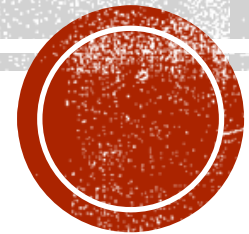


# NOSQL UNIT-1

Dr. SELVA KUMAR S

B.M.S COLLEGE OF ENGINEERING

















# AGENDA

- Introduction to NoSQL
- Different NoSQL Products
- Exploring MongoDB Java/Ruby/Python statements
- Interfacing and Interacting with NoSQL

# NOSQL!

NoSQL databases are currently a hot topic in some parts of computing, with over a hundred different NoSQL databases.

Document Database	Graph Databases
  	 
Wide Column Stores	Key-Value Databases
   	    

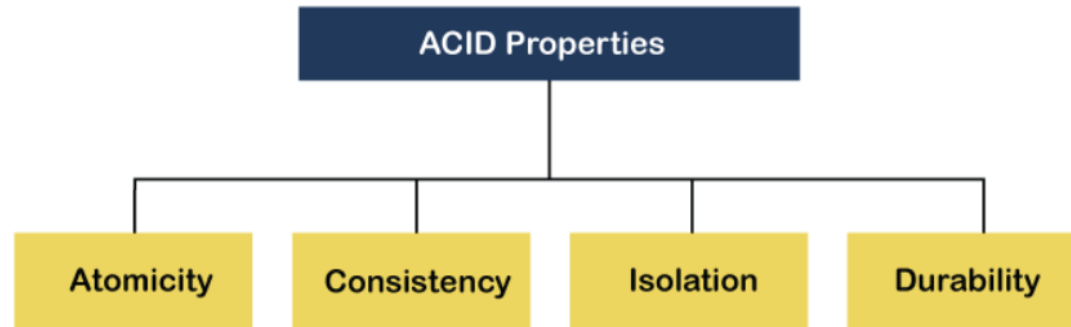


# RDBMS CHARACTERISTICS

- Data stored in columns and tables
- Relationships represented by data
- Data Manipulation Language
- Data Definition Language
- Transactions
- Abstraction from physical layer
- Applications specify what, not how
- Physical layer can change without modifying applications
  - Create indexes to support queries
  - In Memory databases



# TRANSACTIONS – ACID PROPERTIES



- Atomic – All of the work in a transaction completes (commit) or none of it completes
- Consistent – A transaction transforms the database from one consistent state to another consistent state. Consistency is defined in terms of constraints.
- Isolated – The results of any changes made during a transaction are not visible until the transaction has committed.
- Durable – The results of a committed transaction survive failures



# NO SQL?

- NoSQL stands for:
  - No Relational
  - No RDBMS
  - Not Only SQL
- NoSQL is an umbrella term for all databases and data stores that don't follow the RDBMS principles
  - A class of products
  - A collection of several (related) concepts about data storage and manipulation
  - Often related to large data sets



# WHERE DOES NOSQL COME FROM?

- Non-relational DBMSs are not new
- But NoSQL represents a new incarnation
  - Due to massively scalable Internet applications
  - Based on distributed and parallel computing
- Development
  - Starts with Google
  - First research paper published in 2003
  - Continues also thanks to Lucene's developers/Apache (Hadoop) and Amazon (Dynamo)
  - Then a lot of products and interests came from Facebook, Netflix, Yahoo, eBay, Hulu, IBM, and many more



# DYNAMO AND BIGTABLE

- Three major papers were the seeds of the NoSQL movement
  - BigTable (Google)
  - Dynamo (Amazon)
    - Distributed key-value data store
    - Eventual consistency
  - CAP Theorem



# NOSQL AND BIG DATA

- NoSQL comes from Internet, thus it is often related to the “big data” concept
- How much big are “big data”?
  - Over few terabytes Enough to start spanning multiple storage units
- Challenges
  - Efficiently storing and accessing large amounts of data is difficult, even more considering fault tolerance and backups
  - Manipulating large data sets involves running immensely parallel processes
  - Managing continuously *evolving schema* and metadata for *semi-structured and un-structured* data is difficult



# HOW DID WE GET HERE?

- Explosion of social media sites (Facebook, Twitter) with large data needs
- Rise of cloud-based solutions such as Amazon S3 (simple storage solution)
- Just as moving to dynamically-typed languages (Python, Ruby, Groovy), a shift to dynamically-typed data with frequent schema changes
- Open-source community



