBMS products. Different RDBMS product vendors have developed their own database language by extending SQL for their own RDBMS products.

T-SQL stands for Transact Structure Query Language which is a Microsoft product and is an extension of SQL Language.

Example

MS SQL Server - SQL\T-SQL

ORACLE - SQL\PL-SQL

2. T-SQL Server – Data Types

SQL Server data type is an attribute that specifies types of data of any object. Each column, variable and expression has related data type in SQL Server. These data types can be used while creating tables. You can choose a particular data type for a table column based on your requirement.

SQL Server offers seven categories including other category of data types for use.

Exact Numeric Types

Type	From	To
bigint	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
int	-2,147,483,648	2,147,483,647
smallint	-32,768	32,767
tinyint	0	255
bit	0	1
decimal	-10^38 +1	10^38 –1
numeric	-10^38 +1	10^38 –1
money	-922,337,203,685,477.5808	+922,337,203,685,477.5807
smallmoney	-214,748.3648	+214,748.3647

Numeric and decimal are Fixed precision and scale data types and are functionally equivalent.

Approximate Numeric Types

Type	From	To
Float	-1.79E + 308	1.79E + 308
Real	-3.40E + 38	3.40E + 38

Date and Time Types

Type	From	To
datetime (3.33 milliseconds accuracy)	Jan 1, 1753	Dec 31, 9999

smalldatetime (1 minute accuracy)	Jan 1, 1900	Jun 6, 2079
date (1 day accuracy. Introduced in SQL Server 2008)	Jan 1, 0001	Dec 31, 9999
datetimeoffset (100 nanoseconds accuracy. Introduced in SQL Server 2008)	Jan 1, 0001	Dec 31, 9999
datetime2 (100 nanoseconds accuracy. Introduced in **SQL Server 2008**)	Jan 1, 0001	Dec 31, 9999
time (100 nanoseconds accuracy. Introduced in **SQL Server 2008**)	00:00:00.0000000	23:59:59.9999999

Character Strings

Type	Description
char	Fixed-length non-Unicode character data with a maximum length of 8,000 characters.
varchar	Variable-length non-Unicode data with a maximum of 8,000 characters.
Varchar (max)	Variable-length non-Unicode data with a maximum length of 2^{31} characters (Introduced in SQL Server 2005).
text	Variable-length non-Unicode data with a maximum length of 2,147,483,647 characters.

Unicode Character Strings

Type	Description
nchar	Fixed-length Unicode data with a maximum length of 4,000 characters.
nvarchar	Variable-length Unicode data with a maximum length of 4,000 characters.
Nvarchar (max)	Variable-length Unicode data with a maximum length of 2^{30} characters (Introduced in SQL Server 2005).

ntext	Variable-length Unicode data with a maximum length of 1,073,741,823 characters.

Binary Strings

Type	Description
binary	Fixed-length binary data with a maximum length of 8,000 bytes.
varbinary	Variable-length binary data with a maximum length of 8,000 bytes.
varbinary(max)	Variable-length binary data with a maximum length of 2^{31} bytes (Introduced in SQL Server 2005).
image	Variable-length binary data with a maximum length of 2,147,483,647 bytes.

Other Data Types

- **sql_variant**: Stores values of various SQL Server-supported data types, except text, ntext, and timestamp.
- **timestamp**: Stores a database-wide unique number that gets updated every time a row gets updated.
- **uniqueidentifier**: Stores a globally unique identifier (GUID).
- **xml**: Stores XML data. You can store XML instances in a column or a variable (Introduced in SQL Server 2005).
- **cursor**: A reference to a cursor.
- **table**: Stores a result set for later processing.
- **hierarchyid**: A variable length, system data type used to represent position in a hierarchy (Introduced in SQL Server 2008).

3. T-SQL Server – Create Tables

Creating a basic table involves naming the table and defining its columns and each column's data type.

The SQL Server **CREATE TABLE** statement is used to create a new table.

Syntax

Following is the basic syntax of CREATE TABLE statement:

```
CREATE TABLE table_name(
   column1 datatype,
   column2 datatype,
   column3 datatype,
   .....
   columnN datatype,
   PRIMARY KEY( one or more columns ));
```

CREATE TABLE is the keyword telling the database system what you want to do. In this case, you want to create a new table. The unique name or identifier for the table follows the CREATE TABLE statement. Then in brackets comes the list defining each column in the table and what sort of data type it is. The syntax becomes clearer to understand with the following example.

