DataStax DevCenter

Documentation

${ds.localized.time}

Contents

Features

DataStax DevCenter is a free visual query IDE for developers, administrators, and others who want to create and run Cassandra Query Language (CQL) statements against Apache Cassandra and DataStax Enterprise.

New features in DevCenter 1.3.1

- Support for Cassandra 2.1.3 CQL features:
  - Frozen collections including nested collections and `FULL` collection indexes.
  - Inclusion of `IF EXISTS` for `UPDATE` statements.
- Editor and wizard support for the `DateTieredCompactionStrategy`.
- Additional table options available in the **Create** and **Edit Table** wizard:
  - `default_time_to_live`
  - `gc_grace_seconds`
  - `index_interval`
  - `min_index_interval`
  - `max_index_interval`
  - `memtable_flush_period_in_ms`
  - `populate_io_cache_on_flush`
  - `speculative_retry`

Features in DevCenter 1.3.0

- Execute (single or multiple) selected statement(s)
- Wizards
  - Keyspace wizard
  - Table wizard
- A feedback form has been added

Features in DevCenter 1.2.1

- **New features**
  - Prevent execution of CQL scripts containing syntax errors
  - *Copy as INSERT* now supports collections of UDTs or Tuples
- **Bug fixes**
  - Fix issue on Mac OS/X Yosemite which caused first row of results and query trace table to be hidden
  - Fix NPE appearing in log file on some content assist actions

Features in DevCenter 1.2

- Full support for Cassandra 2.1, including:
  - user-defined types (UDTs)
  - tuples
  - `IF EXISTS`
- Syntax highlighting, representation in **Results** tab and **Schema navigator**, validation, content assist, and code snippets for UDTs and tuples
- A new **Query Trace** tab, located next to the **Results** tab, which displays detailed trace event data for the last executed query to aid in understanding query execution and performance

- An improved **New Connection** wizard for creating and managing Cassandra connections

Features in DevCenter 1.1

- Support for Apache Cassandra 2.0.x and DataStax Enterprise 4.0.x (lightweight transactions syntax, static columns, `uuid()`, `now`, etc.)
- New and improved validation and code-assist rules
- Option to use a default keyspace for running a script
- Option to set the maximum number of rows to be returned by a statement
- Copy selected or all results as CSV or CQL inserts
- Option to enabled SSL connection
- See the posting on the DataStax Developer Blog about the DevCenter 1.1 feature set

Features in DevCenter 1.0

- A smart CQL editor that provides
  - syntax highlighting
  - code auto-completion (both keyword and snippet)
  - real-time script validation against the current connection
- Schema explorer view for browsing
  - keyspaces
  - tables
  - other database objects
- Outline view for allowing quick navigating long CQL scripts
- Configuring connections to Cassandra or DataStax Enterprise clusters requiring authorization

Installing DevCenter

Installing DevCenter is easy.

Before you begin

DevCenter is a Java application and at a minimum a Java runtime environment (JRE versions 6, 7, or 8 are supported) is required to run it.

Procedure

1. Download the version appropriate for your operating system.
2. Unzip the downloaded file to a location on your hard drive.
3. Run the executable file, `DevCenter`.

What to do next

Take the DevCenter tutorial.

Using SSL connections

Connecting DevCenter to an SSL-enabled Cassandra or DataStax Enterprise cluster.

Before you begin

- SSL must be configured and working on your cluster.
  - Client-to-node encryption.
  - Node-to-node encryption.
  - Preparing server certificates.
- Install the Java Cryptography Extension (JSE) on your client system.

  Download:

  - Java 8
  - Java 7
  - Java 6

  Installation directory (`jre lib/security`):

  - Linux: `/usr/lib/jvm/jdk1.`*`major`*`.`*`minor_update`*`/jre/lib/security`
  - Mac OS X: `/Library/Java/JavaVirtualMachines/jdk1.`*`major`*`.`*`minor_update`*`/Contents/Home/jre/lib/security`
  - Windows: `C:\Program Files\Java\jre7\lib\security`

  Extract the downloaded file and copy the content of `UnlimitedJCEPolicy` directory to the `jre/lib/security` directory.
- The `keytool` command to manage encryption keys.

  Note: If you cannot find the `keytool` command on a Windows system, read these instructions.

About this task

Procedure

Server verification

1. To perform server verification, the client needs to have the public key certificate of each node in the cluster stored in a local truststore file. This file is password protected (keytool will prompt to create a password). The truststore file and password will be entered into the DevCenter connection manager dialog box (see below).
   a) Using `keytool`, create a truststore file on your client by importing the public key certificates from each node in your cluster.

   ```
   $ keytool -import -v -trustcacerts -alias node0 -file node0.cer -
   keystore .truststore
   $ keytool -import -v -trustcacerts -alias node1 -file node1.cer -
   keystore .truststore
   $ keytool -import -v -trustcacerts -alias node2 -file node2.cer -
   keystore .truststore
   ```

   b) In DevCenter, select **File** > **New** > **Connection** to open the **Connection Manager**.
   c) Add the IP addresses of the nodes in your cluster.
   d) Select **Next**.
   e) Select **This cluster requires SSL** option and enter a full path to (or navigate to) the `truststore` file on your machine.

f) Enter the truststore password.

g) Select **Try to establish a connection** link to verify that you can successfully connect to Cassandra nodes.

Client verification

2. If your cluster requires client verification, you need to perform the following additional steps:

a) Create an SSL certificate for the client host (that is, the system on which DevCenter is installed).

```
$ keytool -genkey -alias client-host -keystore .keystore
```

b) Export the client certificate.

```
$ keytool -export -alias client-host -file client-host.cer -
keystore .keystore
```

The public certificate is stored in the `client-host.cer` file.

c) Copy the public certificate and import it into the truststore on all nodes of the Cassandra cluster which you want DevCenter to be connected to.

```
$ keytool -import -v -trustcacerts -alias client-host -file /tmp/client-
host.cer -keystore /var/tmp/.truststore
```

Note: You may have to ask your cluster administrator if you do not have the proper permissions to modify the truststore file on the cluster nodes.

d) In DevCenter, right-click your connection and select **Properties** to edit the connection in the **Connection Manager**.

e) In **Advanced Settings** (under **Basic Settings**) select the **Client authentication required** option and enter location of the keystore file and keystore password.
The **Connection manager** displays an error if the keystore filepath or password is incorrect.

f) Click the **Try to establish a connection** link to verify your configuration.

g) Click **OK** at the bottom of the **Connection Manager** dialog to create or update the connection.
Now you can enable and have DevCenter communicate with your SSL-enabled cluster.

Reference

GUI

With DevCenter, you can manage multiple connections to Cassandra or DataStax Enterprise clusters and edit and run CQL scripts against those clusters.

The default main window for DevCenter has six panes arranged as left-hand and right-hand columns bordering a central, wider column. The panes are clockwise from the top left-hand corner:

Connection Manager

Used for creating, deleting, opening, and closing cluster connections.

Query Editor

A text editor for writing and executing CQL queries. When more than one file is open, it is a tabbed pane. The cluster against which the queries are executed appears in a dropdown.

Schema Navigator

Shows the schemas (keyspaces) of the currently active cluster.

CQL Scripts

Used to create, delete, or open for editing a CQL script.

Results

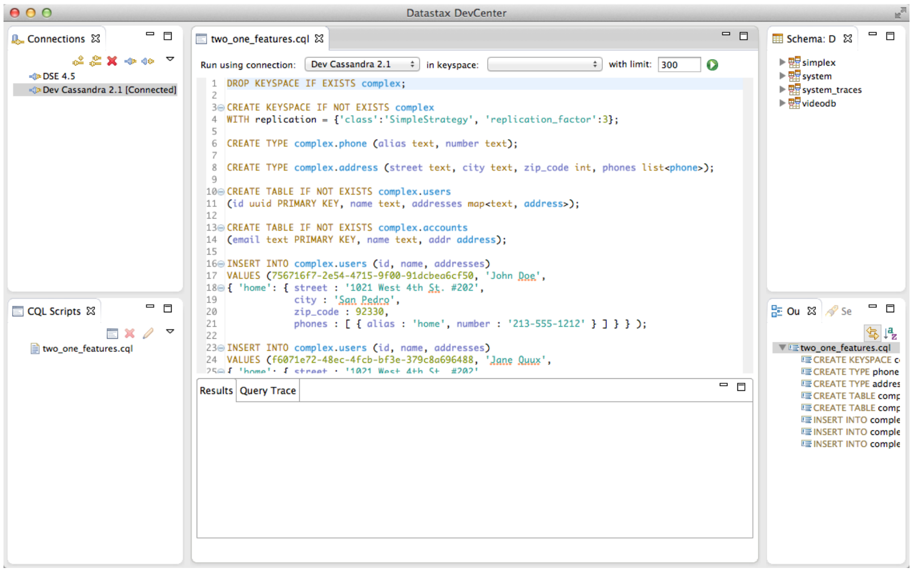
Shows the results of the last run query.