# WHY ARE RDBMS NOT SUITABLE FOR BIG DATA

- The context is Internet
- RDBMSs assume that data are
  - Dense
  - Largely uniform (structured data)
- Data coming from Internet are
  - Massive and sparse
  - Semi-structured or unstructured
- With massive sparse data sets, the typical storage mechanisms and access methods get stretched



# NOSQL DISTINGUISHING CHARACTERISTICS

- Large data volumes
  - Google's "big data"
- Scalable replication and distribution
  - Potentially thousands of machines
  - Potentially distributed around the world
- Queries need to return answers quickly
- Mostly query, few updates
- Asynchronous Inserts & Updates
- Schema-less
- ACID transaction properties are not needed – BASE
- CAP Theorem
- Open source development



# NOSQL VS. SQL

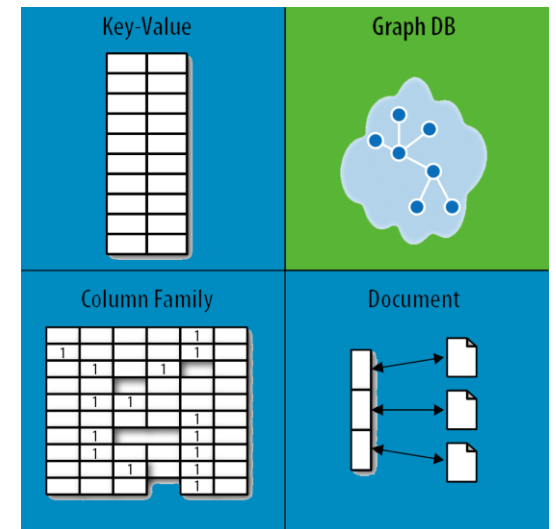
SQL DB	NoSQL DB
Examples: DB2, MySQL, Oracle, Postgress, SQL server	Examples: CouchDb MongoDB, RavenDb, Redis, Cassandra, Hbase, Neo4j, BigTable
These are called RDBMS.	These are called not only SQL database.
Based on ACID properties i.e. Atomicity, Consistency, Isolation and Durability	Based on CAP properties i.e. ( Consistency, Availability and Partition tolerance )
These are table based database i.e. the data are stored in a table with rows and columns.	These databases are document based, key-value pairs or graph based etc.
These are standard schema based (predefined schema)	These are not standard schema based( dynamic schema)
These are scaled vertically. Load can be managed by increasing CPU, RAM etc in the same server.	These are scaled horizontally. A few servers can be added to manage large traffic.
Not preferred for large/big data sets.	Preferred for large/big data sets.
Preferred for complex query execution	Not preferred for complex query execution



# NOSQL DATABASE TYPES

Discussing NoSQL databases is complicated because there are a variety of types:

- Sorted ordered Column Store
  - Optimized for queries over large datasets, and store columns of data together, instead of rows
- Document databases:
  - pair each key with a complex data structure known as a document.
- Key-Value Store :
  - are the simplest NoSQL databases. Every single item in the database is stored as an attribute name (or 'key'), together with its value.
- Graph Databases :
  - are used to store information about networks of data, such as social connections.



# DOCUMENT DATABASES (DOCUMENT STORE)

- Documents
  - Loosely structured sets of key/value pairs in documents, e.g., XML, JSON
  - Encapsulate and encode data in some standard formats or encodings
  - Are addressed in the database via a unique key
  - Documents are treated as a whole, avoiding splitting a document into its constituent name/value pairs
- Allow documents retrieving by keys or contents
- Notable for:
  - MongoDB (used in FourSquare, Github, and more)
  - CouchDB (used in Apple, BBC, Canonical, Cern, and more)



# DOCUMENT DATABASES (DOCUMENT STORE)

- The central concept is the notion of a "document" which corresponds to a row in RDBMS.
- A document comes in some standard formats like JSON (BSON).
- Documents are addressed in the database via a unique *key* that represents that document.
- The database offers an API or query language that retrieves documents based on their contents.
- Documents are schema free, i.e., different documents can have structures and schema that differ from one another. (An RDBMS requires that each row contain the same columns.)