A copy of an existing table can be created using a combination of the CREATE TABLE statement and the SELECT statement. You can check complete details at Create Table Using another Table.

Example

In this example, let's create a CUSTOMERS table with ID as primary key and NOT NULL are the constraints showing that these fields cannot be NULL while creating records in this table:

```
CREATE TABLE CUSTOMERS(
   ID   INT              NOT NULL,
   NAME VARCHAR (20)     NOT NULL,
   AGE  INT              NOT NULL,
   ADDRESS  CHAR (25) ,
   SALARY   DECIMAL (18, 2),
   PRIMARY KEY (ID));
```

You can verify if your table has been created successfully by looking at the message displayed by the SQL server, otherwise you can use the following command:

```
exec sp_columns CUSTOMERS
```

The above command produces the following output.

```
TABLE_QUALIFIER    TABLE_OWNER  TABLE_NAME   COLUMN_NAME  DATA_TYPE
     TYPE_NAME     PRECISION    LENGTH SCALE  RADIX  NULLABLE     REMARKS
     COLUMN_DEF    SQL_DATA_TYPESQL_DATETIME_SUB    CHAR_OCTET_LENGTH
     ORDINAL_POSITION    IS_NULLABLE  SS_DATA_TYPE


TestDB      dbo    CUSTOMERS    ID     4      int    10     4      0      10     0
     NULL   NULL   4      NULL   NULL   1      NO     56


TestDB      dbo    CUSTOMERS    NAME   12     varchar       20     20     NULL
     NULL   0      NULL   NULL   12     NULL   20     2      NO     39


TestDB      dbo    CUSTOMERS    AGE    4      int    10     4      0      10     0
     NULL   NULL   4      NULL   NULL   3      NO     56


TestDB      dbo    CUSTOMERS    ADDRESS       1      char   25     25     NULL
     NULL   1      NULL   NULL   1      NULL   25     4      YES    39


TestDB      dbo    CUSTOMERS    SALARY 3      decimal       18     20     2      10
     1      NULL   NULL   3      NULL   NULL   5      YES    106
```

You can now see that CUSTOMERS table is available in your database which you can use to store required information related to customers.

4. T-SQL Server – Drop Tables

The SQL Server **DROP TABLE** statement is used to remove a table definition and all data, indexes, triggers, constraints, and permission specifications for that table.

Note: You have to be careful while using this command because once a table is deleted then all the information available in the table would also be lost forever.

Syntax

Following is the basic syntax of DROP TABLE statement:

```
DROP TABLE table_name;
```

Example

Let us first verify CUSTOMERS table and then we will delete it from the database:

```
Exec sp_columns CUSTOMERS;
```

The above command shows the following table.

```
TABLE_QUALIFIER    TABLE_OWNER  TABLE_NAME   COLUMN_NAME  DATA_TYPE
     TYPE_NAME     PRECISION    LENGTH SCALE  RADIX  NULLABLE     REMARKS
     COLUMN_DEF    SQL_DATA_TYPESQL_DATETIME_SUB    CHAR_OCTET_LENGTH
     ORDINAL_POSITION    IS_NULLABLE  SS_DATA_TYPE


TestDB      dbo    CUSTOMERS    ID     4      int    10     4      0      10     0
     NULL   NULL   4      NULL   NULL   1      NO     56

TestDB      dbo    CUSTOMERS    NAME   12     varchar       20     20     NULL
     NULL   0      NULL   NULL   12     NULL   20     2      NO     39

TestDB      dbo    CUSTOMERS    AGE    4      int    10     4      0      10     0
     NULL   NULL   4      NULL   NULL   3      NO     56


TestDB      dbo    CUSTOMERS    ADDRESS       1      char   25     25     NULL
     NULL   1      NULL   NULL   1      NULL   25     4      YES    39


TestDB      dbo    CUSTOMERS    SALARY 3      decimal       18     20     2      10
     1      NULL   NULL   3      NULL   NULL   5      YES    106
```

CUSTOMERS table is available in the database, so let us drop it. Following is the command for the same.

```
 DROP TABLE CUSTOMERS;
Command(s) completed successfully.
```

With the above command, you will not get any rows.

```
Exec sp_columns CUSTOMERS;
No rows\data will be displayed
```

End of ebook preview

If you liked what you saw…

Buy it from our store @ https://store.tutorialspoint.com