Query Trace

Shows the trace session of the last run query.

Outline

Shows the CQL statements in the currently open CQL script. Double-clicking takes you to the corresponding statement in the open file.



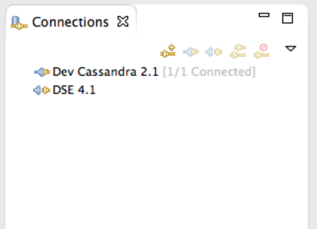
Run using connection: Dev Cassandra 2.1 in keyspace: with limit: 300

```
DROP KEYSPACE IF EXISTS complex;

CREATE KEYSPACE IF NOT EXISTS complex
WITH replication = {'class':'SimpleStrategy', 'replication_factor':3};

CREATE TYPE complex.phone (alias text, number text);

CREATE TYPE complex.address (street text, city text, zip_code int, phones list<phone>);

CREATE TABLE IF NOT EXISTS complex.users
(id uuid PRIMARY KEY, name text, addresses map<text, address>);

CREATE TABLE IF NOT EXISTS complex.accounts
(email text PRIMARY KEY, name text, addr address);

INSERT INTO complex.users (id, name, addresses)
VALUES (756716f7-2e54-4715-9f00-91dcbea6cf50, 'John Doe',
{ 'home': { street : '1021 West 4th St. #202',
            city : 'San Pedro',
            zip_code : 92330,
            phones : [ { alias : 'home', number : '213-555-1212' } ] } } );

INSERT INTO complex.users (id, name, addresses)
VALUES (f6071e72-48ec-4fcb-bf3e-379c8a696488, 'Jane Quux',
{ 'home': { street : '1021 West 4th St. #202'
```

DevCenter is built using the Eclipse RCP (Rich Client Platform) and allows you to customize the panes organization as you prefer.

Connection Manager

The **Connection Manager** provides a simple and friendly way to add and manage connections to existing Cassandra and DataStax Enterprise clusters.



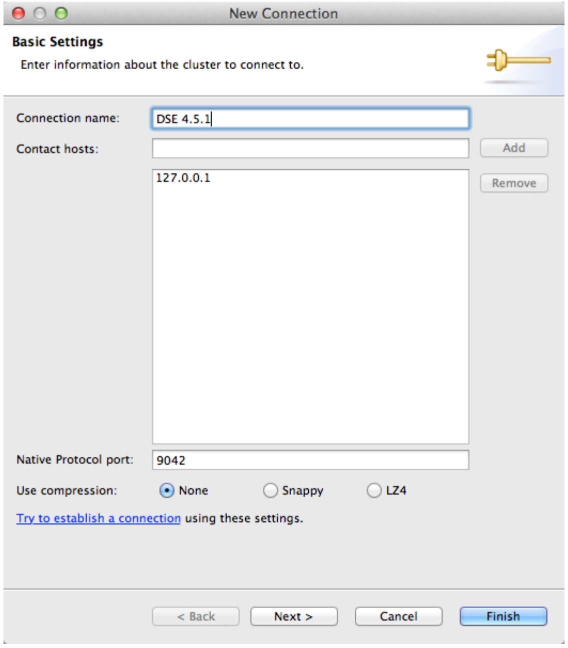
When connected, the square brackets to the right of the connection name display **number-of-connected-nodes / number-of-nodes-in-hosts-list**.

New connection

You add a new connection by either selecting **File** > **New** > **Connection**, by clicking the **New connection** button in the toolbar of the connection manager view, or typing ##+#+N (Macintosh) / Ctrl+Alt+Shift+N (Windows, Linux).

The dialog for creating a new connection allows to configure a connection to an existing cluster by entering:

- a connection name
- one or more IP addresses of nodes in the cluster
- the connection protocol (by default 9042)
- which compression to use (none, Snappy, or LZ4)
- Security configuration
  - whether the cluster is using authentication (either internal or Kerberos); if so, supply the authentication credentials
  - whether SSL is to be used for node-to-node encryption
    - The **Truststore file** contains the public certificates from the cluster nodes, and the Truststore password is used to access the password-protected Truststore file.
      - whether client authentication is to be used (that is, if `require_client_auth` is set to true in the `cassandra.yaml` file); supply the authentication credentials
      - The **Keystore file** contains the private certificate for the client, and the Keystore password is used to access the password-protected Keystore file.



New Connection

Basic Settings

Enter information about the cluster to connect to.

Connection name: DSE 4.5.1

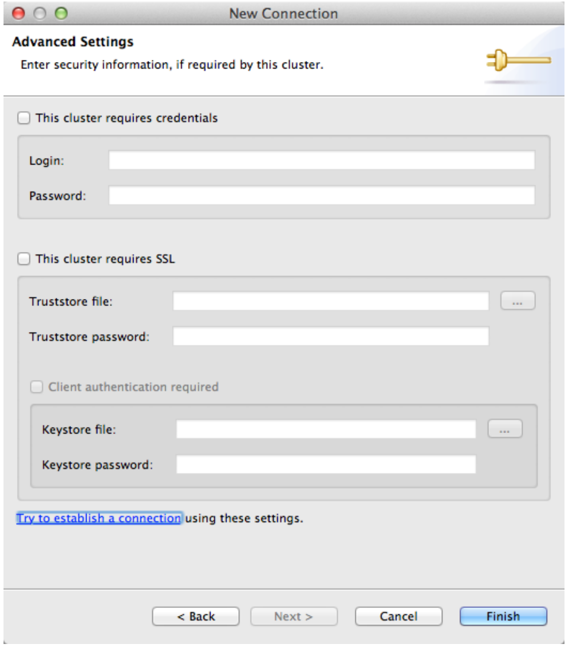
Contact hosts:

127.0.0.1

Native Protocol port: 9042

Use compression: ● None ○ Snappy ○ LZ4

Try to establish a connection using these settings.



New Connection

Advanced Settings

Enter security information, if required by this cluster.

□ This cluster requires credentials

Login:

Password:

□ This cluster requires SSL

Truststore file:

Truststore password:

□ Client authentication required

Keystore file:

Keystore password:

Try to establish a connection using these settings.

Open connection

You open a connection either by clicking the **Open connection** button in the toolbar, typing #+F3 (Macintosh) / Ctrl+F3 (Windows, Linux), or double-clicking the connection name.

Close connection

You close a connection by clicking the **Close connection** button in the toolbar or typing #+F4 (Macintosh) / Ctrl+F4 (Windows, Linux).

Clone connection

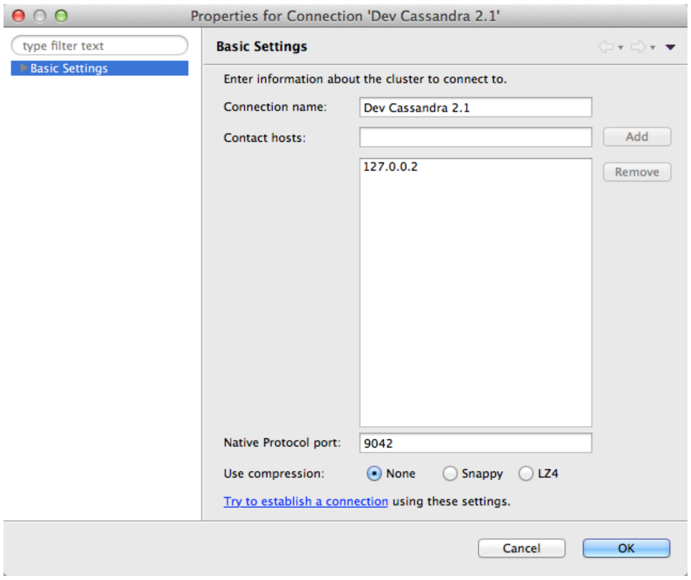
You clone a connection by clicking the **Clone connection** button in the toolbar or typing #+#+D (Macintosh) / Ctrl+Shift+D (Windows, Linux).

Delete connection

You remove a connection by clicking the **Delete connection** button in the toolbar or typing DEL (Macintosh, Windows, Linux).

Editing connection properties

You edit the properties of an existing connection by right-clicking it and selecting **Properties** or typing #+I (Macintosh) / Ctrl+I (Windows, Linux). You are presented with the **Properties for Connection '*connection-name*'** dialog.


Properties for Connection 'Dev Cassandra 2.1'

Basic Settings

Enter information about the cluster to connect to.

