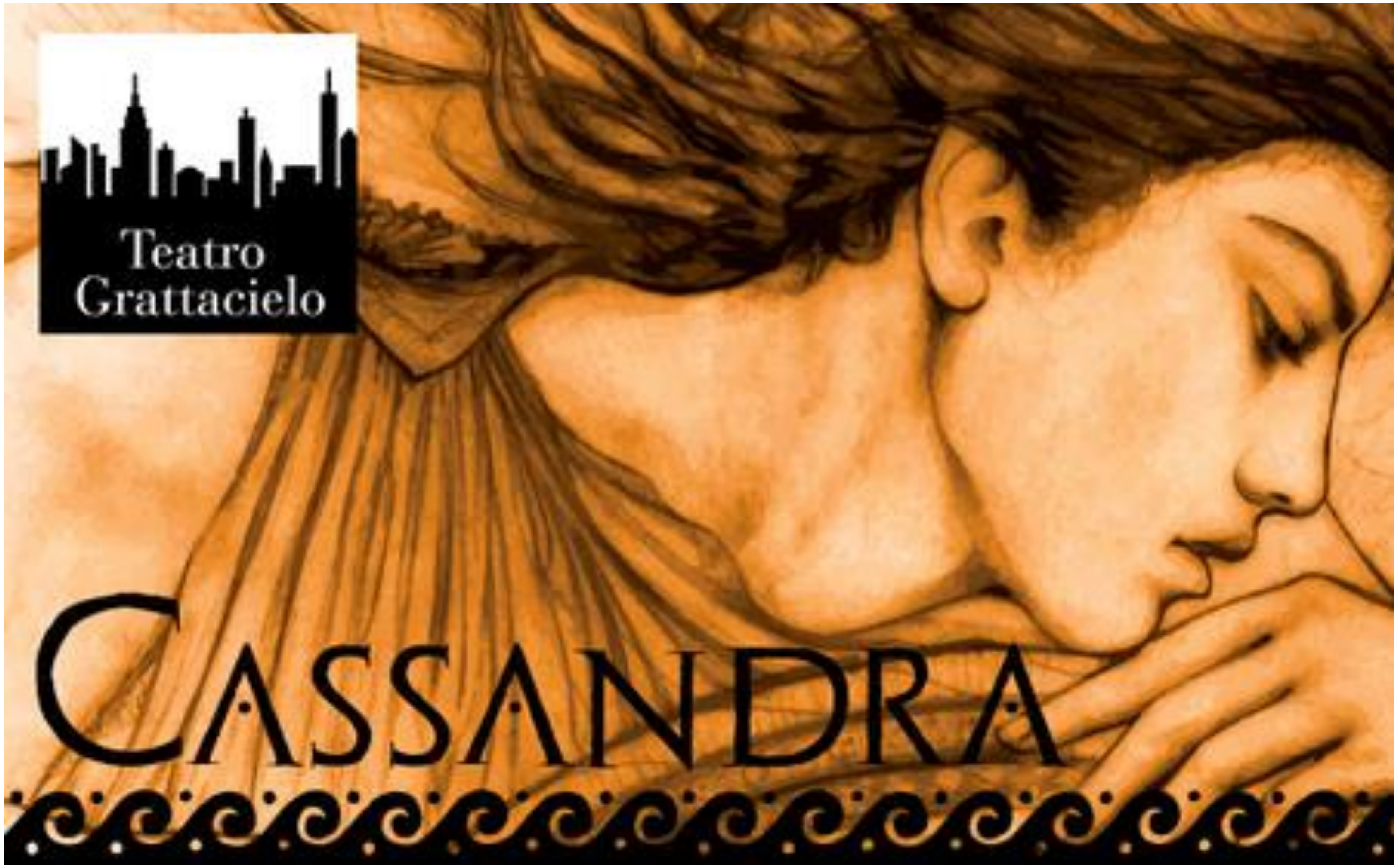
VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Cassandra Tools and Config Files

Lecturer : Dr. Pavle Mogin

SWEN 432
Advanced Database Design and Implementation

Cassandra the Fortune Teller



Plan for Tools and Config Files

- Cassandra Cluster Manager (CCM)
  - Cluster Commands
  - Node Commands
- Cassandra Files
  - `cassandra.yaml` configuration file
  - `cassandra-topology.properties` file used by `PropertyFileSntch`
- `cqlsh`
  - Commands
- Node Tool

CCM Introduction

- Cassandra Cluster Manager is a tool containing scripts and libraries to create, launch, manipulate, and remove Cassandra clusters on local host
- CCM is made as a vehicle for developing and testing Cassandra clusters and not as a production environment
  - A cluster created with CCM provides all of the functionality of a real Apache Cassandra cluster
- CCM has been already installed on our school network for use in SWEN432 assignments
- CCM supports a great number of commands
  - Some apply to the whole cluster,
  - Some to individual nodes, and
  - Some can work either way

