Overview of Neo4j Administration

Table of Contents

About this module

As a Neo4j database administrator, you will be responsible for ensuring that the deployed Neo4j application runs to meet the needs of its users. This includes ensuring that the Neo4j software is up-to-date and configured properly, monitoring the use of Neo4j, performing life cycle activities such as backups, and managing multiple Neo4j instances.

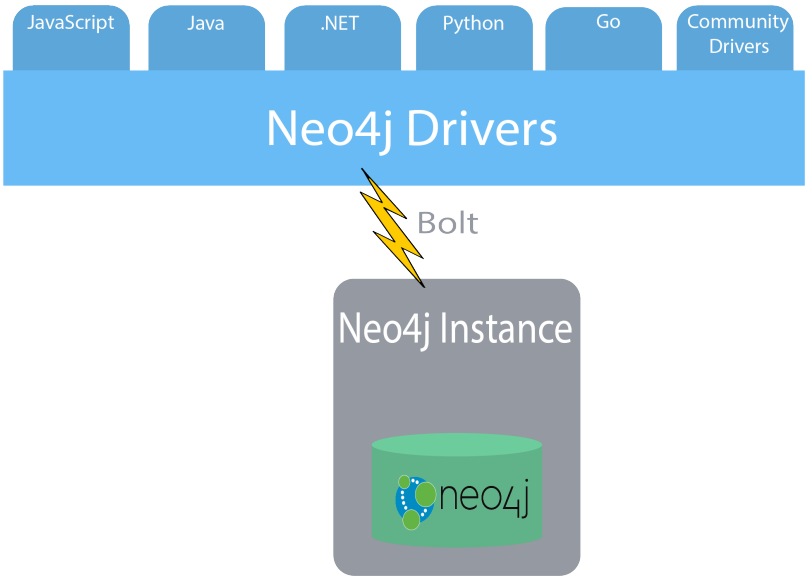
At the end of this module, you should be able to:

- Describe common application architectures that use Neo4j.
- Determine which edition of Neo4j to use.
- Download a specific Neo4j version.
- Determine which deployment option to use.
- Describe the administrative tasks for Neo4j.
- Install Neo4j Enterprise Edition.

Common application architectures with Neo4j

Neo4j can be used in a range of different application architectures. It is a transactional, real-time store for highly connected data.

Applications connect to Neo4j using the binary **Bolt** protocol, which is supported by the official language drivers. Currently the drivers for .NET, Java, JavaScript, Python, and Go are officially supported by Neo4j. There are also a set of other language drivers written by the Neo4j community.



Some applications use Neo4j out-of-the-box, and others extend it with additional functionality from libraries that provide specialized procedures. Examples of these libraries include:

- Neo4j-supported procedures:
  - Graph Algorithms
  - GraphQL
- Community-supported procedures, for example the Awesome Procedures of Cypher (APOC)
- Custom-developed procedures for your application

Regardless of which library your application requires, you must ensure that Neo4j is configured to know how to access these libraries.

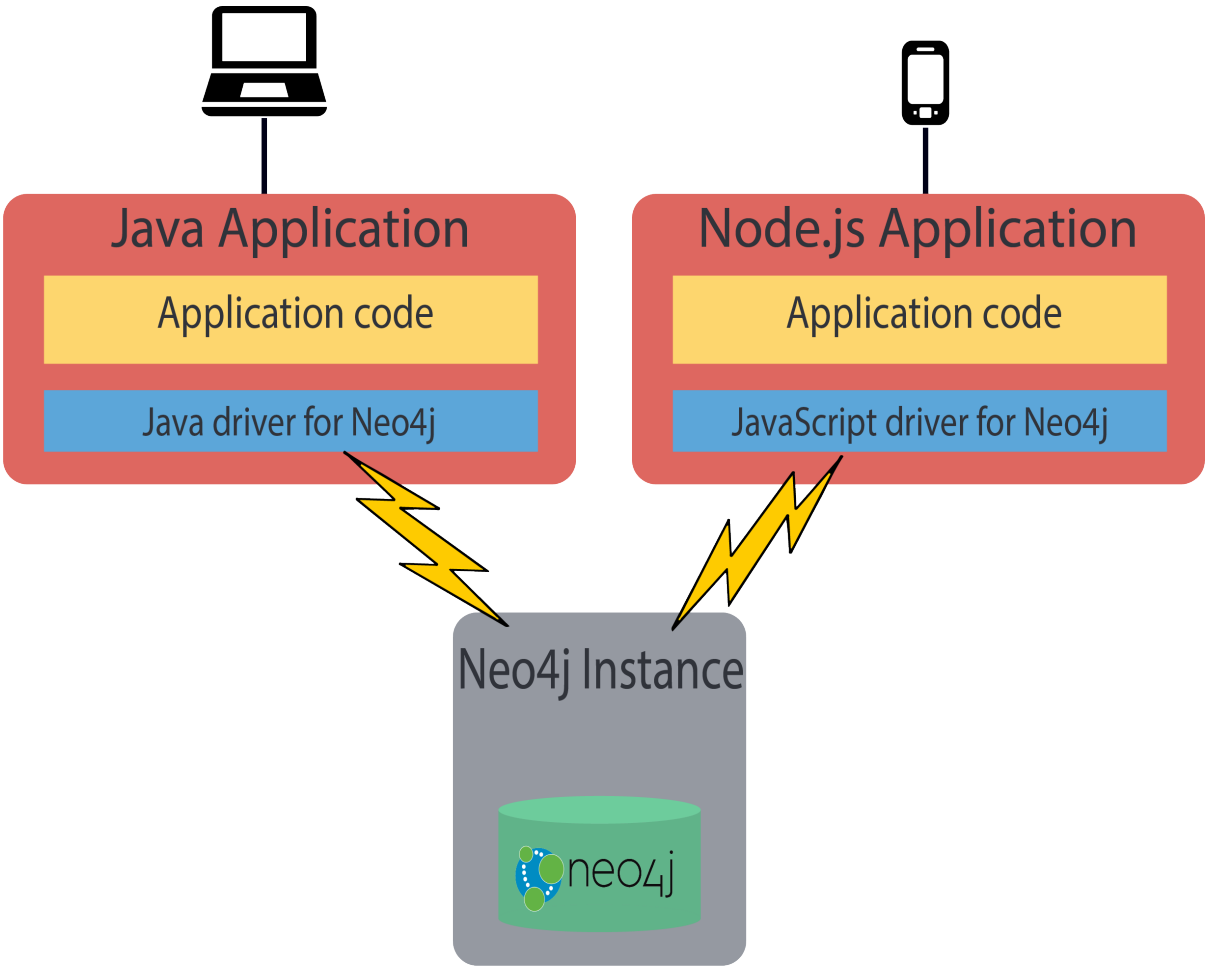
Your application will include at least one Neo4j database, but it could include multiple databases, each with their own Neo4j instance. A Neo4j instance can only support one active database.