Connection name: Dev Cassandra 2.1

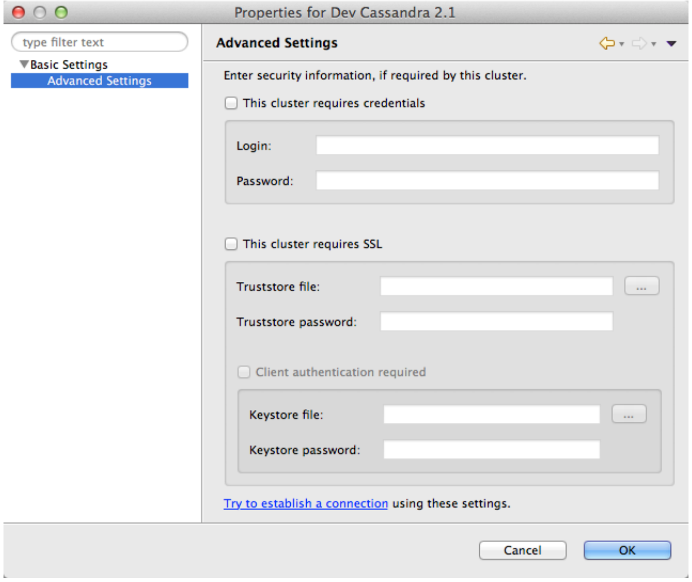
Contact hosts: 127.0.0.2

Native Protocol port: 9042

Use compression: ● None ○ Snappy ○ LZ4

Try to establish a connection using these settings.

You can edit the **Advanced Settings** by expanding the **Basic Settings** leaf.


Properties for Dev Cassandra 2.1

Advanced Settings

Enter security information, if required by this cluster.

□ This cluster requires credentials

Login:

Password:

□ This cluster requires SSL

Truststore file:

Truststore password:

□ Client authentication required

Keystore file:

Keystore password:

Try to establish a connection using these settings.

The two panes of the **Properties** dialog are the same as in the **New connection** dialog.

Query Editor

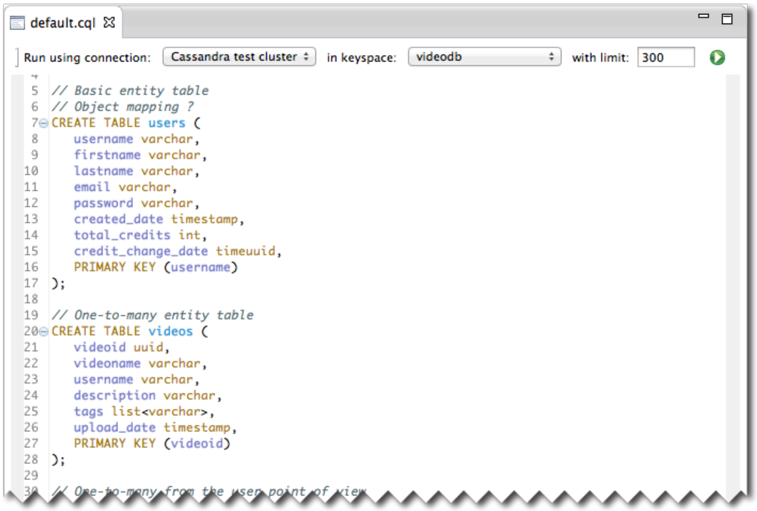
The Query Editor allows you to create, edit and execute CQL scripts.

Editing

You create and edit a CQL script in a tab. Each tab is associated with a connection and the CQL script can be executed with a single click.

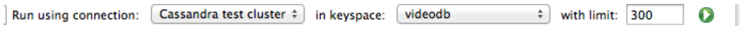
Besides normal editing functionality, such as cutting, copying, and pasting, the Query Editor also has:

- syntax highlighting for CQL
- code completion
- real-time error detection (errors are underlined in red, and hovering over the red error icon reveals the error)



```
// Basic entity table
// Object mapping ?
CREATE TABLE users (
   username varchar,
   firstname varchar,
   lastname varchar,
   email varchar,
   password varchar,
   created_date timestamp,
   total_credits int,
   credit_change_date timeuuid,
   PRIMARY KEY (username)
);

// One-to-many entity table
CREATE TABLE videos (
   videoid uuid,
   videoname varchar,
   username varchar,
   description varchar,
   tags list<varchar>,
   upload_date timestamp,
   PRIMARY KEY (videoid)
);

// One-to-many from the user point of view
```

Run using connection



You can associate a tab in the Query Editor to:

- run with any of the connections in the **Connection Manager**
- use a chosen default keyspace
- limit the number of rows returned by a statement (limit 1000 rows returned)

Execute CQL Script

You execute all the CQL statements in the current editor tab.

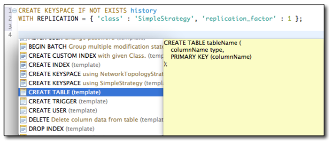
Execute selected statement

You execute a single or multiple statements by highlighting them and pressing #+F11 (Mac OS X) or Alt+F11 (Linux, Windows). The statement may be partially highlighted also. When you execute a partially highlighted statement, the entire statement is first selected and then executed.

Code completion

The query Editor has two types of code completion (activated by pressing **Ctrl-Spacebar**).

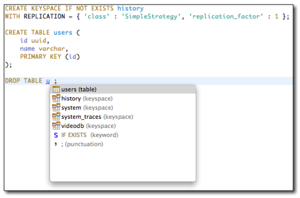
- code snippets: predefined complete CQL statements
- keywords and database identifiers: contextual code completion



```
CREATE KEYSPACE IF NOT EXISTS history
WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };
```

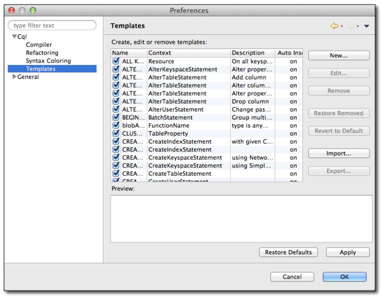
BEGIN BATCH Group multiple modification state
CREATE CUSTOM INDEX with given Class. (temp
CREATE INDEX (template)
CREATE KEYSPACE using NetworkTopologyStrat
CREATE KEYSPACE using SimpleStrategy (templ
CREATE TABLE (template)
CREATE TRIGGER (template)
CREATE USER (template)
DELETE Delete column data from table (templat
DROP INDEX (template)

```
CREATE TABLE tableName (
   columnName type,
   PRIMARY KEY (columnName)
);
```



```
CREATE KEYSPACE IF NOT EXISTS history
WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };

CREATE TABLE users (
    id uuid,
    name varchar,
    PRIMARY KEY (id)
);

DROP TABLE u ;
```

You can add your own code snippets by using the **Preferences** dialog by selecting **File** > **Preferences** > **Cql** > **Templates**.

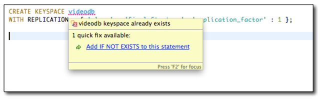


Create, edit or remove templates:

	Name	Context	Description	Auto Ins
☑	ALL K...	Resource	On all keysp...	on
☑	ALTE...	AlterKeyspaceStatement	Alter proper...	on
☑	ALTE...	AlterTableStatement	Add column	on
☑	ALTE...	AlterTableStatement	Alter colum...	on
☑	ALTE...	AlterTableStatement	Alter proper...	on
☑	ALTE...	AlterTableStatement	Drop column	on
☑	ALTE...	AlterUserStatement	Change pas...	on
☑	BEGIN...	BatchStatement	Group multi...	on
☑	blobA...	FunctionName	type is any...	on
☑	CLUS...	TableProperty		on
☑	CREA...	CreateIndexStatement	with given C...	on
☑	CREA...	CreateIndexStatement		on
☑	CREA...	CreateKeyspaceStatement	using Netwo...	on
☑	CREA...	CreateKeyspaceStatement	using Simpl...	on
☑	CREA...	CreateTableStatement		on
☑	CREA...	CreateUserStatement		on

Real-time error detection and correction

Errors are detected by DevCenter in real-time (before a script is executed) based on the selected connection, keyspace, and other metadata about the cluster. The error is indicated by a red underline. By hovering the mouse cursor over the error a list containing possible quick fixes displays.



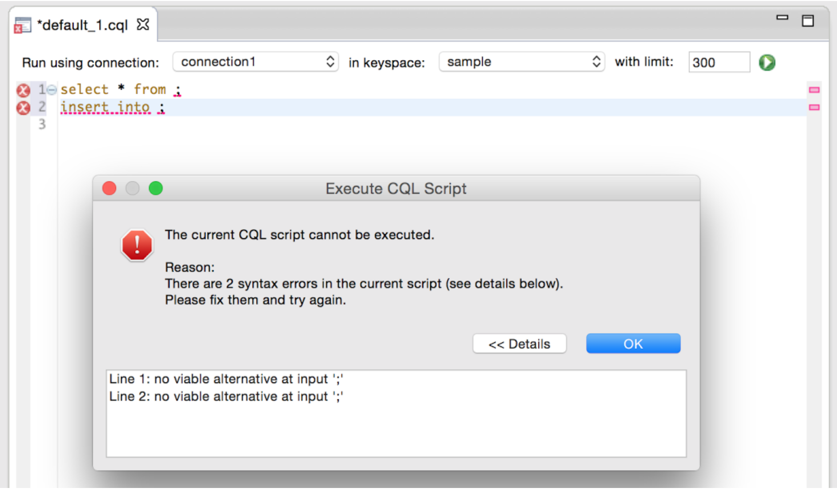
```
CREATE KEYSPACE videodb
WITH REPLICATION ... plication_factor' : 1 };
```

videodb keyspace already exists

1 quick fix available:

Preventing execution of CQL scripts containing syntax errors

If you try to execute CQL with a syntactic error in it, the code does not execute, and a dialog displays. This prevents DevCenter from sending incorrect or incomplete CQL statements. A syntax error disables the execution of the entire script.



