AMBIENT-IT

+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 02/07/2025

Sign up

FoundationDB Training

4 days (28 hours)

Overview

Discover the FoundationDB training course, which will enable you to master the principles of a distributed transactional database and deploy robust clusters in production.

You'll store data, manage multi-key transactions and structure your data using the Tuple and Record layers.

You'll know how to design a scalable transactional storage system, implement a structured catalog with indexes, manage access concurrency, and deploy a resilient FoundationDB cluster.

You'll be able to exploit FoundationDB in a data pipeline or microservices backend, to model metadata, documents or relational objects, within a unified, consistent, high-performance architecture.

As with all our training courses, it will be run on my latest version of the FoundationDB .tool

Objectives

- Understand the operation and architecture of FoundationDB
- Develop client applications (Python, Java) with ACID transactions
- Structure data using the Tuple Layer and Record Layer
- Deploy, monitor and test a fault-tolerant FoundationDB cluster
- Apply FoundationDB to real-life use cases

Target audience

- Data engineers

- Technical architects
- Back-end developers

Prerequisites

- Knowledge of database management
- Basic command of a programming language
- Notions of transactions, data structures and distributed systems
- Familiarity with shell or command-line environments

FoundationDB training program

Introduction to FoundationDB and its positioning

- What is FoundationDB? History and use cases
- NoSQL, NewSQL, ordered key-value: technical typology
- FoundationDB's role in the Apple ecosystem, Snowflake, etc.
- Comparison with other databases: Cassandra, MongoDB, etc.
- Overview of the distributed ACID transaction model
- Multi-model vision and layered architecture

Understanding distributed architecture

- System roles: coordinator, proxies, master, resolvers
- Key space fragmentation and dynamic distribution
- Transaction logging, replication and high availability
- Disk storage: B-Tree engine and Redwood engine
- Monitoring, failover and restart without data loss
- Simultaneous read/write cluster operation

Data structure and key-value space

- Key naming convention (subspaces)
- Binary encoding, typed serialization (Tuple Layer)
- Range queries and lexicographic ordering
- Key/value design practices for performance
- Implicit schemas and application versioning
- Workshop: Manual exploration with fdbcli, range queries, reading/writing typed keys

Distributed transactions and ACID properties

- Strong isolation: serializability, read/write conflicts
- Atomicity between keys and between operations
- Transaction flow (begin ? commit ? retry)

- Conflicts and error handling (conflict range, retries)
- Consistency tests on several concurrent clients
- Workshop: Writing a transactional Python application with conflict management

Client programming with the FoundationDB API

- Client installation (Python, Java or other language)
- Main functions: get, set, clear, range, atomic_ops
- Context-sensitive transactions and automatic closing
- Integration in pipelines or microservices
- Error handling, timeouts, secure withdrawals
- Structured key encoding with Tuple()

Structuring data with Tuple Layer

- Purpose of Tuple Layer: typed and ordered serialization
- Native type to binary key conversion
- Field names, hierarchical key structuring
- Best practices in key-value schema design
- Encapsulation of application logic in keys
- Partial reading or deep browsing

Record Layer: modeling structured data

- Introduction to Apple's Record Layer (CloudKit)
- Defining record schemas
- Typed fields, secondary indexes, pagination
- Comparison with relational databases (SQL without JOIN)
- Deploying a record layer on the cluster
- Workshop: Creating a "user table" model with secondary index in the Record Layer

Advanced indexing and range queries

- Primary and secondary indexes
- Compound, sorted and multi-key indexes
- Graph traversal via indexed relations
- Efficient scanning of sorted ranges (prefix matching)
- Optimization of reads via index schemes

Designing a multi-model application

- Combining documentary and relational models
- Management of critical metadata (catalogs, histories)
- Inter-model synchronization and global consistency
- Centralize multiple types of data in a single database
- Use cases: recommendation engine or event history

Deploying a FoundationDB cluster

- Local, Docker or Kubernetes installation
- Role configuration, replication, replica sets
- Startup, shutdown, horizontal scaling
- Storage strategies (SSD, HDD, Redwood vs. SQLite)
- Workshop: Deploying a multi-node mini-cluster with failure simulation and rebalancing

Monitoring, testing and reliability

- Logs, metrics, system status
- Load testing, fault injection (simulation framework)
- Resilience: fault tolerance, automatic election
- Alerting and critical thresholds (latency, space, quorum)
- Best practices for continuous operation

Real-life use cases and architecture choices

- Case studies : Snowflake, iCloud, large catalogs
- Critical metadata storage and transactional systems
- When to choose FoundationDB over Cassandra, PostgreSQL or etcd?
- Integration into native cloud architectures (CI/CD, microservices)
- Known limits, possible extensions, open source roadmap

Companies concerned

This course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced IT technology, or to acquire specific business knowledge or modern methods.

Positioning on entry to training

Positioning at the start of training complies with Qualiopi quality criteria. As soon as registration is finalized, the learner receives a self-assessment questionnaire which enables us to assess his or her estimated level of proficiency in different types of technology, as well as his or her expectations and personal objectives for the forthcoming course, within the limits imposed by the selected format. This questionnaire also enables us to anticipate any connection or security difficulties within the company (intra-company or virtual classroom) which could be problematic for the follow-up and smooth running of the training session.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire verifies the correct acquisition of skills.

Certification

A certificate will be awarded to each trainee who has completed the entire course.