CCM Commands

- CCM is used from the command line
- To get the list of all available CCM commands

```
[~] % ccm -help
```

- To learn about a particular command

```
[~] % ccm <command> -help
```

- Cluster wide commands are invoked by

```
[~] % ccm <cluster_cmd> [options]
```

- Node commands are invoked by

```
[~] % ccm <node_name> <node_cmd> [options]
```

- CCM can be used to create several clusters
  - After creating a new cluster, `ccm` makes it current
  - All subsequent `ccm` commands are applied on the current cluster

CCM Create Command

- Usage: `ccm create [options] cluster_name`
- Options (the only two we care about):

  `-n NODES,` and

  `-s` (start nodes)
- Description:
  - Creates, populates, and starts a new cluster with `NODES` number of nodes (where `NODES` is a single `int` or a sum of elements of a colon-separated list of `int`-s for multi data centre setups)
- Example:

```
[~] % ccm create -n 3:4 -s multi_dc
Current cluster is now: multi_dc
```

  - The two data centres will be named by `ccm` as `dc1` and `dc2`
  - `dc1` will have 3 and `dc2` will have 4 nodes
  - The `multi_dc` cluster will have 7 nodes

Files Created by CCM

- In the course of creating a cluster, ccm creates a number of files
  - The directory `~/cassandra-training/<cluster_name>` contains the `cluster.conf` file and a subdirectory for each node
  - The `cluster.conf` file contains information about the end point snitch, cluster name, and seed nodes
  - Each node subdirectory contains the `node.conf` file and a number of subdirectories, including `conf` subdirectory
  - The `node.conf` file contains information about:
    - Data center name, that is by default `dcj`, and `j∈{1,… }`
    - Node name that is `nodei`, where `i∈ {1, …, NODES}`
    - Node IP address that is 127.0.0.i
    - Node JMX port that is 7i00
    - Port 9160 is used for trift to listen for clients on, port 7000 for the internal cluster communication on all nodes, and port 9042 for CQL native transport to listen for clients on

`conf` Subdirectory Files

- We consider only some of files contained in `conf`:
  - The `cassandra.yaml` is a compulsory Cassandra configuration file that contains information about:
    - The cluster name, node IP address and common port numbers, dynamic snitch, endpoint snitch, hinted handoff status, the partitioner, seeds, tombstone limits, and different timeout limits
  - The `logback.xml` is another compulsory Cassandra server configuration file
  - The `cassandra-topology.properties` is an optional file used by `PropertyFileSnitch`
    - It contains cluster topology information in the form

      (node IP address = data centre name, rack identifier)
  - The `cassandra-rackdc.properties` is an optional file used by `GossipingPropertyFileSnitch`

Various CCM Cluster Commands (1)

- CCM can create multiple clusters, the command

```
ccm list
```

  returns the list of existing clusters with a star denoting the current cluster
- The command

```
ccm switch cluster_name
```

  makes `cluster_name` the current cluster
- The command

```
ccm status
```

  displays the status of all nodes of the current cluster and shows Cassandra instances running on local host

Various CCM Cluster Commands (2)

- The command
  ```
  ccm start
  ```
  starts all not started nodes of the current cluster
- The command
  ```
  ccm stop
  ```
  stops all nodes of the current cluster
- The command
  ```
  ccm remove [cluster_name]
  ```
  removes the current, or the specified cluster and deletes all data

CCM `add` Command

- Usage: `ccm add [options] node_name`
- Description: adds a new node to the current cluster
- Options (we care for):

`-s`	Configure this node as a seed
`-b`	Set auto bootstrap for the node
`-i ITFS`	Set the node IP address to `ITFS`
`-j JMX_PORT`	Set the jmx port for the node to `JMX_PORT`
`-n INITIAL_TOKEN`	Set the value of the initial token for `node_name` to `INITIAL_TOKEN`
`-d DATA_CENTER`	Data centre name this node is part of (**only for multi dc data centeres**)

- Example

```
ccm add -i 127.0.0.8 -j 7800 -b -d dc1  -n
-3074457345618258604 node8
```

CCM Node Commands (1)

- The command `ccm node_name show` displays the content of the file `~/cassandra_training/current_cluster_name/node_name/node.conf`
- The command `ccm node_name remove` removes the node `node_name` (stopping it if needed and deleting all its data)
- The command `ccm node_name start` starts the node `node_name` providing that it is a seed node, or a seed node is already active
- The command `ccm node_name stop` stops the node `node_name`

CCM Node Commands (2)