Run using connection: connection1 in keyspace: sample with limit: 300

```
select * from ;
insert into ;
```

The current CQL script cannot be executed.

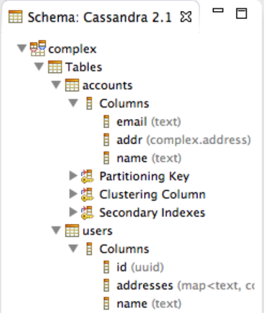
Reason:
There are 2 syntax errors in the current script (see details below).
Please fix them and try again.

```
Line 1: no viable alternative at input ';'
Line 2: no viable alternative at input ';'
```

Schema Navigator

The Schema Navigator provides a tree control that displays the structure and details of the schema (for example, keyspaces, tables) for the current connection.

It displays the schema for the current active tab in the **Query Editor** pane.



Schema navigator contextual menu

Right-clicking in the Schema navigator pane (with no node selected) displays a contextual menu. The menu items are:

- **New Keyspace**: displays the **New Keyspace** wizard
- **New Table**: displays the **New Table** wizard

Keyspace node contextual menu

Right-clicking on a keyspace node in the Schema navigator pane on a selected node displays a contextual menu. The menu items are:

- **New Keyspace**: displays the **New Keyspace** wizard
- **New Table**: displays the **New Table** wizard
- **Clone Keyspace**: displays the **New Keyspace** wizard with values of fields set to the selected keyspace's settings
- **Edit Keyspace**: displays the **Edit Keyspace** wizard (which contains the same fields as the **New Keyspace** wizard) for editing the selected keyspace's settings

Table node contextual menu

Right-clicking on a table node in the Schema navigator pane on a selected node displays a contextual menu. The menu items are:

- **New Keyspace**: displays the **New Keyspace** wizard
- **New Table**: displays the **New Table** wizard
- **Clone Table**: displays the **New Table** wizard with values of fields set to the selected table's settings
- **Edit Table**: displays the **Edit Table** wizard (which contains the same fields as the **New Table** wizard) for editing the selected table's settings

When editing a table, only primary key column names can be changed. Likewise, when editing a table's column, only non-primary key columns may have their types changed.

Wizards resynchronize with database after partial failure

If more than one statement is generated (typically in **Edit Keyspace** or **Edit Table** wizards), a statement may fail although previous ones succeeded. In this situation, the wizard dialog is kept open, and its contents are resynchronized with the database.

New Keyspace wizard

The **New Keyspace** wizard generates a valid CQL statement that creates a new keyspace in the specified cluster.

To display the **New Keyspace** wizard:

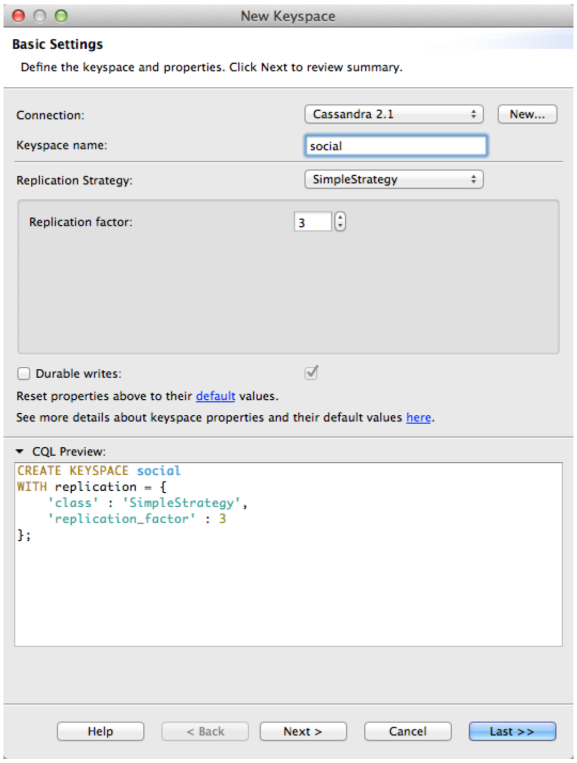
- select the **File** > **New** > **Keyspace** menu item
- right-click in the schema navigator and select **New Keyspace**
- press the key shortcut ##+#+K(Mac OS X) or `Ctrl+Alt+Shift+K` (Linux / Windows)

You navigate between panels using the **Previous** and **Next** buttons. A **Last** button also skips from the current panel to the **Summary** panel, if you won't be editing any fields on those panels.

The first panel (**Basic Settings**) of the **New Keyspace** wizard displays. There are two panes on this panel: one that displays keyspace settings and allows editing of them, and a **CQL Preview** pane that displays the CQL statement for creating the keyspace generated from the values in the first pane.

The editable fields are:

- **Connection**: choose an existing connection or create a new one
- **Keyspace name**: type in a valid keyspace name (invalid names are noted in red)
- **Replication strategy**: which data replication strategy class (`SimpleStrategy` single data center or `NetworkTopologyStrategy` multiple data centers) to use; the strategy determines where replicas are placed
    - **Replication factor**: which replication factor to use; the factor is the number of replicas across the cluster
- **Durable writes**: by setting this property, data written to the keyspace bypasses the commit logs



Basic Settings

Define the keyspace and properties. Click Next to review summary.

Connection: Cassandra 2.1

Keyspace name: social

Replication Strategy: SimpleStrategy

Replication factor: 3

□ Durable writes: ☑

Reset properties above to their default values.

See more details about keyspace properties and their default values here.

CQL Preview:

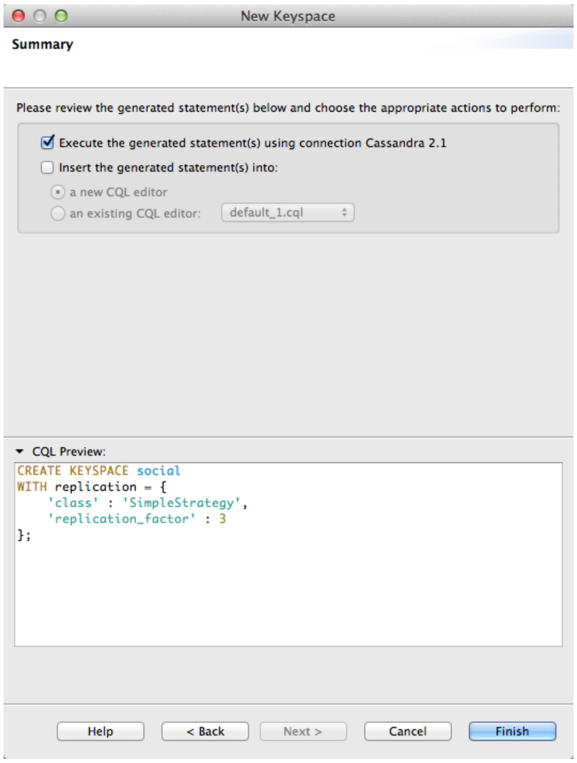
```
CREATE KEYSPACE social
WITH replication = {
    'class' : 'SimpleStrategy',
    'replication_factor' : 3
};
```

After selecting **Next**, the second panel (**Summary**) of the **New Keyspace** wizard displays.

The editable fields are:

- Whether to execute the generated statement(s) using the current connection or not
- Whether to insert the generated statement(s) into

- a new CQL editor pane
- an existing CQL editor pane



Please review the generated statement(s) below and choose the appropriate actions to perform:

☑ Execute the generated statement(s) using connection Cassandra 2.1

□ Insert the generated statement(s) into:

● a new CQL editor

○ an existing CQL editor: default_1.cql

```
CREATE KEYSPACE social
WITH replication = {
    'class' : 'SimpleStrategy',
    'replication_factor' : 3
};
```

Both panels have an uneditable **CQL preview** pane showing the CQL statement to be generated according to the current values of the wizard's editable fields.

New Table wizard

The **New Table** wizard generates a valid CQL statement that creates a new table in the specified keyspace.