Many deployed applications use Causal Clustering which you will learn about later in this training. Causal Clustering is an architecture which provides data safety and performance for critical systems that could be distributed all over the world.

Example architecture #1: Using Neo4j as the primary database

Some applications are developed using data that is stored in a single graph. They can be developed using any language for which there is a driver that can connect to Neo4j.

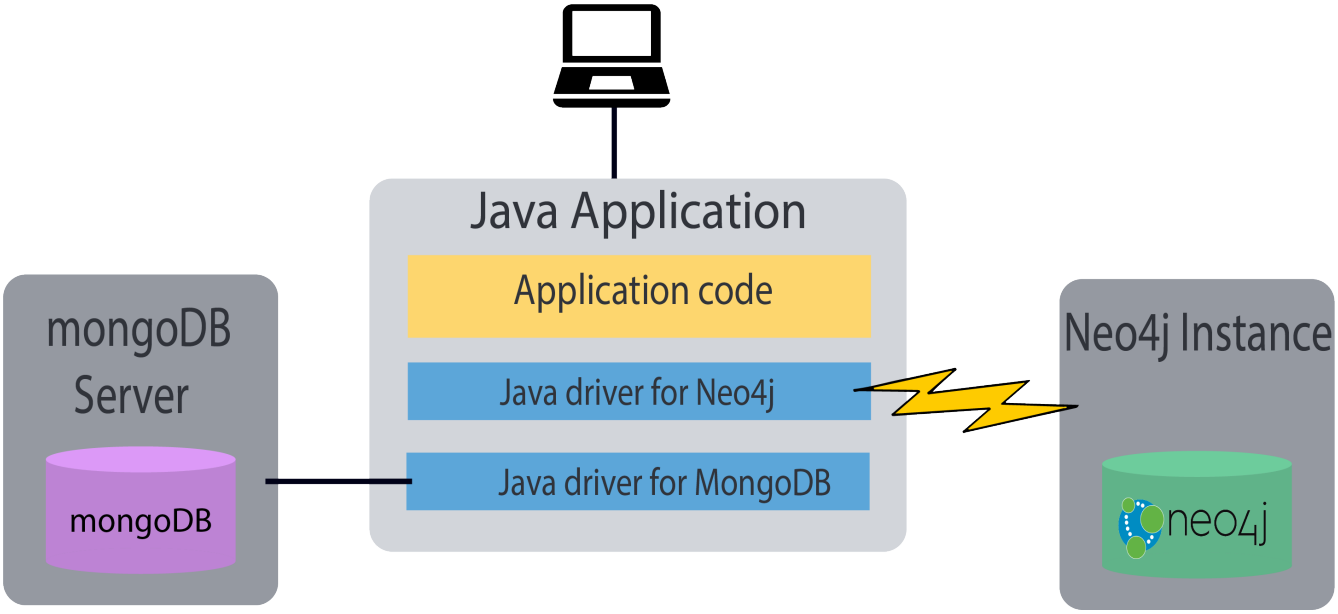
Each application (Neo4j client) communicates with Neo4j to access the graph using the Bolt or HTTP protocol. Neo4j-supported and some community drivers use the Bolt protocol. The clients can be JVMs, .NET resources, a Web server, all of which can be accessed by an end-user.



