**USER DEFINED DATA TYPES IN CQL**

Handling several fields of connected data in a table is made easier with the use of a user-defined type. By representing the linked fields of information using a user-defined type rather than storing them in separate tables, applications that previously required numerous tables can be made simpler and use fewer tables.

User-defined types (UDTs) are defined using CQL. Using the create\_type\_statement, alter\_type\_statement, and drop\_type\_statement functions listed below, such a type can be established, changed, and eliminated. However, after it is built, a UDT is only called by its name:

user\_defined\_type::= udt\_name

udt\_name::= [ keyspace\_name '.' ] identifier

Using a user-defined type is demonstrated in the address type example.

These commands let you to create, modify, and remove a user-defined type:

* CREATE TYPE
* ALTER TYPE
* DROP TYPE

**CREATE TYPE :**

Cassandra 2.1 and up need the creation of a user-defined type. One or more typed fields with associated data, like the address information (street, city, and postal code), are contained in a user-defined type.

Creating a new user-defined type is done using a CREATE TYPE statement defined by:

CREATE TYPE [IF NOT EXISTS]

keyspace\_name.type\_name(

field\_name cql\_datatype[,]

[field\_name cql\_datatype] [,...]

)

IF NOT EXISTS

Suppresses the error if the type already exists in the keyspace. UDT scope is keyspace-wide.

type\_name

Unique name for the type, CQL types are reserved for a list see type names.

Points to consider with UDTs:

* Unless the IF NOT EXISTS option is used, attempting to create an already existing type will result in an error. If the type already exists, using it will result in a no-op statement.
* A type can only be utilized in the keyspace in which it was generated since it is inherently connected to that keyspace. If a keyspace name comes before the type name during creation, that keyspace is used for creation. If not, it is made in the keyspace that is currently in use.
* Since most UDT scenarios require freezing as of Cassandra, the table definition above has frozen<address>. For further information, please refer to the frozen section.

After defining the UDT, you can create a table that has columns with the UDT. CQL collection columns and other columns support the use of user-defined types.

**ALTER TYPE :**

An existing user-defined type can be modified using an ALTER TYPE statement:

 ALTER TYPE field\_name

[ALTER field\_name TYPE new\_cql\_datatype

| ADD (field\_name cql\_datatype[,...])

| RENAME field\_name TO new\_field\_name[AND ...]]

Use AND in between the clauses to modify the UDT more than once.

ALTER field\_name TYPE new\_cql\_datatype

Change the data type of a field. Specify the field name and the new cql datatype.

ADD (field\_name cql\_datatype[,...])

Add fields by entering a field name followed by the data type in a comma separated list; the values for existing rows is set to null.

RENAME field\_name TO new\_field\_name

Enter the old name and new name of the field.

The statement will return an error if the type is not found, unless IF EXISTS is used, in which case the operation is not necessary.

To the type, add a new field (ALTER TYPE address ADD country text). For all values of the type created before to the addition, that new field will be NULL. The statement will produce an error if the new field already exists, unless IF NOT EXISTS is used, in which case there is no need to perform the procedure.

Rename the type's fields. Unless IF EXISTS is used, in which case the operation is a no-op, the statement will return an error if the field or fields do not exist.

**DROP TYPE :**

You can drop an existing user-defined type using a DROP TYPE statement:

DROP TYPE [IF EXISTS] keyspace\_name.type\_name

A type is instantly and irreversibly removed when it is dropped. An error will occur if you try to drop a type while it is still being used by another type, table, or function.

The operation is a no-op if the type dropped does not exist; otherwise, an error will be returned unless IF EXISTS is applied.