The **New Table** wizard consists of five panels:

- **Basic Settings**
- **Primary Key Settings**
- **Advanced Settings**
- **Compaction**
- **Summary**

You navigate between panels using the **Previous** and **Next** buttons. A **Last** button also skips from the current panel to the **Summary** panel, if you won't be editing any fields on those panels.

To display the **New Table** wizard:

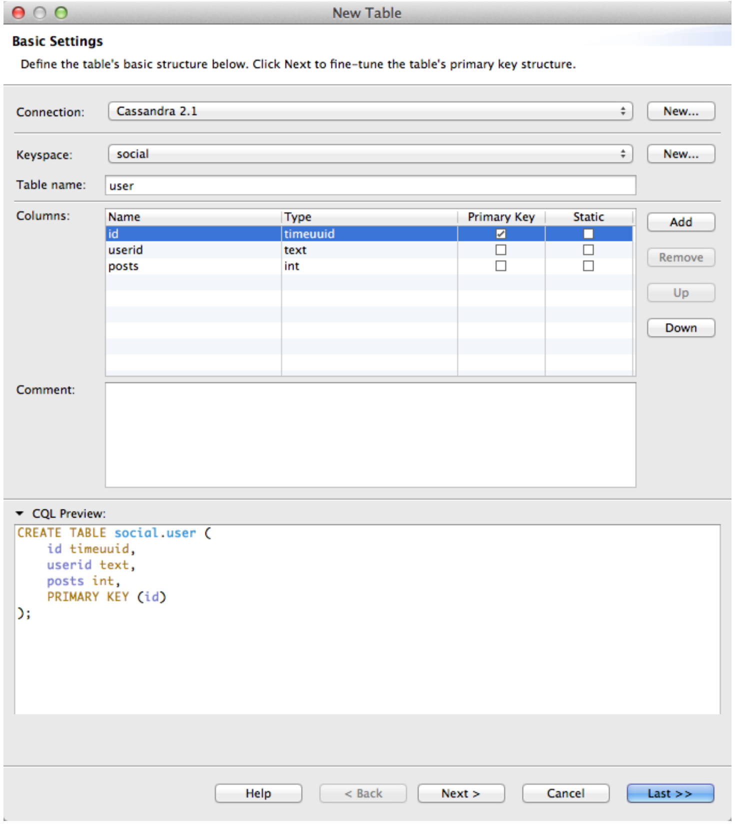
- select the **File** > **New** > **Table** menu item
- right-click in the schema navigator and select **New Table**
- press the key shortcut ##+#+T(Mac OS X) or `Ctrl+Alt+Shift+T` (Linux / Windows)

The first panel (**Basic Settings**) of the **New Table** wizard displays. There are four panes on this panel which display editable table settings and a **CQL Preview** pane that displays the CQL statement for creating the table generated from the values in the preceding three panes.

The editable fields are:

- **Connection**: choose an existing connection or create a new one
- **Keyspace**: choose an existing keyspace or create a new one

- **Table name**: type in a valid table name (invalid names are noted in red)
- **Columns**: add, remove, or move up or down a column
- **Comment**: add or edit a comment property for the table



Basic Settings

Define the table's basic structure below. Click Next to fine-tune the table's primary key structure.

Connection: Cassandra 2.1

Keyspace: social

Table name: user

Columns:

Name	Type	Primary Key	Static
id	timeuuid	☑	☐
userid	text	☐	☐
posts	int	☐	☐

Comment:

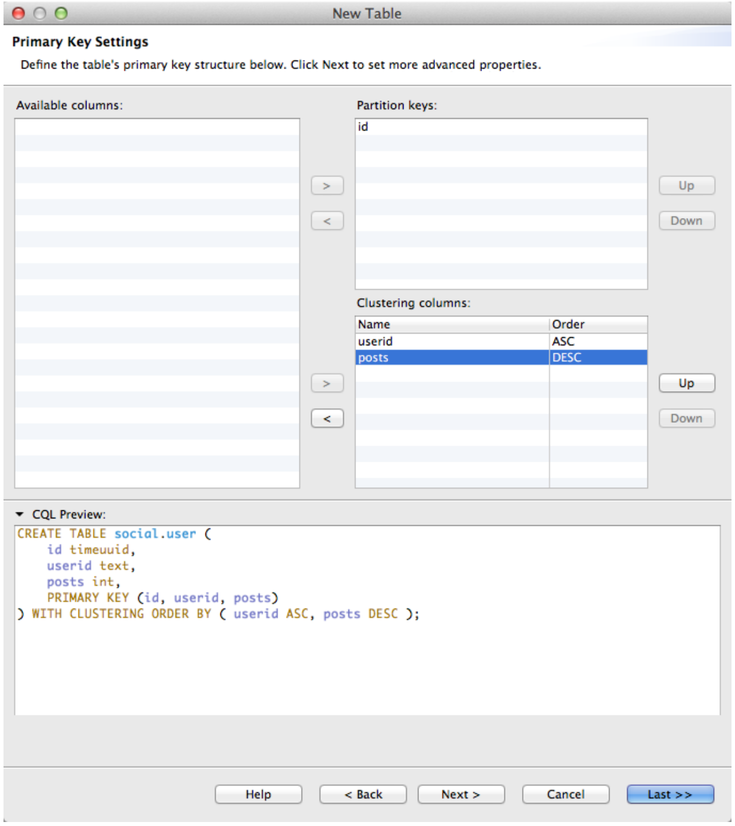
CQL Preview:

```
CREATE TABLE social.user (
    id timeuuid,
    userid text,
    posts int,
    PRIMARY KEY (id)
);
```

After selecting **Next**, the second panel (**Primary Key Settings**) of the **New Table** wizard displays.

The panel allows you to define the primary key structure for the table. They can be simple or compound. The editable fields are:

- **Available columns**: displays the available columns
- **Partition keys**: displays the partition key for the table (which determines the node on which the data is stored)
- **Clustering columns**: displays the clustering columns for the table



Primary Key Settings

Define the table's primary key structure below. Click Next to set more advanced properties.

Available columns:

Partition keys:

id

Clustering columns:

Name	Order
userid	ASC
posts	DESC

CQL Preview:

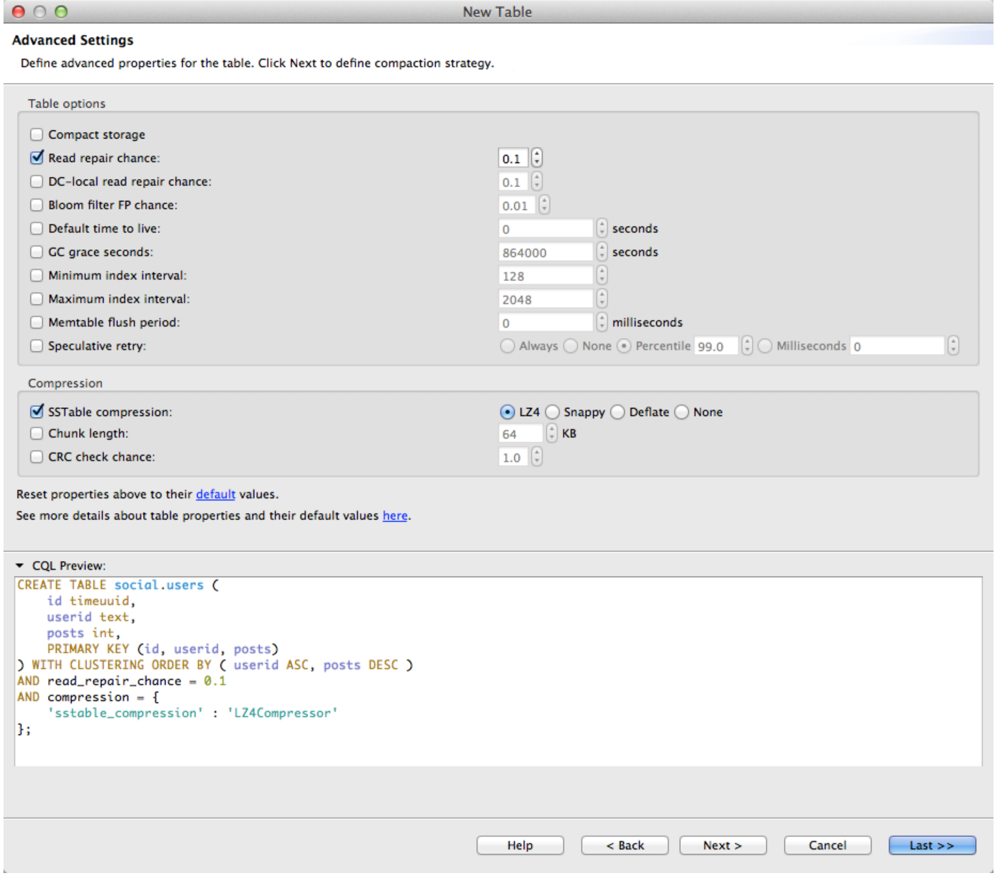
```
CREATE TABLE social.user (
    id timeuuid,
    userid text,
    posts int,
    PRIMARY KEY (id, userid, posts)
) WITH CLUSTERING ORDER BY ( userid ASC, posts DESC );
```

After selecting **Next**, the final panel (**Advanced Settings**) of the **New Table** wizard displays.