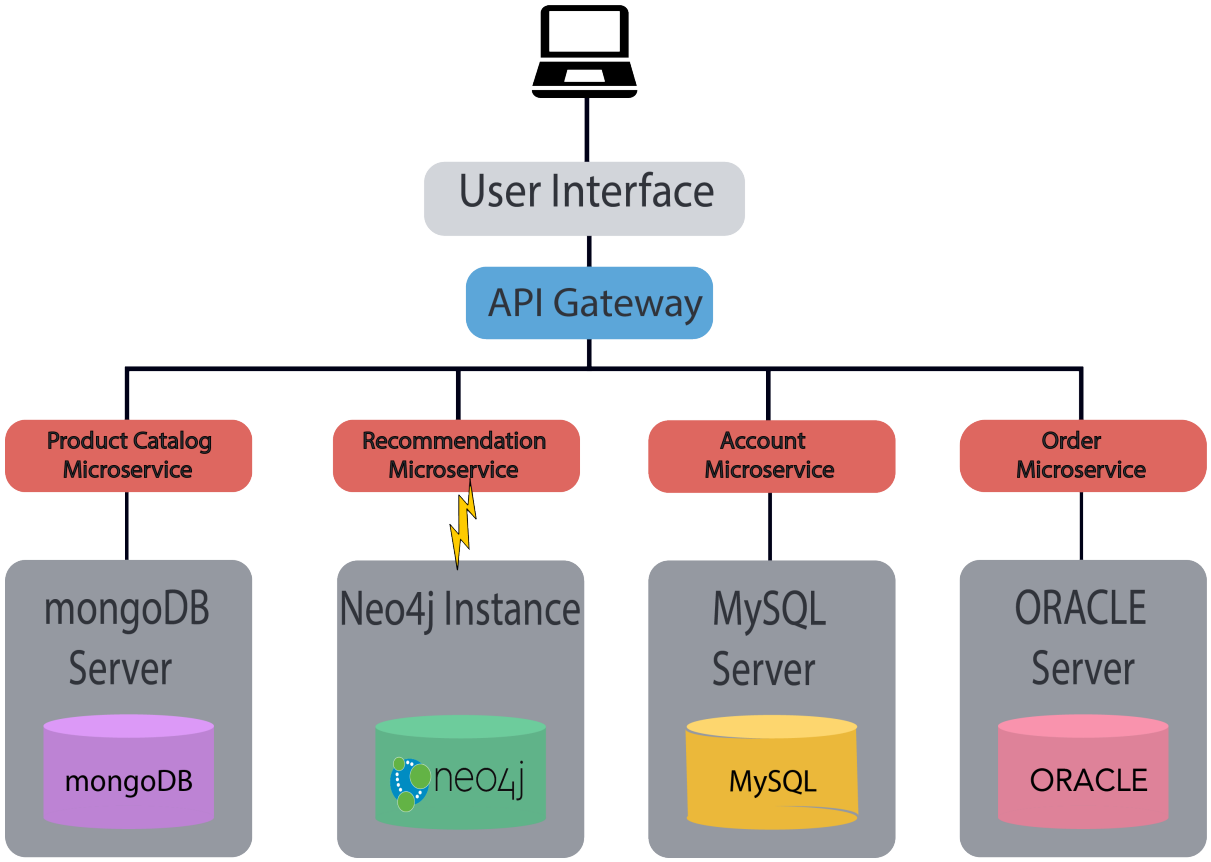
As a Neo4j database administrator, you are responsible for the availability and consistency of the Neo4j database.

Example architecture #2: Integrating Neo4j with other databases

Neo4j is often used in combination with other data sources (polyglot persistence), such as relational or NoSQL databases. Data can be fetched from the different systems in order to be manipulated by an application and subsequently written back to Neo4j. Other solutions can involve loading of data from a number of databases into Neo4j in order to efficiently do connected data analysis on the combined data set.



With cloud-hosted microservices, the implementation chosen for the microservice is one that performs best for the type of transaction. In a cloud-based microservice architecture, there may be a number of services that Neo4j can provide to the end-users.