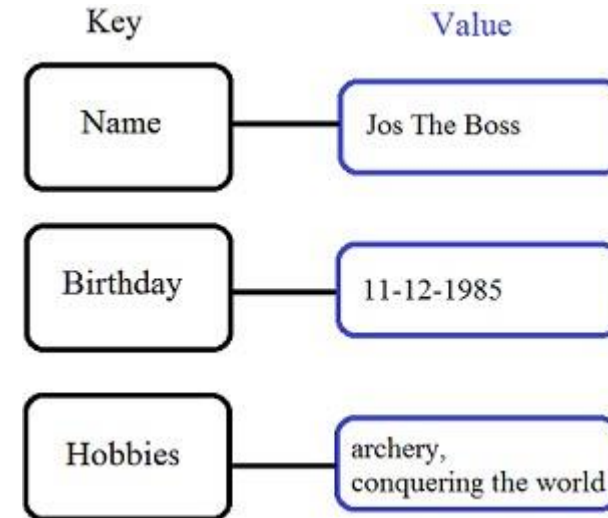
# DOCUMENT DATABASES, JSON

```
{
  _id: ObjectId("51156a1e056d6f966f268f81"),
  type: "Article",
  author: "Derick Rethans",
  title: "Introduction to Document Databases with MongoDB",
  date: ISODate("2013-04-24T16:26:31.911Z"),
  body: "This arti..."
},
{
  _id: ObjectId("51156a1e056d6f966f268f82"),
  type: "Book",
  author: "Derick Rethans",
  title: "php|architect's Guide to Date and Time Programming with PHP",
  isbn: "978-0-9738621-5-7"
}
```



# KEY/VALUE STORES

- Store data in a schema-less way
- Store data as maps
  - HashMaps or associative arrays
  - Provide a very efficient average running time algorithm for accessing data
- Notable for:
  - Couchbase (Zynga, Vimeo, NAVTEQ, ...)
  - Redis (Craigslist, Instagram, StackOverflow, flickr, ...)
  - Amazon Dynamo (Amazon, Elsevier, IMDb, ...)
  - Apache Cassandra (Facebook, Digg, Reddit, Twitter,...)
  - Voldemort (LinkedIn, eBay, ...)
  - Riak (Github, Comcast, Mochi, ...)



# SORTED ORDERED COLUMN-ORIENTED STORES

- Data are stored in a column-oriented way
  - Data efficiently stored
  - Avoids consuming space for storing nulls
  - Columns are grouped in column-families
  - Data isn't stored as a single table but is stored by column families
  - Unit of data is a set of key/value pairs
    - Identified by "row-key"
    - Ordered and sorted based on row-key
- Notable for:
  - Google's Bigtable (used in all Google's services)
  - HBase (Facebook, StumbleUpon, Hulu, Yahoo!, ...)

**Row Store**

Last Name	First Name	E-mail	Phone #	Street Address

- + Easy to add a new record
- Might waste time reading in unnecessary data

**Column Store**

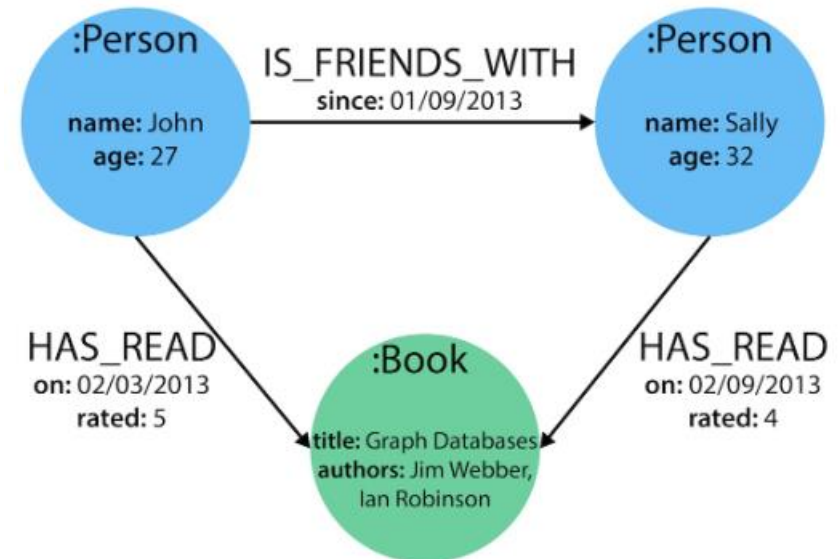
Last Name	First Name	E-mail	Phone #	Street Address

- + More data value locality
- Inserts and SELECT \* might require multiple seeks



# GRAPH DATABASES

- Graph-oriented
- Everything is stored as an edge, a node or an attribute.
- Each node and edge can have any number of attributes.
- Both the nodes and edges can be labelled.
- Labels can be used to narrow searches.