The editable fields are:

- **Table options**
    - **Compact storage**: whether to store data so as to conserve disk space
    - **Read repair chance**: the basis for invoking read repairs on reads in clusters (a value between 0 and 1)
    - **DC-local read repair chance**: the probability of read repairs being invoked over all replicas in the current data center
    - **Bloom filter FP chance**: the desired false-positive probability for SSTable Bloom filters
    - **Replicate on write** (Cassandra 2.0 only): if true, replicates writes to all affected replicas regardless of the consistency level specified by the client for a write request
    - **Default time to live** The default expiration time in seconds for a table. Used in MapReduce scenarios when you have no control of TTL.
    - **GC grace seconds** Specifies the time to wait before garbage collecting tombstones (deletion markers). The default value allows a great deal of time for consistency to be achieved prior to deletion. In many deployments this interval can be reduced, and in a single-node cluster it can be safely set to zero.
    - **Minimum index interval** The minimum value to control the sampling of entries from the partition index, configure the sample frequency of the partition summary by changing these properties.
    - **Maximum index interval** The maximum value to control the sampling of entries from the partition index, configure the sample frequency of the partition summary by changing these properties.
    - **Index interval** (Cassandra 2.0 only): The interval value to control the sampling of entries from the partition index.
    - **Memtable flush period** Forces flushing of the memtable after the specified time in milliseconds elapses.
    - **Speculative retry** Overrides normal read timeout when `read_repair_chance` is not 1.0, sending another request to read.

- **Compression**
  - **SSTable compression**: the compression to use (`LZ4`, `Snappy`, `Default`, or `None`)
  - **Chunk length**: size of block used by compression scheme
  - **CRC check chance**: the probability with which checksums are checked during read
- Reset properties above to their **default** values.
- See more details above table properties and their default **here**.



Advanced Settings

Define advanced properties for the table. Click Next to define compaction strategy.

Table options

□ Compact storage
☑ Read repair chance: 0.1
□ DC-local read repair chance: 0.1
□ Bloom filter FP chance: 0.01
□ Default time to live: 0 seconds
□ GC grace seconds: 864000 seconds
□ Minimum index interval: 128
□ Maximum index interval: 2048
□ Memtable flush period: 0 milliseconds
□ Speculative retry: ○ Always ○ None ● Percentile 99.0 ○ Milliseconds 0

Compression

☑ SSTable compression: ● LZ4 ○ Snappy ○ Deflate ○ None
□ Chunk length: 64 KB
□ CRC check chance: 1.0

Reset properties above to their default values.

See more details about table properties and their default values here.

CQL Preview:

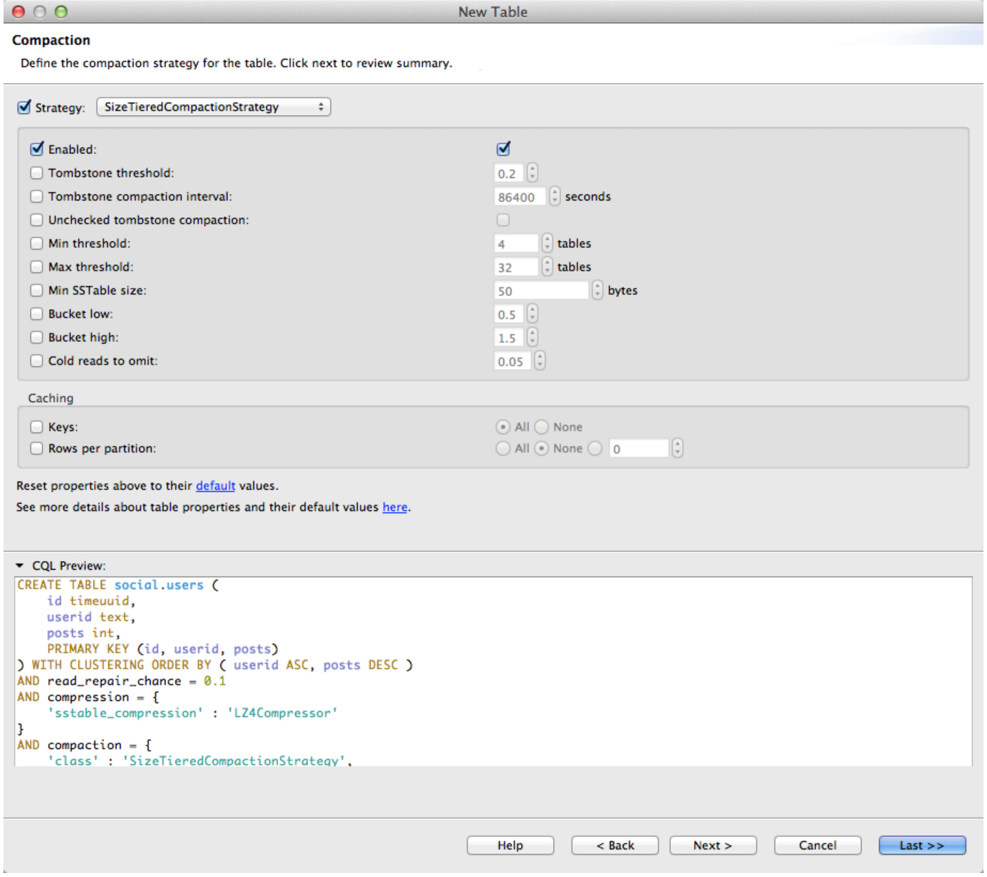
```
CREATE TABLE social.users (
    id timeuuid,
    userid text,
    posts int,
    PRIMARY KEY (id, userid, posts)
) WITH CLUSTERING ORDER BY ( userid ASC, posts DESC )
AND read_repair_chance = 0.1
AND compression = {
    'sstable_compression' : 'LZ4Compressor'
};
```

After selecting **Next**, the **Compaction** panel of the **New Table** wizard displays.

The editable fields are:

- **Strategy**: sets the compaction strategy class to use
  - **Enabled**
  - **Tombstone threshold**
  - **Tombstone compaction interval**
  - **Unchecked tombstone compaction**
  - **Min threshold**
  - **Max threshold**
  - **Min SSTable size**
  - **Bucket low**
  - **Bucket high**
  - **Cold reads to omit**
- **Caching** (Cassandra 2.1 and higher)
  - **Keys**: `ALL` or `NONE`
  - **Rows per partition**: number of CQL rows, `NONE`, or `ALL`
- **Caching** (Cassandra 1.2 and 2.0)

- **Keys**: `ALL`, `KEYS_ONLY`, `ROWS_ONLY`, or `NONE`
- Reset properties above to their **default** values.
- See more details above table properties and their default **here**.



Compaction

Define the compaction strategy for the table. Click next to review summary.

☑ Strategy: SizeTieredCompactionStrategy

☑ Enabled:	☑
□ Tombstone threshold:	0.2
□ Tombstone compaction interval:	86400 seconds
□ Unchecked tombstone compaction:	□
□ Min threshold:	4 tables
□ Max threshold:	32 tables
□ Min SSTable size:	50 bytes
□ Bucket low:	0.5
□ Bucket high:	1.5
□ Cold reads to omit:	0.05

Caching

□ Keys:	● All ○ None
□ Rows per partition:	○ All ● None ○ 0

Reset properties above to their default values.

See more details about table properties and their default values here.

