



NoSQL Databases Training: MongoDB Data Modelling, Redis Caching and Cassandra



NoSQL Databases Training: MongoDB Data Modelling, Redis Caching and Cassandra

Introduction:

NoSQL Databases Training on MongoDB data modelling, Redis caching and Cassandra is a 5-day course for application developers, data engineers and database administration staff, ending with a Document Data Model and Query Pack. Teams often move data into document or key-value stores without modelling for access patterns, so collections bloat, queries scan whole datasets, caches serve stale records and clusters develop write hotspots. Nominees already build or operate application data stores, and every day is taught as a hands-on lab on running database instances. CoreConcept Training Center delivers this NoSQL databases course.

Course Objectives:

- Assess application workloads against document, key-value, wide-column and graph store families and their CAP trade-offs
- Design MongoDB document schemas using embedding, referencing and named schema design patterns
- Write CRUD operations and aggregation pipelines and support them with indexes verified through explain plans
- Configure Redis as a cache layer with suitable data structures, TTL expiry and an eviction policy
- Model Cassandra tables from queries and plan replica sets, shard keys, access control and backup for production use
- Produce a Document Data Model and Query Pack for a case application

Target Audience:

- Application development staff who build services on document or key-value data stores
 - Data engineering staff who load, reshape and serve data from NoSQL platforms
 - Database administration staff taking on MongoDB, Redis or Cassandra alongside relational engines
 - Platform and DevOps staff who provision, scale and monitor distributed data clusters
 - Solution design staff who choose the data store for new applications
-

Course Outline:

Day 1: NoSQL Families, CAP Trade-Offs and Workload Fit

- Four NoSQL Families Compared Document, Key-Value, Wide-Column, Graph
- CAP Theorem Trade-Offs Between Consistency, Availability and Partition Tolerance
- Eventual Consistency Versus Multi-Document ACID Transactions in MongoDB
- NoSQL Versus Relational Selection Matrix for Application Workloads
- Access Pattern Inventory of a Current Application Data Estate

Day 2: MongoDB Document Modelling and Schema Design Patterns

- Embedding Versus Referencing Decision Rules for Related Documents
- Bucket Pattern Grouping Time-Series Data and Handling Outliers
- Polymorphic Pattern and Single Collection Pattern for Mixed Entities
- Computed Values Pattern and Archive Pattern for Hot Data
- JSON Schema Validation and Document Versioning for Evolving Collections

Day 3: MongoDB Querying, Indexing and Redis Key-Value Caching

- MongoDB CRUD Operators, Projections and Bulk Write Operations
- Aggregation Pipeline Stages Match, Group, Lookup and Unwind
- Compound, Multikey and TTL Index Design Checked With Explain Plans
- Redis Data Structures Strings, Hashes, Sorted Sets and Streams
- Redis Cache-Aside Pattern With TTL Expiry and Eviction Policies

Day 4: Replication, Sharding, Wide-Column Modelling and Operations

- MongoDB Replica Sets, Elections, Write Concern and Read Preference
- Shard Key Selection by Cardinality, Frequency and Monotonic Change
- Cassandra Query-First Tables With Partition and Clustering Keys
- Access Control, TLS Encryption and Backup Restore for NoSQL Stores
- Monitoring Hit Ratios, Slow Queries and Replication Lag Metrics

Day 5: Lab Building a Document Data Model With Queries and Indexes

- Case Application Brief and Access Pattern Workload Sizing
 - Lab Applying Schema Design Patterns to the Case Collections
 - Lab Writing Aggregation Pipelines and Index Set for Case Queries
 - Lab Adding a Redis Cache Layer and Shard Key Proposal
 - Document Data Model and Query Pack Completion and Peer Review
-

Skills You Will Gain:

- Access Pattern Analysis
- Document Schema Design
- Aggregation Pipeline Development
- NoSQL Index Tuning
- Cache Layer Design
- Shard Key Evaluation
- Wide-Column Table Modelling
- Cluster Health Monitoring

Why Attend This Course:

- Deliver a Document Data Model and Query Pack to the application lead or data platform owner for design sign-off
- Choose between embedding and referencing, or between MongoDB, Redis and Cassandra, with evidence drawn from access patterns
- Avoid collection scans, stale cache reads and shard hotspots before they reach production users
- Share schema pattern checklists, index review steps and cache configuration templates with development colleagues

Conclusion:

Back at work, the participant hands the application lead or data platform owner a Document Data Model and Query Pack that records access patterns, chosen schema design patterns, aggregation pipelines, the index set, a caching plan and a shard key proposal. The team uses it to decide which data belongs in a document, key-value or wide-column store before build work starts. After the first production release, the team should review slow query logs, cache hit ratio and shard balance against the assumptions recorded in the pack.

Frequently Asked Questions FAQ:

What should participants know before the NoSQL databases training?

Participants should already write application code or database queries and read JSON documents. No prior MongoDB, Redis or Cassandra experience is needed. Familiarity with a command line and basic SQL helps participants move quickly through the daily labs.

How does the NoSQL databases training differ from a relational database design course?

It models data from application access patterns, embedding or duplicating data where queries need it, and scales through replication and sharding. A relational design course centres on entity-relationship models, normalisation and table constraints, which this course touches only for comparison.

When should a team choose NoSQL databases instead of a relational database?

Choose a NoSQL store when data shapes vary, access patterns are known, and horizontal scale or low-latency reads matter more than ad hoc joins. Strong cross-entity integrity and complex reporting usually still favour a relational database, and many systems use both.

What does a participant take back from the NoSQL databases training?

Each participant takes back a Document Data Model and Query Pack for a case application, with access patterns, schema design pattern choices, aggregation pipelines, an index set checked with explain plans, a Redis caching plan and a shard key proposal.