# DEALING WITH BIG DATA AND SCALABILITY

- Issues with scaling up when the dataset is just too big
- RDBMS were not designed to be distributed
- Traditional DBMSs are best designed to run well on a “single” machine
  - Larger volumes of data/operations requires to upgrade the server with faster CPUs or more memory known as ‘scaling up’ or ‘Vertical scaling’
- NoSQL solutions are designed to run on clusters or multi-node database solutions
  - Larger volumes of data/operations requires to add more machines to the cluster, Known as ‘scaling out’ or ‘horizontal scaling’
  - Different approaches include:
    - Master-slave
    - Sharding (partitioning)



# NOSQL, NO ACID

- RDBMSs are based on ACID (Atomicity, Consistency, Isolation, and Durability) properties
- NoSQL
  - Does not give importance to ACID properties
  - In some cases completely ignores them
- In distributed parallel systems it is difficult/impossible to ensure ACID properties
  - Long-running transactions don't work because keeping resources blocked for a long time is not practical



# BASE TRANSACTIONS

- Acronym contrived to be the opposite of ACID
  - **B**asically **A**vailable,
  - **S**oft state,
  - **E**ventually **C**onsistent
- Characteristics
  - Weak consistency – stale data OK
  - Availability first
  - Best effort
  - Approximate answers OK
  - Aggressive (optimistic)
  - Simpler and faster

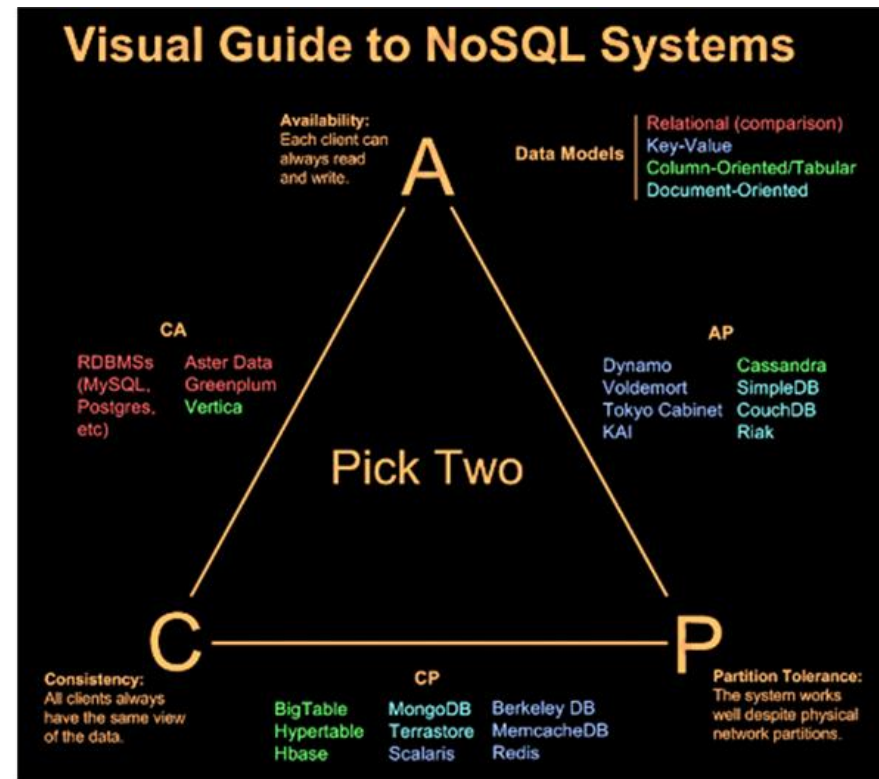


# CAP THEOREM

A congruent and logical way for assessing the problems involved in assuring ACID-like guarantees in distributed systems is provided by the CAP theorem