- The following ccm commands may be applied on active nodes only
- The command `ccm node_name ring` displays a detailed information about the ring connecting to the node `node_name`
- The command `ccm node_name status` displays a detailed information about the whole current cluster
- The command `ccm node_name cqlsh` launches a cqlsh session connected to the node `node_name` and displays `cqlsh >` prompt

cqlsh

- The Cassandra installation includes the `cqlsh` utility
  - A command line client for executing CQL commands interactively
  - `cqlsh` supports tab completion
- `cqlsh` also supports `cqlsh` commands, we consider:
  - CAPTURE
  - CONSISENCY
  - COPY
  - DESCRIBE
  - EXIT
  - SHOW
  - SOURCE
  - TRACING

`cqlsh` Commands (1)

- `CAPTURE`
  - Captures command output and appends it to a file
  - **Synopsis:** `CAPTURE ('<file>' | OFF )`
  - **Description:**
    - Query result output is captured and not shown on the console
    - Errors and output from `cqlsh`-only commands still appear
    - To stop capturing, use `CAPTURE OFF`
- `CONSISTENCY`
  - Shows the current consistency level, or given a level, sets it.
  - **Synopsis:** `CONSISTENCY level`
  - **Description:**
    - Providing an argument to the `CONSISTENCY` command sets the consistency level for future requests.
    - Providing no argument shows the current consistency level

`cqlsh` Commands (2)

- `COPY`
  - Imports and exports CSV (comma-separated values) data to and from Cassandra
  - **Synopsis:**

```
COPY table_name ( column, ...)
FROM ( 'file_name' | STDIN )

COPY table_name ( column , ... )
TO ( 'file_name' | STDOUT )
```

- `DESCRIBE`
  - Provides information about the connected Cassandra cluster, or about the data objects stored in the cluster.
  - **Synopsis:**

```
DESCRIBE ( CLUSTER | SCHEMA ) | KEYSPACES | (
KEYSPACE keyspace_name ) | TABLES | ( TABLE
table_name )
```

- `EXIT`

`cqlsh` Commands (3)

- `SOURCE`
  - Executes a file containing CQL statements
  - **Synopsis:** `SOURCE 'file'`
  - **Description:**
    - The output for each statement, if there is any, appears in turn, including any error messages
    - Errors do not abort execution of the file
- `TRACING`
  - Enables or disables request tracing
  - **Synopsis:** `TRACING ( ON | OFF )`
  - **Description :**
    - After turning on tracing, database activity creates output that can help you understand Cassandra internal operations and troubleshoot performance problems
    - To stop tracing execute `TRACING OFF`

Node Tool

- The `nodetool` utility is a command line interface for managing a cluster
- The `nodetool` commands are executed by issuing

```
~/cassandra_training/cluster_name/nodei/
bin/nodetool [-p 7i00] <cmd>
```

- The `nodetool` has a great number of commands
  - To get the list of all commands:

```
~/cassandra_training/cluster_name/nodei/bin/nodetool
  [-p 7i00] help
```

  - To get help on a specific command:

```
~/cassandra_training/cluster_name/nodei/bin/nodetool
  [-p 7i00] help <cmd>
```

`nodetool getendpoints` Command

- The command provides IP addresses of replica nodes that own a given partition key
- Synopsis:

```
~/cassandra_training/cluster_name/nodei/b
in/nodetool getendpoits <keyspace>
<table> partition key
```

- An alternate command format:

```
ccm nodei nodetool getendpoints <keyspace>
<table> partition key
```

- Description:
  - The partitioner returns a token for the `key`
  - Cassandra will return endpoints regardless whether or not data exist on the identified node for that token

`nodetool getendpoints` Example

```
cqlsh> create keyspace multi_dc
with replication =
{'class': 'NetworkTopologyStrategy',
'dc1':2, 'dc2':3};
cqlsh> use multi_dc ;
cqlsh:multi_dc> create table user
(username text primary key, email text);
```

```
embassy: [~] % ccm node1 nodetool getendpoints
multi_dc user pavle
127.0.0.8
127.0.0.1
127.0.0.4
127.0.0.5
127.0.0.6
```

`nodetool rebuild` Command

- `rebuild`
  - Rebuilds data by streaming from other nodes
- Synopsis:

```
nodetool rebuild [--] <data center>
```

  - `data_center` is the name of the data centre from which to select source data for streaming
- Description:
  - This command operates on multiple nodes in a cluster
  - Rebuild only streams data from a single source replica per range
  - This command is used to bring up a new data centre in an existing cluster
  - For example, when adding a new data center, you would run the following on all nodes in the new data center:

```
nodetool rebuild -- name_of_existing_data_center
```

Summary

- Cassandra Cluster Manager (`ccm`) is a tool containing scripts and libraries to create, launch, manipulate, and remove Cassandra clusters on local host
  - Cluster commands
  - Node commands
- Files created by `ccm` reside in:

```
      ~/cassandra-training/<cluster_name>
~/cassandra-training/<cluster_name>/<node_name>
```

- `cqlsh` commands complement CQL commands
- `nodetool` utility is a command line interface for managing a cluster