Graph Databases using OrientDB over Tinkerpop3 – Course Outline

# Duration

4 days

# Objectives

After this course the participants will be able to

1. Understand when to use graph database
2. Design and build graph models for their business entities and relationships
3. Set up and configure OrientDB graph database
4. Perform all graph operations and algorithms on OrientDB
5. Understand Tinkerpop3 API and integrate with OrientDB
6. Overview of Apache Spark
7. Graph parallel computation using GraphX – component in Spark
8. Case study on application of graph database in Social networking

# Audience

This course is for application developers and architects who would like to understand and implement graph database in their application

# Pre-requisite

1. Good programming skills in Java 1.8+
2. Used maven to setup and build project

# Outline of Topics covered

## Day 1 – Session 1

1. Overview of NoSQL databases – typical characteristics, when to use them, when not to use them
   1. Key – Value database
   2. Document database
   3. Graph database
2. Overview of Graph databases
   1. Graph data structures
   2. Graph data modelling
   3. Graph data processing and algorithms
3. Overview of OrientDB
   1. Install and configure **(Hands-on)**
   2. OrientDB Studio – overview **(Hands-on)**
   3. OrientDB Console – Overview **(Hands-on)**

## Day 1 – Session 2

1. Perform CRUD using OrientDB
   1. Build a basic Graph model in OrientDB **(Hands-on)**
   2. Orient DB – SQL API overview
   3. Implement CRUD operations **(Hands-on)**
   4. Graph functionality and traversals **(Hands-on in parallel)**
2. Fetching Strategies
   1. Fetch strategy on query and on record loading
   2. Managing circular dependencies
   3. Execute a query with custom fetch plan **(Hands-on)**
3. Overview of Tinkerpop3
   1. Objective of Tinkerpop3
   2. Tinkerpop3 subprojects and their roles
   3. Setting up and configuring Gremlin **(Hands-on)**
   4. Create graph with vertices and edges and perform traversal using TinkerGraph as the graph instance **(Hands-on)**

## Day 2 – Session 1

1. Overview of case study (modelling a collaboration graph in the context of an enterprise (projects, employees, departments, roles, etc) or in an educational institution)
   1. Define problem statement
   2. Come up with relational model of entities and their relations **(Hands-on)**
   3. Come up with Graph model for the same **(Hands-on)**
   4. Compare and contrast relational model with graph model

***Note: All following Hands-on will be dealt in the context of implementing the case study mentioned in this session***

1. Overview of Tinkerpop3 Gremlin Graph Structure API
   1. Create Graph with vertices and edges using in-memory TinkerGraph and OrientDB **(Hands-on)**
   2. Show advantages of using Tinkerpop by switching between TinkerGraph and OrientDB **(Hands-on in parallel)**
   3. Crate Property Graph with properties assigned to both edges and vertices **(Hands-on in parallel)**
   4. Updating graph **(Hands-on)**
   5. Transaction support through TransactonalGraph support
   6. Overview of Graph features and determining what features are supported in each graph database
2. Graph Process API for traversal
   1. Basic graph traversals **(Hands-on in parallel)**

## Day 2 – Session 2

1. Gremlin Graph in detail
   1. Vertex properties
      1. Multi, meta properties
      2. Permissions, Auditing, Provenance
   2. Graph Variables
      1. Schema information
      2. Global permissions
      3. System user information
   3. Define Graph with relevant information for the case study **(Hands-on)**
   4. Graph transactions
      1. Configuring transaction
      2. Retries during failure
      3. Threaded transactions
      4. Defining transanctional services for case study **(Hands-on)**

## Day 3 – Session 1

1. Reading, Writing using Gremlin I/O
   1. GraphReder and GraphWriter APIs to do I/O on every part of Graph **(Hands-on in parallel)**
2. Exporting and importing using Gremlin I/O
   1. GraphML Reader / Writer **(Hands-on in parallel)**
   2. GraphSON Reader / Writer **(Hands-on in parallel)**
   3. Binary Gryo Reader / Writer **(Hands-on in parallel)**
3. Namespace conventions
4. Graph Traversal Steps
   1. Overview of list of Steps
   2. Predicates
   3. Barrier Steps
   4. Lambdas
   5. Traversal in the context of case study **(Hands-on)**

## Day 3 – Session 2

1. Traversal Strategy
   1. Overview of list of strategies
   2. Registering listeners with EventStrategy
2. Gremlin Traversal patterns (some from the following list)
   1. Backtrack Pattern
   2. Except/Retain Pattern
   3. Flow Rank Pattern
   4. Path Pattern
   5. Loop Pattern
   6. Split/Merge Pattern
   7. MapReduce Pattern
   8. Pattern Match Pattern
   9. Tree Pattern
   10. Applying pattern in case study **(Hands-on)**

## Day 4 – Session 1

1. Fine tuning
   1. Step Closures and User defined steps
   2. Depth first vs Breadth first **(Hands-on)**
   3. Traversal Optimization **(Hands-on\_**
2. Indexes
   1. Manual and Automatic index
   2. Index types – SB-tree, Hash, Full text, Lucene full text, Lucene spatial index
3. Security
   1. Database security
   2. Server Security
   3. Database encryption
   4. Secure SSL connections
   5. OrientDB Web server
4. Gremlin Server
   1. Why Rexster?
   2. Rexster configuration
   3. Basic REST API **(Hands-on in parallel)**
   4. Monitoring
   5. Mapping a URI to JSON
   6. Rexster MIME types
   7. Using RexsterGraph **(Hands-on)**
   8. Writing and Managing Extensions

## Day 4 – Session 2

1. Apache Spark **(Hands-on in parallel)**
   1. Overview
   2. Linking with Spark
   3. Initializing Spark with Shell
   4. Resilient Distributed Datasets (RDDs)
      1. Parallelized Collections
      2. External Datasets
      3. RDD Operations
      4. RDD Persistence
   5. Shared Variables
   6. Deploying to cluster
   7. Launching Spark jobs from Java
2. GraphX Component in Spark **(Hands-on in parallel)**
   1. Property Graph
   2. Overview of Graph Operators
   3. Caching and uncachin
   4. Pregel API
   5. Graph Builders
   6. Vertex and Edge RDDs
   7. Graph Algorithms

# Hardware & Network Requirements

1. All participants to have individual desktops with at least i3 2nd gen with 4GB RAM
2. Access to internet

# Software Requirements

1. Windows XP/7/8 Operating System
2. Java 1.8
3. Eclipse IDE (preferably Luna or higher)