At most two of the following three can be maximized at one time

- Consistency
  - Each client has the same view of the data
- Availability
  - Each client can always read and write
- Partition tolerance
  - System works well across distributed physical networks



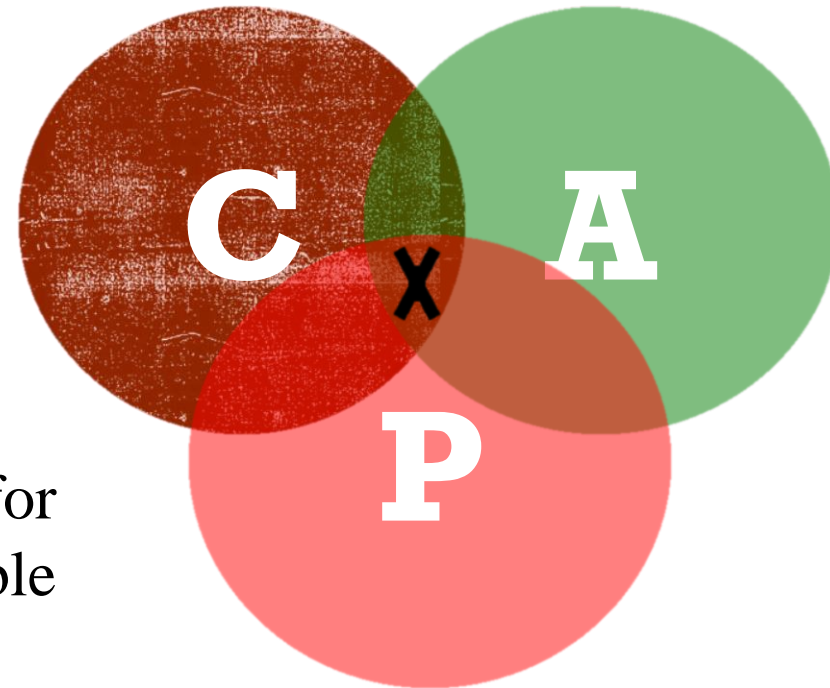
# CAP THEOREM: TWO OUT OF THREE

- CAP theorem – At most two properties on three can be addressed
- The choices could be as follows:
  1. Availability is compromised but consistency and partition tolerance are preferred over it
  2. The system has little or no partition tolerance. Consistency and availability are preferred
  3. Consistency is compromised but systems are always available and can work when parts of it are partitioned



# CONSISTENCY OR AVAILABILITY

- Consistency and Availability is not “binary” decision
- AP systems relax consistency in favor of availability – but are not inconsistent
- CP systems sacrifice availability for consistency- but are not unavailable
- This suggests both AP and CP systems can offer a degree of consistency, and availability, as well as partition tolerance



# PERFORMANCE

- There is no perfect NoSQL database
- Every database has its advantages and disadvantages
  - Depending on the type of tasks (and preferences) to accomplish
- NoSQL is a set of concepts, ideas, technologies, and software dealing with
  - Big data
  - Sparse un/semi-structured data
  - High horizontal scalability
  - Massive parallel processing
- Different applications, goals, targets, approaches need different NoSQL solutions



# WHERE WOULD I USE IT?

- Where would I use a NoSQL database?
- Do you have somewhere a large set of uncontrolled, unstructured, data that you are trying to fit into a RDBMS?
  - Log Analysis
  - Social Networking Feeds (many firms hooked in through Facebook or Twitter)
  - External feeds from partners
  - Data that is not easily analyzed in a RDBMS such as time-based data
  - Large data feeds that need to be massaged before entry into an RDBMS



# GETTING INITIAL HANDS-ON EXPERIENCE

- Create account : Datastax.com

The screenshot displays the DataStax Astra dashboard for a user named Selva Kumar S. The interface includes a top navigation bar with a 'Live Chat' button and a user profile dropdown. A left sidebar contains navigation links for 'Dashboard', 'Databases', 'BMS', and 'Streaming', along with a 'Sample App Gallery' and 'Other Resources' section. The main content area features a 'Dashboard' header, a promotional banner for free credits, and a 'Current Plan' section showing a 'Free' plan with '\$25.00' in credits remaining and '\$0.00' spent. A 'Usage For Current Billing Period' section shows 0 read/write requests, 46.37 KB of storage consumed, and 11.59 MB of data transfer. An 'Active Database (1)' table is partially visible at the bottom.