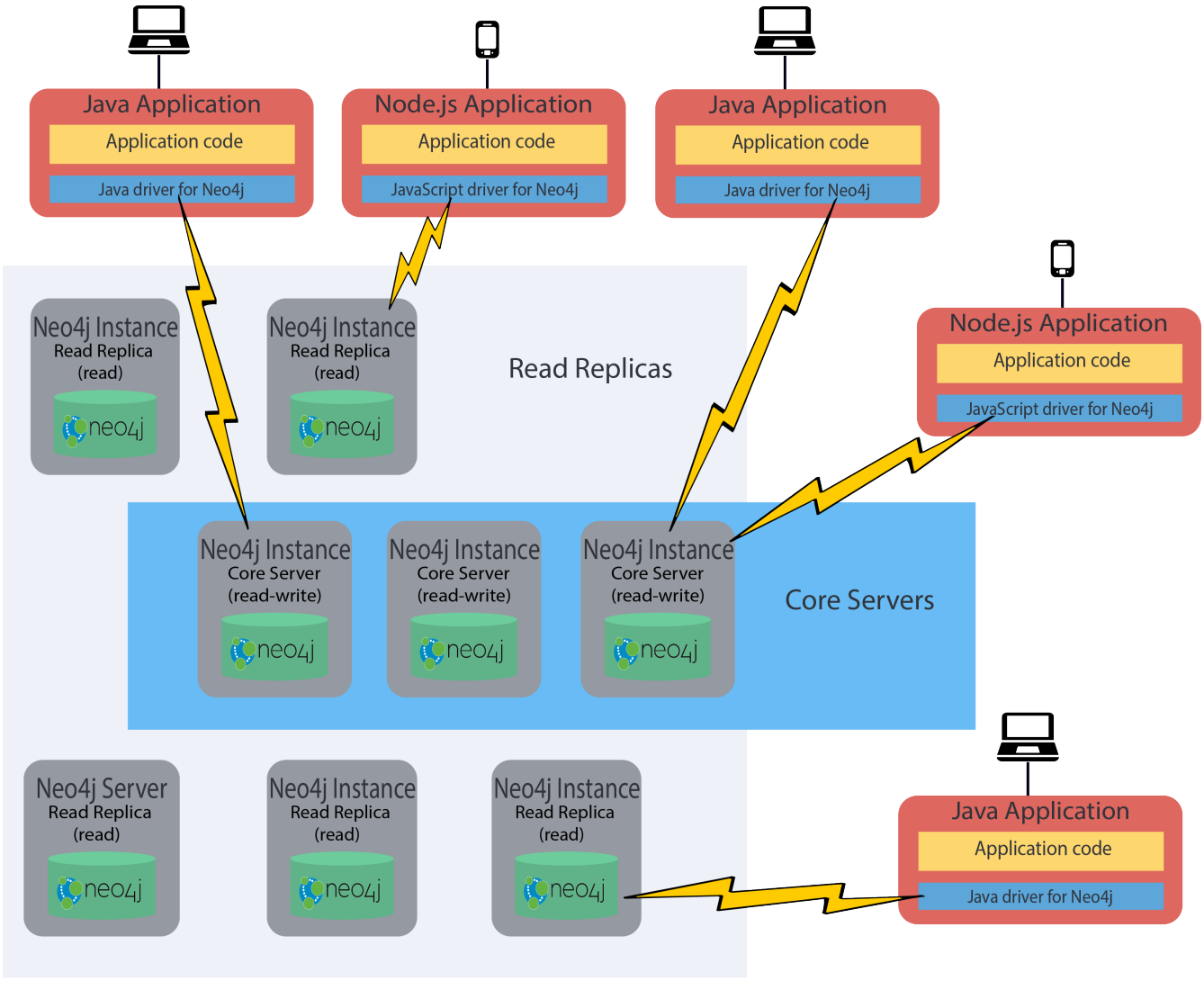


As a Neo4j database administrator, you are responsible for the availability and consistency of the Neo4j database, and for making the data accessible by applications. Application developers are responsible for the logic in transferring and manipulating data between the databases.

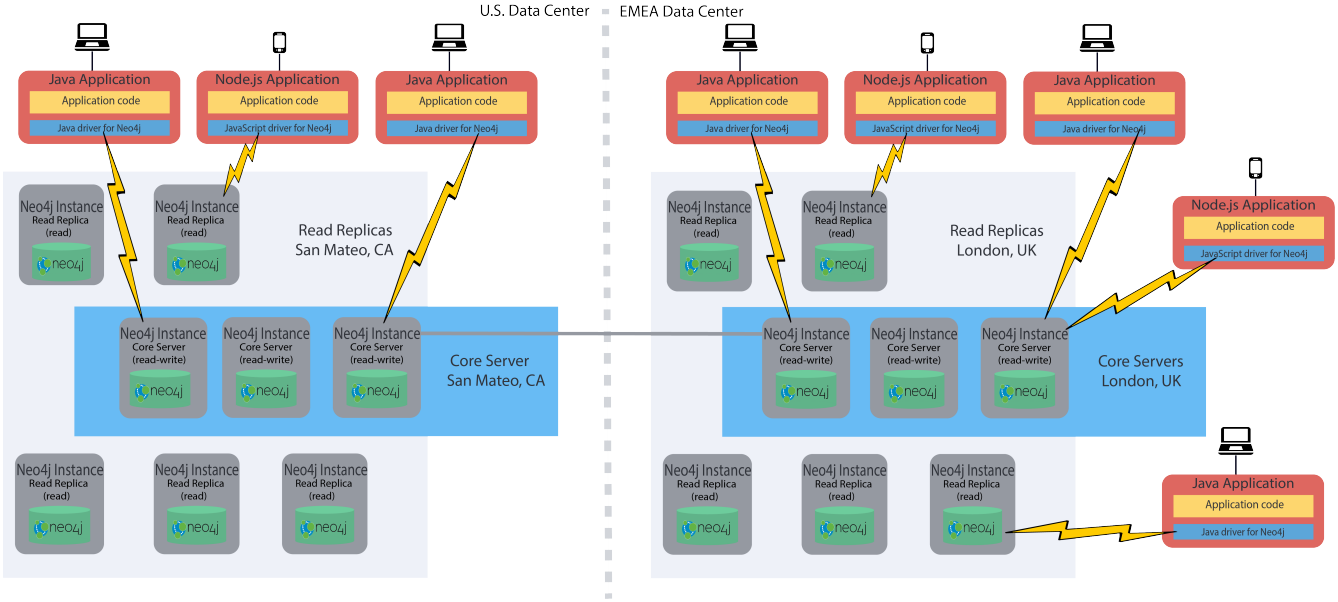
Example architecture #3: Using Neo4j Causal Clustering