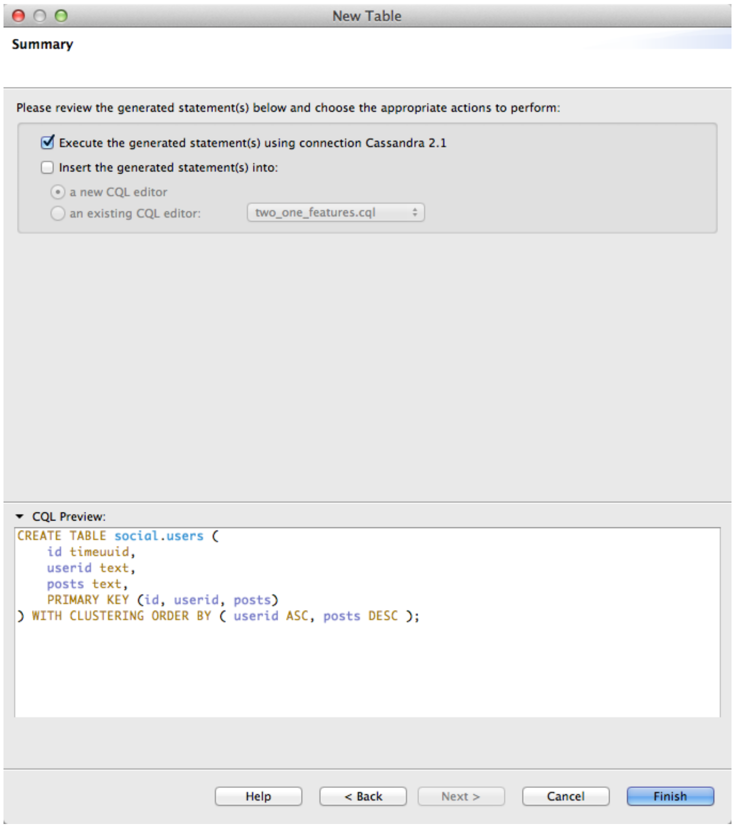
CQL Preview:

```
CREATE TABLE social.users (
    id timeuuid,
    userid text,
    posts int,
    PRIMARY KEY (id, userid, posts)
) WITH CLUSTERING ORDER BY ( userid ASC, posts DESC )
AND read_repair_chance = 0.1
AND compression = {
    'sstable_compression' : 'LZ4Compressor'
}
AND compaction = {
    'class' : 'SizeTieredCompactionStrategy',
```

After selecting **Next**, the final **Summary** panel of the **New Table** wizard displays.

The editable fields are:

- Execute the generated statement(s) using connection Cassandra 2.1
- Insert the generated statements into:
  - a new CQL editor
  - an existing CQL editor



Summary

Please review the generated statement(s) below and choose the appropriate actions to perform:

☑ Execute the generated statement(s) using connection Cassandra 2.1

□ Insert the generated statement(s) into:

● a new CQL editor

○ an existing CQL editor: two_one_features.cql

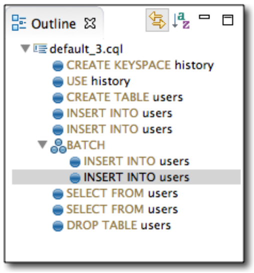
CQL Preview:

```
CREATE TABLE social.users (
    id timeuuid,
    userid text,
    posts text,
    PRIMARY KEY (id, userid, posts)
) WITH CLUSTERING ORDER BY ( userid ASC, posts DESC );
```

Outline

The Outline lists the statements that make up the CQL script loaded in the current tab in the Query Editor.

The Outline allows for quick navigation within large CQL scripts by selecting the statement in the tree that section of the **Query Editor** where the statement occurs is displayed.



```
▼ default_3.cql
    CREATE KEYSPACE history
    USE history
    CREATE TABLE users
    INSERT INTO users
    INSERT INTO users
  ▼ BATCH
      INSERT INTO users
      INSERT INTO users
    SELECT FROM users
    SELECT FROM users
    DROP TABLE users
```

Link to Editor

Toggles between whether the CQL statement is linked to its location in the Query Editor tab or not.

Order Alphabetically

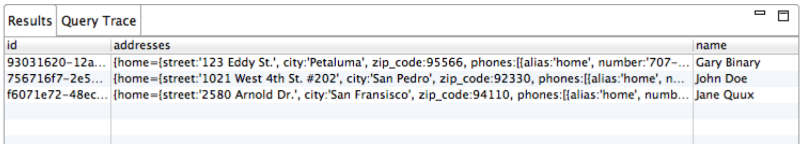
Toggles between whether the CQL statements are listed in alphabetical order or in the order which they occur in the CQL script in.

Results

The **Results** pane shows the latest results from a CQL query executed in the Query Editor tab in tabular form.

The **Results** pane has two parts:

- a tabular grid which displays results from the last query executed
- a status bar which displays pertinent information about executed statements



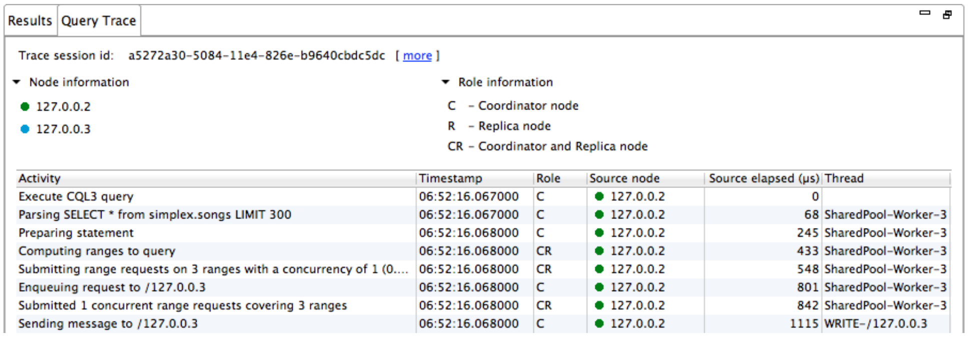
id	addresses	name
93031620-12a...	{home={street:'123 Eddy St.', city:'Petaluma', zip_code:95566, phones:[{alias:'home', number:'707-...	Gary Binary
756716f7-2e5...	{home={street:'1021 West 4th St. #202', city:'San Pedro', zip_code:92330, phones:[{alias:'home', n...	John Doe
f6071e72-48ec...	{home={street:'2580 Arnold Dr.', city:'San Fransisco', zip_code:94110, phones:[{alias:'home', numb...	Jane Quux

In the grid display, you can:

- reorder and resize columns
- sort rows by a column by clicking on the column heading
- copy the data from the results by
  - a single selected row (click on the row)
  - multiple rows #+click (Macintosh) / Ctrl+Click (Windows / Linux)
  - all of the rows (either as CSV or an INSERT statement)

Query Trace

The **Query Trace** pane shows the trace session from the last CQL query executed in the Query Editor tab in tabular form.



Trace session id: a5272a30-5084-11e4-826e-b9640cbdc5dc [more]

Node information

● 127.0.0.2

● 127.0.0.3

Role information

C – Coordinator node

R – Replica node

CR – Coordinator and Replica node

Activity	Timestamp	Role	Source node	Source elapsed (µs)	Thread
Execute CQL3 query	06:52:16.067000	C	● 127.0.0.2	0	
Parsing SELECT * from simplex.songs LIMIT 300	06:52:16.067000	C	● 127.0.0.2	68	SharedPool-Worker-3
Preparing statement	06:52:16.068000	C	● 127.0.0.2	245	SharedPool-Worker-3
Computing ranges to query	06:52:16.068000	CR	● 127.0.0.2	433	SharedPool-Worker-3
Submitting range requests on 3 ranges with a concurrency of 1 (0....	06:52:16.068000	CR	● 127.0.0.2	548	SharedPool-Worker-3
Enqueuing request to /127.0.0.3	06:52:16.068000	C	● 127.0.0.2	801	SharedPool-Worker-3
Submitted 1 concurrent range requests covering 3 ranges	06:52:16.068000	CR	● 127.0.0.2	842	SharedPool-Worker-3
Sending message to /127.0.0.3	06:52:16.068000	C	● 127.0.0.2	1115	WRITE-/127.0.0.3

See also:

- Tracing
- Request tracing in Cassandra 1.2
- Advanced request tracing

.

Trace session_id

The Trace `session_id` uniquely identifies the tracing session. Trace data is saved in Cassandra for 24 hours and can be queried using statements found by clicking the 'more' link. For example,

```
Tracing session information is stored for 24 hours. Session and session
 event data can be queried with:
SELECT * FROM system_traces.sessions WHERE session_id =
 d6aa92a0-4f48-11e4-8d9b-65dde8fcb188;
SELECT * FROM system_traces.events WHERE session_id =
 d6aa92a0-4f48-11e4-8d9b-65dde8fcb188;
```

Node information

The **Node information** section lists each node participating in the query along with a color coding for easier identification in the trace table results.

Role information

The **Role Information** section describes the roles that participating nodes may play in each trace event:

C

Coordinator node

R

Replica node

CR

Coordinator and Replica node

.

Trace columns

The columns in the **Query trace** are:

Activity

Description of the activity in the trace event.

Timestamp

The timestamp of the event on the source node.

Role

The role of the node for this event (Coordinator, Replica, Coordinator and Replica).

Source node

The source node where the event occurred.

Source elapsed (µs)

Elapsed time of the event on the source node in microseconds.

Thread

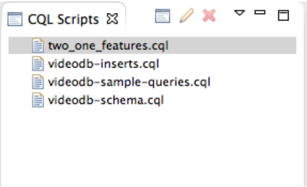
The name of the thread on the source node that executed the event.

CQL Scripts

The **CQL Scripts** view provides a simple file manager for creating, editing, and deleting CQL scripts.