**DataStax Astra**

Current Organization: selva.cse@bmsce.ac.in

**Dashboard**

Claim up to \$3,000 in free credits for your qualified Startup! [Dismiss] [Request]

**Current Plan**

Free

Credits Remaining: **\$25.00**

\$0.00 spent / \$25.00

Want to upgrade your Astra serverless experience? Add a credit card to get more. [Upgrade]

**New!** Invite Friends, Get More Free Astra Credits

Both you and your friends will get \$25 in Astra credits when they sign up. That's worth 20 million reads/writes, and 80GB storage! [Invite Friends]

**Databases** [Create Serverless Database]

**Usage For Current Billing Period**

Read Requests	Write Requests	Storage Consumed	Data Transfer
0	0	46.37 KB	11.59 MB

**Active Database (1)**

Name	Database ID	Reads	Writes	Storage	Data Transfer	Status
------	-------------	-------	--------	---------	---------------	--------

# CREATE A DATABASE

DataStax  
**Astra**

## Create a Database

1

Enter the Basic Details

Database Name

Database Name

This field is required.

Keyspace Name ⓘ


Keyspace Name


Want to know more? [Read our Docs](#) to learn more about keyspaces.


2

Select a Provider and Region

You've got access to 3 free regions in GCP. Unlock all regions by [upgrading to the Pay as you go plan](#).

 Google Cloud

 Amazon Web Services

 Microsoft Azure

Unlock All Regions

Select an Area

Select a Region

North America

0 of 6 regions selected

# DASHBOARD

Dashboard / Serverless Databases

BMS

Active

Load Data

Connect

Overview

Health

Connect

CQL Console

CDC

Settings

Usage For Current Billing Period for BMS ⓘ

Status Active

Read Requests

0

Write Requests

0

Storage Consumed ⓘ

46.37 KB


Data Transfer

11.59 MB

Regions

Add Region

Our Pay as you go plan allows you to add multiple regions.

Provider	Area	Region	Region Name	Datacenter ID	Region Availability
 Google Cloud	Asia Pacific	asia-south1	Mumbai, India	1864e0cc-c2bc-47db-90a5-27980de93337-1 ⓘ	Online ⓘ

Keyspaces

Add Keyspace

Managing multiple applications? Consider keeping each application in a separate keyspace – whatever a [keyspace](#) is.

Keyspace



31

# CONNECT TAB

[Dashboard](#) / [Serverless Databases](#)

**BMS** Active

[Load Data](#) [Connect](#)

[Overview](#) [Health](#) **[Connect](#)** [CQL Console](#) [CDC](#) [Settings](#)

**Quick Start Guide** 🚀

New to Astra and looking for more help? Check out our getting started guide to learn how to load sample data, sample apps, and more.

[Get Instructions](#) [Skip Instructions](#)


### Select a Method


Not sure which method to choose? Get a [quick overview](#) of connection methods. [Need more met](#)


[APIs](#) **[Drivers](#)** [SDK](#) [Integrations](#)


We offer two types of Drivers - [Cloud Native](#) and Legacy. We suggest using the [Cloud Native driver](#) for leveraging more modern APIs and Legacy driver for working with the CQL APIs. The instructions below assume you are adding driver support to an existing application.


Legacy ▼

☒  Node.js

☐  Python

☐  Java

☐  C++

☐  C#

32

# CQL CONSOLE

[Dashboard](#) / [Serverless Databases](#)

**BMS** Active

[Overview](#) [Health](#) [Connect](#) **[CQL Console](#)** [CDC](#) [Settings](#)

## Connect to your CQL Console

Interact with your database through Cassandra Query Language (CQL). Need help? Check out our [quick reference guide on CQL](#).  
Alternatively, you can connect to your Astra DB database using the [standalone version of CQLSH](#).

```
Connected as selva.cse@bmsce.ac.in.  
Connected to cndb at cassandra.ingress:9042.  
[cqlsh 6.8.0 | Cassandra 4.0.0.6816 | CQL spec 3.4.5 | Native protocol v4]  
Use HELP for help.  
token@cqlsh> 
```