Causal Clustering is used in production environments where high throughput, continuous availability, and reliability are important factors. This feature is available with Neo4j Enterprise Edition.

With Causal Clusters, you configure multiple Neo4j instances that communicate with each other about updates to the database. Causal Clusters are used when data needs to reside in multiple physical locations, or if you want to implement a high availability architecture where access to data will not be affected if a Neo4j instance goes down.



A common use for Neo4j is when different data is required in a set of geographic locations. For example, an online retailer may have different data in the US and Europe. Some data could be shared between different geographically located data centers, but the most heavily updated data needs to physically reside closer to the end-user applications. Neo4j Causal Clusters can be configured to span geographic locations.



As a Neo4j database administrator, you will be responsible for configuring and monitoring Causal Clusters. You will work with architects to determine the appropriate configuration of the Causal Cluster, taking into account aspects such as: uptime requirements, performance requirements, and redundancy required in order to handle events of Neo4j instance failure or data center failure.

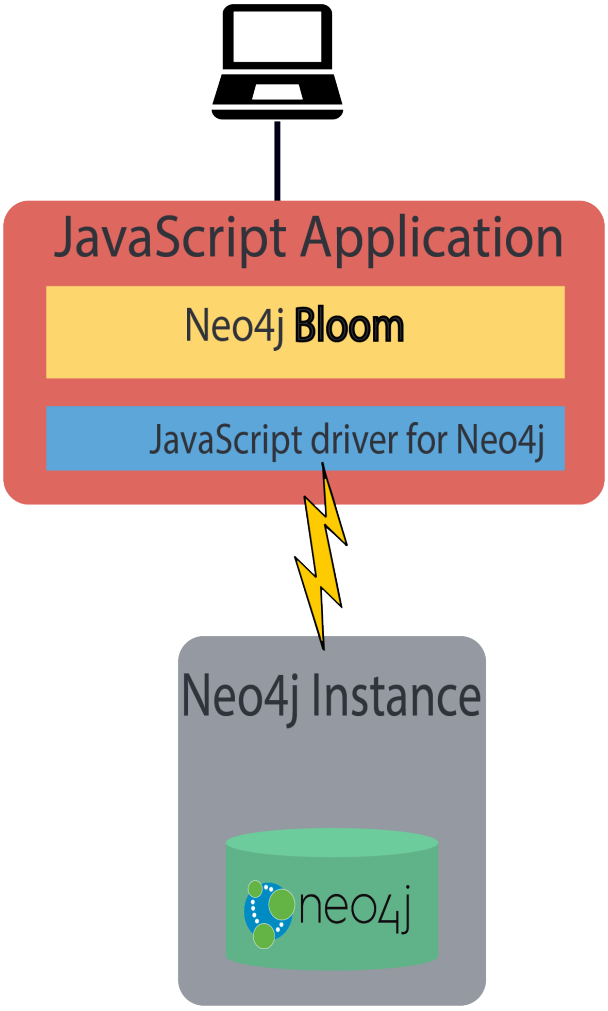
Some decisions you may need to make are:

- Number and locations of data centers
- Number and location of Core Servers and Read Replicas
- Sizing of servers (hardware and CPU)
- How to route requests in order to obtain optimal performance
- Which servers to use as backup servers

We will cover the configuration, management and monitoring of Causal Clusters in depth later in this training.