There are two types of files: scratchpads (new scripts having a default name) and already saved scripts. DevCenter saves CQL scripts (both scratchpads and named scripts) in your DevCenter workspace under these directories:

- scratchpads: `user-home /.devcenter/DevCenter/.default`
- CQL scripts: `user-home /.devcenter/DevCenter/CQLScripts`



two_one_features.cql
videodb-inserts.cql
videodb-sample-queries.cql
videodb-schema.cql

You can select one or more CQL scripts by typing #+click (Macintosh) / Ctrl+Click (Windows / Linux).

Note: When you open a CQL script for editing using **File** > **Open File ...**, any changes you make are to your workspace copy of the CQL script and not to the original file.

New CQL script

You create a new CQL script by selecting the **File** > **New CQL Script**, typing #+#+N (Macintosh) / Ctrl+Shift+N (Windows, Linux), or clicking the **New CQL Script** icon in the toolbar of this view.

Open CQL script

You edit an existing script by clicking the **Open CQL script** action or typing #+F3 (Macintosh) / Ctrl+F3 (Windows, Linux).

You can also import an existing script by selecting **File** > **Open File**.

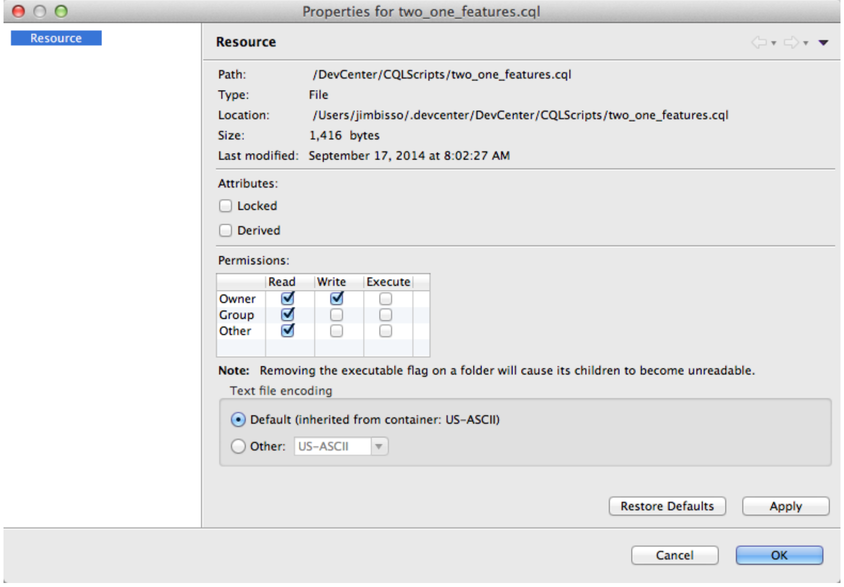
Delete CQL script

You delete a script by clicking the **Delete CQL script** icon or typing DEL (Macintosh,, Windows, Linux).

Note: in case you must delete your DevCenter workspace directory, make sure you have copied or moved the existing CQL scripts to a different location on your disk.

Properties

You view a CQL script's properties by right-clicking on the script and selecting **Properties** in the contextual menu or typing #+I (Macintosh) / Ctrl+I (Windows, Linux).



Path: /DevCenter/CQLScripts/two_one_features.cql
Type: File
Location: /Users/jimbisso/.devcenter/DevCenter/CQLScripts/two_one_features.cql
Size: 1,416 bytes
Last modified: September 17, 2014 at 8:02:27 AM

Attributes:

☐ Locked

☐ Derived

Permissions:

	Read	Write	Execute
Owner	☑	☑	☐
Group	☑	☐	☐
Other	☑	☐	☐

Note: Removing the executable flag on a folder will cause its children to become unreadable.

Text file encoding

● Default (inherited from container: US-ASCII)

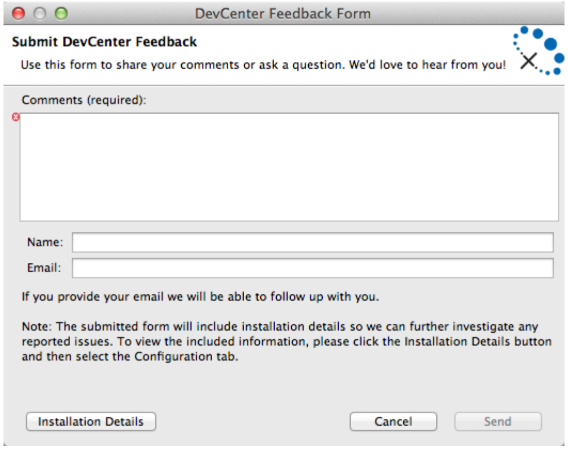
○ Other: US-ASCII

You see all the file information (for example, the time the file was last modified and its read, write, execute permissions).

Submit Feedback form

A form for submitting feedback on the DevCenter IDE is included under the Help menu.

By selecting **Help** > **Submit Feedback**, you can send feedback directly to DataStax, The only required field is the **Comment** textarea. The submitted feedback includes installation details for the computer you are running DevCenter on.



Submit DevCenter Feedback

Use this form to share your comments or ask a question. We'd love to hear from you!

Comments (required):

Name:

Email:

If you provide your email we will be able to follow up with you.

Note: The submitted form will include installation details so we can further investigate any reported issues. To view the included information, please click the Installation Details button and then select the Configuration tab.

Keyboard shortcuts

Macintosh OS X	Windows or Linux	Description
General		
#+N	Ctrl+N	Generic New wizard.
Editing and running CQL scripts		
#+#+N	Ctrl+Shift+N	New CQL script.
##+#+K	Ctrl+Alt+Shift+K	New keyspace.
##+#+T	Ctrl+Alt+Shift+T	New table.
#+F3	Ctrl+F3	Open CQL script.
DEL	DEL	Delete CQL script.
#+I	Ctrl+I	Open CQL script properties.
#+F11	Alt+F11	Execute current CQL script. (If statements are selected then only execute those statements.)
#+A	Ctrl+A	Select all CQL scripts.
#+Space	Ctrl+Space	Display auto-completion popup.
Connections panel		
##+#+N	Ctrl+Alt+Shift+N	New connection.
#+F3	Ctrl+F3	Open connection.
#+F4	Ctrl+F4	Close connection
#+#+D	Ctrl+Shift+D	Clone connection.
DEL	DEL	Delete connection.
CQL scripts panel		
#+.	Ctrl+.	Jump to next warning / error.
#+#+.	Ctrl+Shift+.	Jump to previous warning / error.

Macintosh OS X	Windows or Linux	Description
`#+1`	`Ctrl+1`	Display quick fixes popup.
`#+#+F`	`Ctrl+Shift+F`	Format code.
`#+click`	`Ctrl+click`	Multiple row selection in results pane.

FAQ

Why does DevCenter on Mac OS Yosemite need Java 6?

When launching DevCenter on Mac OS X Yosemite, you might see an error message stating: "To open 'DevCenter' you need to install the legacy Java SE 6 runtime. Click 'More Info...' to visit the legacy Java SE 6 download website". Although DevCenter is compatible with any recent Java version, Mac OS X Yosemite might force you to use Java SE 6. If this happens, simply follow the link in the message and install Java SE 6.

Why am I seeing the “Failed to load the JNI shared library" error when starting DevCenter?

It's likely to be because you're using a 32-bit application on a 64-bit JVM or vice versa.

Why aren't all of my database objects showing up in the auto-completion popup?

To retrieve an up-to-date view of the database schema, DevCenter requires an active connection to the cluster, so if you are working offline, only the database objects defined in the current script will be available for auto-completion.

I just got "an internal error occurred during Xtext validation" dialog. What can I do about it?

Increase JVM MaxHeapSize by adding `-Xmx512m` or `-Xmx1024m` to the *`DevCenter_install_dir/`*`DevCenter.app/Contents/MacOS/DevCenter.ini` (on Macintosh) or *`DevCenter_install_dir/`*`DevCenter.ini` (on Windows / Linux).

Why can't I see my settings from DevCenter alpha-0?

In order to support connections that require authentication or use compression, we’ve made some changes to the settings files. There were also other changes to the way we store preferences like color schemes, font settings, etc. Unfortunately we haven’t been able to find a solution that worked reliably and offered a friendly migration of the old type of settings.

Tips for using DataStax documentation

Navigating the documents

To navigate, use the table of contents or search in the left navigation bar. Additional controls are:

|⇄|	Hide or display the left navigation.
«»	Go back or forward through the topics as listed in the table of contents.
✎	Toggle highlighting of search terms.
🖶	Print page.
🐦	See doc tweets and provide feedback.
⁞\	Grab to adjust the size of the navigation pane.
¶	Appears on headings for bookmarking. Right-click the ¶ to get the link.
⊕	Toggles the legend for CQL statements and nodetool options.

Other resources

You can find more information and help at:

- Documentation home page
- Datasheets
- Webinars
- Whitepapers
- Developer blogs
- Support