**DATA DEFINITION LANGUAGE:**

Data Definition Language (DDL) is a standard for commands that define the different structures in a database. DDL statements create, modify, and remove database objects such as tables, indexes, and users. Common DDL statements are CREATE, ALTER, and DROP [1].

**Domain Type in SQL**

The SQL-92 standard supports a variety of built-in domain types:

* **char**(n) (or **character**(n)): fixed-length character string, with user-specified length.
* **varchar**(n) (or **character varying**): variable-length character string, with user-specified maximum length.
* **int** or **integer**: an integer (length is machine-dependent).
* **smallint**: a small integer (length is machine-dependent).
* **numeric** (*p, d*): a fixed-point number with user-specified precision, consists of *p* digits (plus a sign) and *d* of *p* digits are to the right of the decimal point. E.g., **numeric** (*3, 1*) allows 44.5 to be stored exactly but not 444.5.
* **real** or **double precision**: floating-point or double-precision floating-point numbers, with machine-dependent precision.
* **float**(n): floating-point, with user-specified precision of at least *n* digits.
* **date**: a calendar date, containing four digit year, month, and day of the month.
* **time**: the time of the day in hours, minutes, and seconds.

SQL-92 allows arithmetic and comparison operations on various numeric domains, including, interval and *cast* (*type coercion*) such as transforming between *smallint* and *int*. It considers strings with different length are compatible types as well [2].

SQL-92 allows **create domain** statement, e.g.,

**create domain** *person-name* **char**(20)

**Schema Definition in SQL**

**SQL Server** 2005 implemented the concept of a database object **schema**. A **schema** is a distinct namespace to facilitate the separation, management, and ownership of database objects. It removed the tight coupling of database objects and owners to improve the security administration of database objects.

**DATA MANIPULATION LANGUAGE:**

Data Manipulation Language (DML) is a vocabulary used to retrieve and work with data in SQL Server 2016. Use these statements to add, modify, query, or remove data from a SQL Server database [3].

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Clause** |  | **Can be used in these statements** | | [FROM (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms177634.aspx) |  | DELETE, SELECT, UPDATE | | [Hints (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms187713.aspx) |  | DELETE, INSERT, SELECT, UPDATE | | [OPTION Clause (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms190322.aspx) |  | DELETE, SELECT, UPDATE | | [OUTPUT Clause (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms177564.aspx) |  | DELETE, INSERT, MERGE, UPDATE | | [Search Condition (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms173545.aspx) |  | DELETE, MERGE, SELECT, UPDATE | | [Table Value Constructor (Transact-SQL)](https://msdn.microsoft.com/en-us/library/dd776382.aspx) |  | FROM, INSERT, MERGE | | [TOP (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms189463.aspx) |  | DELETE, INSERT, MERGE, SELECT, UPDATE | | [WHERE (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms188047.aspx) |  | DELETE, SELECT, UPDATE | | [WITH common\_table\_expression (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms175972.aspx) |  | DELETE, INSERT, MERGE, SELECT, UPDATE | |

**The Select Clause**

The SQL SELECT clause specifies the fields, constants, and expressions to display in the query results.

SYNTAX

SELECT [ALL | DISTINCT] [TOP nExpr [PERCENT]] Select\_List\_Item [AS Column\_Name] [, ...]

**The Where Clause**

The WHERE clause specifies join and filter conditions that determine the rows that the query returns. Join operations in the WHERE clause function the same as JOIN operations in the FROM clause.

SYNTAX

[WHERE JoinCondition | FilterCondition [AND | OR JoinCondition | FilterCondition] ...]

**The From Clause**

The FROM clause specifies one or more tables containing the data that the query retrieves from.

SYNTAX

FROM [FORCE] Table\_List\_Item [, ...]

[[JoinType] JOIN DatabaseName!]Table [[AS] Local\_Alias]

[ON JoinCondition [AND | OR [JoinCondition | FilterCondition] ...] [4].

**The Rename Operation**

With RENAME statement you can rename a table.

SYNTAX

RENAME TABLE {tbl\_name} TO {new\_tbl\_name};[5].

**Tuple Variable**

A tuple is an ordered multiset of attributes, which are ordered pairs of domain and value; or just a row. These variables can then be used throughout the expression. Think of it as being something like the rename operator. A relvar (relation variable) is a set of ordered pairs of domain and name, which serves as the header for a relation. A relation is a set of tuples.