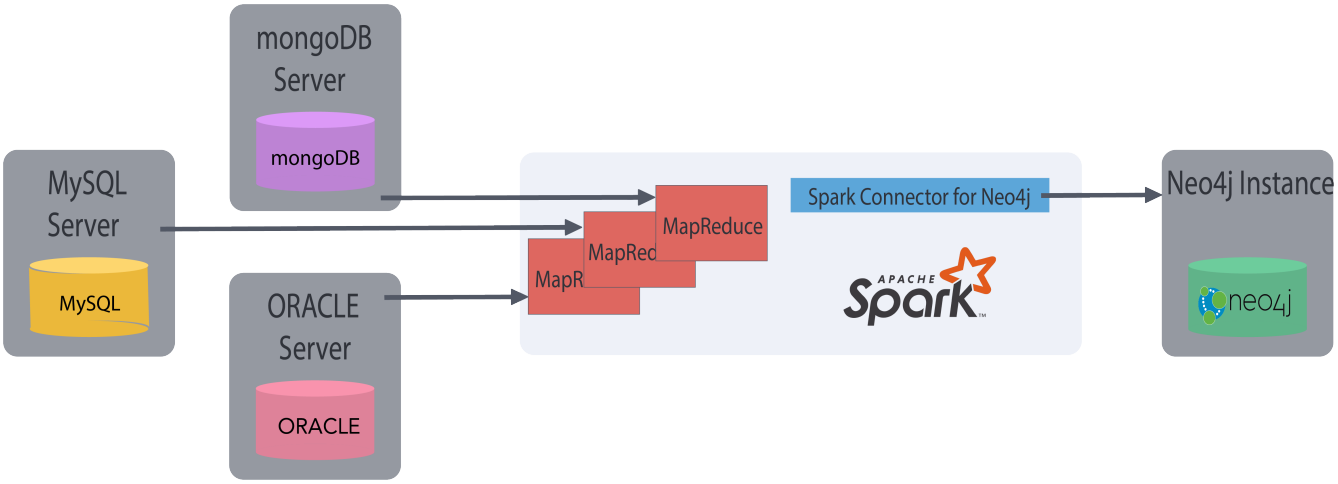
Example architecture #4: Using Neo4j standalone for graph analytics

A common use case for Neo4j is for data scientists analyze complex patterns in connected data. Neo4j Bloom, or some other data visualization tool, may be used as the front-end. The following are two common patterns this use case: . Neo4j is used to find and analyze connections in data from other data sources. To cater for this, standalone Neo4j database is loaded with data from several sources. As a Neo4j database administrator, you will ensure that the Neo4j database used for this purpose is kept up-to-date and that the data is secure and appropriately backed up. . Data scientists analyze the data in a Neo4j production database. In order to safeguard the production database from potentially heavy queries, a dedicated Read Replica is set . In this case, the analytics database is always up-to-date with the production database, and its administration is a part of the regular Causal Cluster maintenance work (see Example architecture #3).



Example architecture #5: Neo4j integrated with compute-heavy solutions

Some enterprises need to consolidate large amounts of data for analysis. The data can come from data lakes, NoSQL databases, document stores, relational, all of which is analyzed and placed into a Neo4j database for analytics. A common architecture for streaming and analyzing data for consolidations is Apache Spark.



As a Neo4j database administrator, you will work with architects to ensure that the Neo4j database in this type of environment is properly configured and available to the computational engines that

will write to the graph.

Neo4j versions

All supported versions of Neo4j are available on the Neo4j download page. On the same page, you can also find pre-releases of the next release. These offer an opportunity to explore coming features. However, it is important to note that functionality in preview releases can be changed without notice. Additionally, the preview releases are not certified for production use.

General Availability releases are our certified releases. They include all features and functionalities intended for that version and supported for production deployments. Production deployments should only use *General Availability* releases.

WARNING	Data migration between non *General Availability* versions is unsupported.

Neo4j Editions

There are two versions of Neo4j to choose from: Community Edition and Enterprise Edition. The version you use will depend on the features you require, the nature of your application that uses Neo4j, and the level of professional support you would like to receive from Neo4j.

A full comparison between the Community and Enterprise Editions for the current release of Neo4j can be found at Compare Community and Enterprise Editions.

Community Edition

The Community Edition is a full functioning version of Neo4j suitable for single instance deployments. It has full support for key Neo4j features, such as ACID compliance, Cypher and access via the binary protocol and HTTP APIs. It is ideal for smaller internal or do-it-yourself projects that do not require high levels of scaling or professional services and support. The Community Edition is free to download and use and is available from the Neo4j download page or from the Neo4j GitHub repository.

Enterprise Edition

The Enterprise Edition extends the functionality of the Community Edition to include key features for performance and scalability such as a clustering architecture for high availability and online backup functionality. It is the right choice for production systems with availability requirements or needs for scaling up or out. Enterprise Edition requires a license from Neo4j. You can download Neo4j Enterprise Edition from the Neo4j download page. When you install Neo4j Enterprise Enterprise, you have a 30 day evaluation license.

Installing Neo4j

After you have determined which Edition and version of Neo4j you must install for your application, you should follow the steps outlined in the Neo4j Operations Manual. The instructions include actions that must be taken before installing Neo4j. Later in this training, you will learn how to get started managing and monitoring a Neo4j Enterprise Edition.

Upgrading Neo4j

The procedure for upgrading your Neo4j installation will depend upon what release you are upgrading from and to. If you are upgrading to a major release, the upgrade may include a data migration step. The Neo4j Operations Manual provides instructions for upgrading a Neo4j installation.

