**CASSANDRA QUERY LANGUAGE**

WHAT IS CASSANDRA QUERY LANGUAGE ?

The computer languages used to create queries in databases and information systems are known as query languages. The main query language used to interact with the Apache Cassandra database is called Cassandra Query Language.

A model resembling SQL is provided by CQL. Tables with rows and columns storing the data are used for storage. Because of this, the terms "tables," "rows," and "columns" have the same meanings in this article as they do in SQL.

The common query language for working with relational databases is SQL (Structured Query Language). Relational query languages, like SQL, are table-based, fixed-schema databases that support moderate data quantities and are vertically scalable. NoSQL query languages, like CQL, are highly scalable, manage unstructured data, do not require a set schema, do not require joins, and are simple to scale. They can also query over horizontally distributed server clusters.



Basic Cassandra (CQL) includes:

**Keyspace** — Similar to an RDBMS database, a keyspace is a container for application data that must have a name and a set of associated attributes. Cassandra keyspace is a SQL database.

**Column Families/Tables** — A keyspace consists of a number of Column Families/Tables. A Cassandra column family is a SQL table.

**Primary Key / Tables** — A Primary Key consists of a Row/Partition Key and a Cluster Key, and functions to enable users to uniquely identify internal rows of data. A Row/Partition Key determines the node on which data is stored. A Cluster Key determines the sort order of data within a particular row.

In this document, we will employ the following conventions to help specify the CQL syntax:

* The language rules will be provided in a BNF variant notation that is informal. Specifically, for optional things, we'll use square brackets ([ item ]), and for repeated items, we'll use \* and + (where + implies at least one).
* For simplicity, the grammar will also follow the following convention: terminal keywords will be presented in "all caps," while non-terminal terms will be lowercase and link to their meaning. But keep in mind that, as they are identifiers, keywords are actually case-insensitive. Additionally, we will use regexp to define some early construction, which we will denote with re(<some regular expression>).
* The grammar is given only for documentational purposes and omits a few little elements. For example, even if the syntax in this manual says otherwise, the comma on the last column definition in a CREATE TABLE statement is optional but supported if present. Furthermore, not all grammar-accepted sentences are necessarily proper CQL.
* Running text references to keywords or CQL code segments will appear in a fixed-width font.

**CQL DOCUMENTATION :**

**CQL language documentation includes:**

* Definitions for Conventions, Identifiers and Keywords, Constants, Terms, Comments,
* Statements, and Prepared Statements
* CQL Data Types
* Definitions of statements used to create, modify, and remove keyspaces and tables, such as Create Keyspace, Alter Keyspace, and Drop Keyspace; Create Table, Alter Table, and Drop Table; Truncate; and Describe.
* Data Manipulation statements, such as CQL Select, Insert, Update, Delete, and Batch
* Secondary Index statements such as Create Index and Drop Index
* Materialized Views statements such as Create

Materialized View, Alter Materialized View, and Drop Materialized View

* Security definitions for Database Roles, Users, and Data Control
* Definitions for Scalar Functions and Aggregate Functions
* Definitions for the arithmetic operators that CQL supports — Number Arithmetic and Datetime Arithmetic
* JSON Support documentation
* Trigger definitions for Create Trigger and Drop Trigger
* Appendices for CQL Keywords, CQL Reserved Types, and Dropping Compact Storage
* Cassandra CQL updates from 3.0.1 to 3.4.5