Tuple variables can be used in SQL, and are defined in the **from** clause:

**select distinct** *pname, A.money#*

**from** *borrower* ***as*** *S, money* ***as*** *A*

**where** *S.money# = A.money#* [6].

**String Operations**

|  |  |
| --- | --- |
| [ASCII()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_ascii) | Returns numeric value of left-most character |
| [BIN()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_bin) | Returns a string representation of the argument |
| [BIT\_LENGTH()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_bit-length) | Returns length of argument in bits |
| [CHAR\_LENGTH()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_char-length) | Returns number of characters in argument |
| [CHAR()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_char) | Returns the character for each integer passed |
| [CHARACTER\_LENGTH()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_character-length) | A synonym for CHAR\_LENGTH() |
| [CONCAT\_WS()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_concat-ws) | Returns concatenate with separator |
| [CONCAT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_concat) | Returns concatenated string |
| [CONV()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_conv) | Converts numbers between different number bases |
| [ELT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_elt) | Returns string at index number |
| [EXPORT\_SET()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_export-set) | Returns a string such that for every bit set in the value bits, you get an on string and for every unset bit, you get an off string |
| [FIELD()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_field) | Returns the index (position) of the first argument in the subsequent arguments |
| [FIND\_IN\_SET()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_find-in-set) | Returns the index position of the first argument within the second argument |
| [FORMAT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_format) | Returns a number formatted to specified number of decimal places |
| [HEX()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_hex) | Returns a string representation of a hex value |
| [INSERT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_insert) | Inserts a substring at the specified position up to the specified number of characters |
| [INSTR()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_instr) | Returns the index of the first occurrence of substring |
| [LCASE()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_lcase) | Synonym for LOWER() |
| [LEFT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_left) | Returns the leftmost number of characters as specified |
| [LENGTH()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_length) | Returns the length of a string in bytes |
| [LOAD\_FILE()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_load-file) | Loads the named file |
| [LOCATE()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_locate) | Returns the position of the first occurrence of substring |
| [LOWER()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_lower) | Returns the argument in lowercase |
| [LPAD()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_lpad) | Returns the string argument, left-padded with the specified string |
| [LTRIM()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_ltrim) | Removes leading spaces |
| [MAKE\_SET()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_make-set) | Returns a set of comma-separated strings that have the corresponding bit in bits set |
| [MID()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_mid) | Returns a substring starting from the specified position |
| [OCT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_oct) | Returns a string representation of the octal argument |
| [OCTET\_LENGTH()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_octet-length) | A synonym for LENGTH() |
| [ORD()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_ord) | If the leftmost character of the argument is a multi-byte character, returns the code for that character |
| [POSITION()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_position) | A synonym for LOCATE() |
| [QUOTE()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_quote) | Escapes the argument for use in an SQL statement |
| [RIGHT()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_right) | Returns the specified rightmost number of characters |
| [RPAD()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_rpad) | Appends string the specified number of times |
| [SOUNDEX()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_soundex) | Returns a soundex string |
| [SOUNDS LIKE](http://www.tutorialspoint.com/sql/sql-string-functions.htm#operator_sounds-like) | Compares sounds |
| [SPACE()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_space) | Returns a string of the specified number of spaces |
| [STRCMP()](http://www.tutorialspoint.com/sql/sql-string-functions.htm#function_strcmp) | Compares two strings |

**Ordering the Display of Tuples**

SQL allows the user to control the order in which tuples are displayed.

* **order by** makes tuples appear in sorted order (ascending order by default).
* **desc** specifies descending order.
* **asc** specifies ascending order.

**Duplicate Tuples**

* Formal query languages are based on mathematical relations. Thus no duplicates appear in relations.
* As duplicate removal is expensive, SQL allows duplicates.
* To remove duplicates, we use the **distinct** keyword.
* To ensure that duplicates are not removed, we use the **all** keyword.

**REFERENCE:**

[1]<http://whatis.techtarget.com/definition/Data-Definition-Language-DDL>

[2] <http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter4/node31.html>

[3] <https://msdn.microsoft.com/en-us/library/ff848766.aspx>

[4] <https://www.google.com/search?q=rename+operation+in+sql&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:en-US:official&client=firefox-beta>

[5] <http://beginner-sql-tutorial.com/sql-rename.htm>

[6] <http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter4/node8.html>