Supported software and hardware

To install and use Neo4j, the system(s) that host Neo4j Enterprise Edition have specific requirements for:

- JVM
- Operating system
- Hardware architecture
- Memory
- Disk
- Filesystem

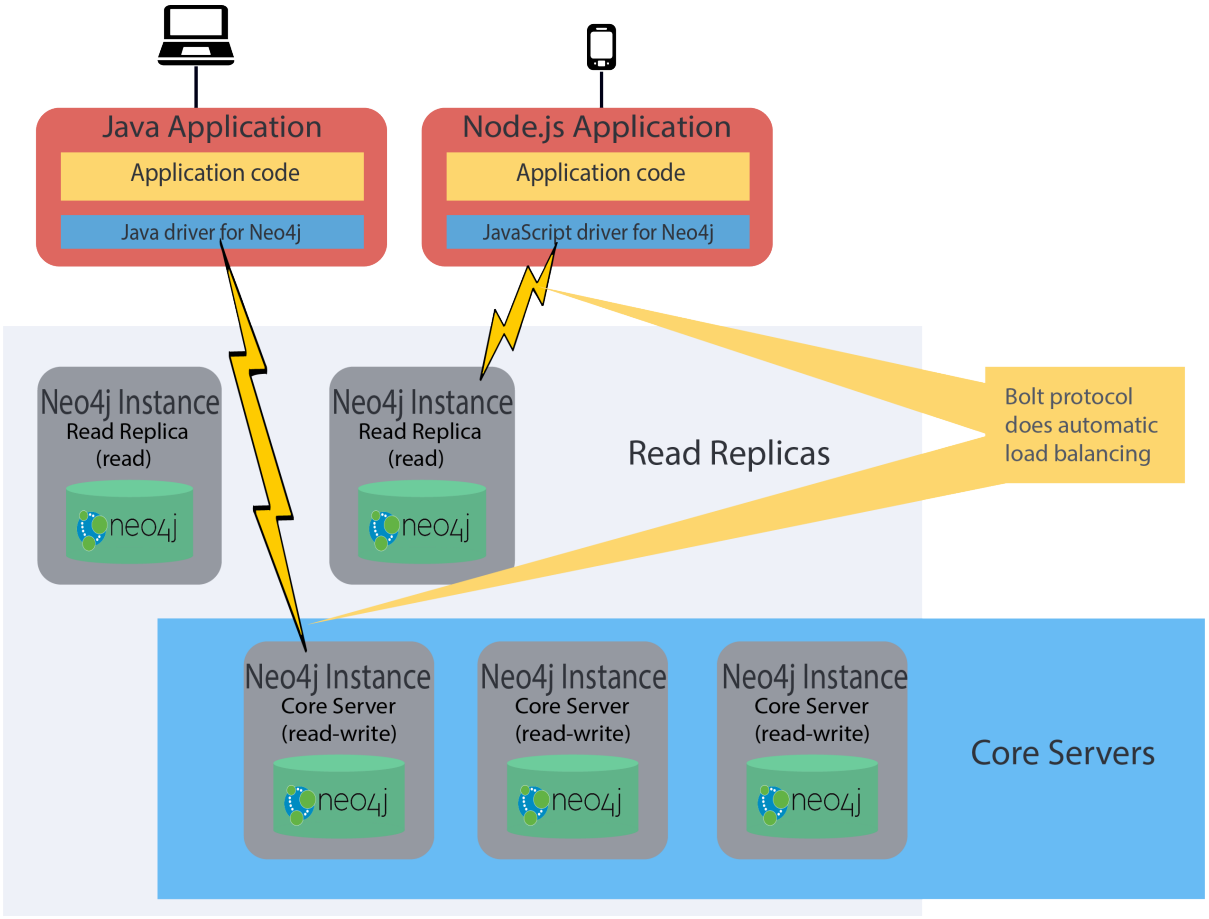
You should consult the System Requirements to learn more.

Neo4j deployment options

Early in the development of an application, a decision will be made whether to deploy Neo4j as a standalone server, embedded within the application, or to deploy Neo4j in the Cloud.

Server mode deployments

This is the most common deployment, and it is recommended for all onsite deployments that do not require embedded mode. In this architecture, Neo4j runs as a database server and can be accessed through binary and http APIs for data querying and updating.



Key advantages of server mode

- **Binary Bolt Protocol** or HTTP APIs allowing clients to send requests and receive responses over the wire
- When using one of the supported drivers together with Causal Clustering, **load balancing** is provided by Neo4j. It can also be configured to meet specific criteria.
- **Platform independence** for the client/application accessing the server APIs due to dedicated language drivers
- Neo4j and query language extensions via user defined procedures
- Independent management of the database from the application
- Easy configuration and provisioning of Neo4j instances

