Introduction to Cassandra for Developers (3 Days)

Course Description:

The Cassandra (C*) database is a massively scalable NoSQL database that provides high availability and fault tolerance, as well as linear scalability when adding new nodes to a cluster. It has many powerful capabilities, such as tunable and eventual consistency, that allow it to meet the needs of modern applications, but also introduce a new paradigm for data modeling that many organizations do not have the expertise to use in the best way
This course provides an in-depth introduction to using Cassandra and creating good data models with Cassandra. It is technical and comprehensive, with a focus on the practical aspects of working with C*. It introduces all the important concepts needed to understand Cassandra, including enough coverage of internal architecture to make good decisions. It is hands-on, with labs that provide experience in all the important areas. It covers CQL (Cassandra Query Language) in depth, as well as covering the Java API for writing Cassandra clients.

After taking this course, you will have learned what you need to productively work with Cassandra as well as guidelines for using it in an optimal manner. You'll also understand some of the "anti-patterns" that lead to non-optimal C* data models. You'll be familiar with CQL and with the Java client library, and be ready to work on production systems involving Cassandra.

This course includes coverage of Cassandra 3.x, but is also suitable for users of Cassandra 2. Features introduced in Cassandra 3 are clearly indicated.

Class Prerequisites

Experience in the following *is required* for this Apache Cassandra class:

- Reasonable Java experience for the Java driver labs, some knowledge of databases.

Goals

- Be familiar with the operation and structure of C*
- Be able to install and set up a C* database
- Use the C* tools, including cqlsh, nodetool, and ccm (Cassandra Cluster Manager)
- Be familiar with the C* architecture, and how a C* cluster is structured
- Understand how data is distributed and replicated in a C* cluster
- Understand core C* data modeling concepts, and use them to create well-structured data models
- Use data replication and eventual consistency intelligently
- Understand and use CQL to create tables and query for data
- Know and use the CQL data types (numerical, textual, uuid, etc.)
- Understand the various kinds of primary keys available (simple, compound, and composite primary keys)
- Use more advanced capabilities like collections, counters, secondary indexes, CAS (Compare and Set), static columns, and batches
- Be familiar with the Java client API
- Use the Java client API to write client programs that work with C*
- Build and use dynamic queries with QueryBuilder
- Understand and use asynchronous queries with the Java API

Outline

1. **Cassandra Overview**
    - A. Why We Need Cassandra
    - B. High level Cassandra Overview
    - C. Cassandra Features
    - D. Basic Cassandra Installation and Configuration

2. **Cassandra Architecture and CQL Overview**

   . Cassandra Architecture Overview
    - A. Cassandra Clusters and Rings
    - B. Data Replication in Cassandra
    - C. Cassandra Consistency / Eventual Consistency
    - D. Introduction to CQL
    - E. Defining Tables with a Single Primary Key
    - F. Using cqlsh for Interactive Querying
    - G. Selecting and Inserting/Upserting Data with CQL
    - H. Data Replication and Distribution
    - I. Basic Data Types (including uuid, timeuuid)

3. **Data Modeling and CQL Core Concepts**

   . Defining a Compound Primary Key
    - A. CQL for Compound Primary Keys
    - B. Partition Keys and Data Distribution
    - C. Clustering Columns
    - D. Overview of Internal Data Organization
    - E. Additional Querying Capabilities
    - F. Result Ordering - ORDER BY and CLUSTERING ORDER BY
    - G. UPDATE and DELETE Queries
    - H. Result Filtering, ALLOW FILTERING
    - I. Batch Queries
    - J. Data Modeling Guidelines
    - K. Denormalization
    - L. Data Modeling Workflow
    - M. Data Modeling Principles
    - N. Primary Key Considerations
    - O. Composite Partition Keys
    - P. Defining with CQL
    - Q. Data Distribution with Composite Partition Key
    - R. Overview of Internal Data Organization

4. **Additional CQL Capabilities**

   . Indexing
    - A. Primary/Partition Keys and Pagination with token()
    - B. Secondary Indexes and Usage Guidelines
    - C. Cassandra Counters
    - D. Counter Structure and Definition

E. Using Counters
F. Counter Limitations
G. Cassandra collections
H. Collection Structure and Uses
I. Defining Collections (set, list, and map)
J. Querying Collections (Including Insert, Update, Delete)
K. Limitations
L. Overview of Internal Storage Organization
M. Static Column: Overview and Usage
N. Static Column Guidelines
O. Materialized View: Overview and Usage
P. Materialized View Guidelines

5. Data Consistency In Cassandra

. Overview of Consistency in Cassandra

A. CAP Theorem
B. Eventual (Tunable) Consistency in C* - ONE, QUORUM, ALL
C. Choosing CL ONE
D. Choosing CL QUORUM
E. Achieving Immediate Consistency
F. Using other Consistency Levels
G. Internal Repair Mechanisms (Read Repair, Hinted Handoff)
H. Lightweight Transactions (LWT)/ Compare and Set (CAS)
I. Overview of Lightweight Transactions
J. Using LWT, the [applied] Column
K. IF EXISTS, IF NOT EXISTS, Other IF conditions
L. Basic CAS Internals
M. Overhead and Guidelines

6. Practical Considerations

. Dealing with Write Failure

A. Unavailable Nodes and Node Failure
B. Requirements for Write Operations
C. Key and Row Caches
D. Cache Overview
E. Usage Guidelines
F. Multi-Data Center Support
G. Overview
H. Replication Factor Configuration
I. Additional Consistency Levels - LOCAL/EACH QUORUM
J. Deletes
K. CQL for Deletion
L. Tombstones
M. Usage Guidelines

7. The Java Client API

. API Overview

- A. Introduction
- B. Architecture and Features
- C. Connecting to a Cluster
- D. Cluster and Cluster.Builder
- E. Contact Points, Connecting to a Cluster
- F. Session Overview and API
- G. Working with Sessions
- H. The Query API
- I. Overview
- J. Dynamic Queries, Statement, SimpleStatement
- K. Processing Query Results, ResultSet, Row
- L. PreparedStatement, BoundStatement
- M. Binding Values and Querying with PreparedStatements
- N. CQL to Java Type Mapping
- O. Working with UUIDs
- P. Working with Time/Date Values
- Q. Working with Batches of SimpleStatement and PreparedStatement
- R. Dynamic Queries and QueryBuilder
- S. QueryBuilder Overview and API
- T. Building SELECT, DELETE, INSERT, and UPDATE Queries
- U. Creating WHERE Clauses
- V. Other Query Examples
- W. Configuring Query Behavior
- X. Setting LIMIT and TTL
- Y. Working with Consistency
- Z. Using LWT
- AA. Working with Driver Policies
- BB. Load Balancing Policies - RoundRobinPolicy, DCAwareRoundRobinPolicy
- CC. Retry Policies - DefaultRetryPolicy, DowngradingConsistencyRetryPolicy, Other Policies
- DD. Reconnection Policies
- EE. Asynchronous Querying Overview
- FF. Synchronous vs. Asynchronous Querying
- GG. Executing Asynchronous Queries
- HH. java.util.concurrent.Future
- II. Cassandra ResultSetFuture
- JJ. Future Result Processing