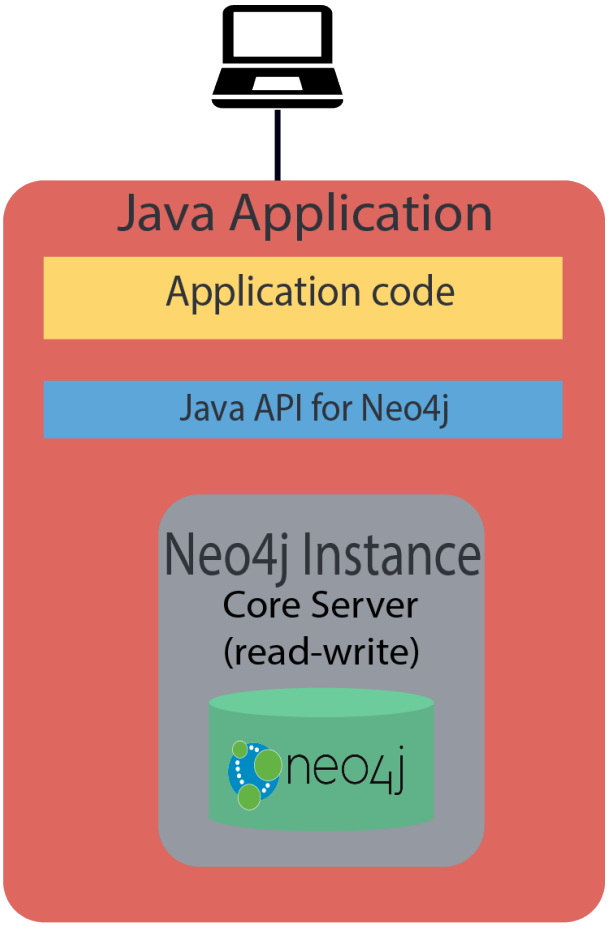
For the Neo4j database administrator

When deployed as a server, you should expect to perform many common administration, configuration, and troubleshooting tasks as you would for any database deployment. Neo4j provides utilities and tools for enabling these capabilities, which you will learn about later in this training. This training teaches you about the configuration that will aid you in ensuring a robust and stable deployment.

This training will primarily emphasize the production decisions and administrative tasks associated with a server mode deployment.

Embedded deployments

This architecture is more common in OEM setups, where Neo4j runs integrated as part of a third party product. When Neo4j is embedded directly into an application, you get all the functionality of the database directly accessible through the Java APIs right from the application code. This makes it very easy to directly work with the database and get lightning fast performance.



Key advantages of embedded mode

- Low latency: Since Neo4j is in the application, there will be no network overhead.
- Choice of APIs: Access to a diversity of APIs for creating and querying data via the Neo4j Core API, traversal framework and Cypher query language.

For the Neo4j database administrator

Your hands-on administration work with Neo4j is very minimal with embedded deployments. Most of the core administration tasks, such as initiating backups, monitoring, and configuration should be built into the application that is embedding Neo4j.

Neo4j in the Cloud

Neo4j has been successfully hosted in these environments:

- Amazon EC2
- Google Cloud (GCP)
- Microsoft Azure
- Kubernetes

To learn more about Neo4j Cloud deployments see the Developer Guide to Cloud Deployment that is updated regularly.

Administrative tasks for Neo4j

In this training, you will learn how to perform these common administrative tasks for a server mode deployment:

- Downloading and installing Neo4j
- Managing a Neo4j database
- Managing plugins used with the Neo4j database
- Managing logging
- Managing caching
- Monitoring queries
- Backing up and restoring databases
- Managing Causal Clusters
- Managing database security
- Performance tuning
- Troubleshooting problems

Exercise: Setting up Neo4j Enterprise Edition on your system

We are using Neo4j Enterprise Edition for the hands-on exercises in this training, as some Neo4j features are not included in Community Edition. Neo4j Enterprise Edition is available for download with a 30 day evaluation license. All screen captures for this training are from an *Amazon EC2 debian instance* and will vary from what you experience if you are running on a different platform.

Before you begin

If you have already installed Neo4j Desktop, you should stop any database that is active and shut down Neo4j Desktop.

You should refer to the Neo4j Operations Manual for instructions that are specific to your platform.

Exercise steps:

1. Except for debian-based distributions, download Neo4j from the Enterprise Edition download page as follows:

a. Select the latest release of Neo4j Enterprise Edition for your platform.

b. You must provide some identifying information to start your 30 day evaluation.

c. You will receive an email with a link for downloading the software, as well as instructions for installing it on your system.

2. Install Neo4j Enterprise Edition following the instructions for your platform. Ensure that the system requirements are met (specifically, the version of Java required).

3. Confirm that files have been installed as described here.

References

You should consult the Neo4J Operations Manual for more information about the installation requirements and procedures for your target platform, as well as details for Neo4j administration tasks.

The Java Developer Manual covers using embedded mode for Neo4j and intended for developers.

Check your understanding

Question 1

Suppose your organization has two applications that require Neo4j. Each application uses a different Neo4j database and the clients access the database in server mode. For these two databases, how many Neo4j installations are required?

Select the correct answer.

- ▢ one, that will service two databases.
- ▢ two, one Neo4j installation for each database.
- ▢ two, one Core Server, and one Read Replica Server.
- ▢ three, two Core Servers, and one Read Replica Server.

Question 2

Which features below are available only in the Enterprise Edition of Neo4j?

Select the correct answers.

- ▢ ACID transactions
- ▢ Causal Clusters
- ▢ Bolt protocol
- ▢ Hot backups

Question 3

What type of process must Neo4j run in?

Select the correct answer.

- ▢ Daemon
- ▢ JVM
- ▢ ESB Container
- ▢ Kubernetes

Summary

You should now be able to:

- Describe common application architectures that use Neo4j.
- Determine which edition of Neo4j to use.
- Download a specific Neo4j version.
- Determine which deployment option to use.
- Describe the administrative tasks for Neo4j.
- Install Neo4j Enterprise Edition.