# CQL DATA DEFINITION COMMANDS

- **CREATE KEYSPACE** – Creates a KeySpace in Cassandra.
- **USE** – Connects to a created KeySpace.
- **ALTER KEYSPACE** – Changes the properties of a KeySpace.
- **DROP KEYSPACE** – Removes a KeySpace
- **CREATE TABLE** – Creates a table in a KeySpace.
- **ALTER TABLE** – Modifies the column properties of a table.
- **DROP TABLE** – Removes a table.
- **TRUNCATE** – Removes all the data from a table.
- **CREATE INDEX** – Defines a new index on a single column of a table.
- **DROP INDEX** – Deletes a named index.

# CQL DATA MANIPULATION COMMANDS

- **INSERT** – Adds columns for a row in a table.
  - **UPDATE** – Updates a column of a row.
  - **DELETE** – Deletes data from a table.
  - **BATCH** – Executes multiple DML statements at once.
- 
- CQL Clauses
  - **SELECT** -- This clause reads data from a table

# DEMO

- describe keyspaces;
- or desc keyspaces;
- use demo;
- describe tables;
- describe table emp;
- describe type emp;

# DEMO CONTD..

- `create table student ( name text, usn text, mob int, id int, primary key(id));`
- `desc student;`
- `select * from student;`
- `insert into student(id, usn,name,mob) values (1, 'lbm20cs001','abc',2334233423);`
- `update student set name='xyz' where id=1;`
- `update student set name='xyz' where id=2; // no error instead create new row`

# DEMO CONTD..

- `CREATE TABLE emp( emp_id int PRIMARY KEY, emp_name text, emp_city text, emp_sal varint, emp_phone varint );`
- `Select * from emp;`
- `INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES(1,'ram', 'Hyderabad', 9848022338, 50000);`
- `UPDATE emp SET emp_city='Delhi',emp_sal=50000 WHERE emp_id=2;`
- `DELETE emp_sal FROM emp WHERE emp_id=3;`

# DEMO CONTD..

- `uuid()` function which is very important to insert value and to uniquely generates “guaranteed unique” UID value Universally.
- `Create table function4(Id uuid primary key, name text);`
- `Insert into function4 (Id, name) values (1, 'Ashish'); // fails`
- `Insert into function4(Id, name) values (now(), 'Ashish'); //correct`

# DEMO CONTD..

- Alter table emp add (emails set<text>, frameworks list<text>, previous\_jobs map<text, int>);
- INSERT INTO emp (emp\_id, emp\_name, emp\_city, emp\_phone, emp\_sal, emails, frameworks, previous\_jobs) VALUES(6, 'ram', 'Hyderabad', 9848022338, 50000, {'ram@gmail.com', 'ram@hotmail.com'}, ['C#', 'java', 'node.js'], {'Amazon':4, 'TCS':3});

# INTERFACING AND INTERACTING WITH NOSQL

- In programming language to connect application with database there is a programming Pattern.
- Three Easy steps are following :
  - Create a connection (which is called a Session)
  - Use the session to execute the query.
  - Close the connection/session.

# JAVA CODE TO CONNECT DB

- **Step-1:**

To create the session used the following Java code.

```
try (DseSession session = DseSession.builder()
    .withCloudSecureConnectBundle
      ("/path/to/secure-connect-database_name.zip")
    // Database Credentials
    .withAuthCredentials("DBUserName", "DBPassword")
    .build()) {
```

- **Step-2:**

To execute the CQL used the following Java code.

```
session.execute(
    SimpleStatement.builder("SELECT password
                            FROM keyspace-name.Table-name
                            WHERE email = ?")
    .addPositionalValues("name@datastax.com")
    .build());
```

- **Step-3:**

To close the Session used the following Java code.

```
// Close happens automatically here
// - otherwise use session.close()
session.close()
```

# PYTHON CODE TO CONNECT DB

- **Step-1:**

To create the session used the following Python code.

```
cluster = Cluster(  
    cloud = {'secure_connection_bundle'  
            : '/path / to / secure-connect-database_name.zip'},  
    auth_provider = PlainTextAuthProvider('DBUsername', 'DBPassword'))  
    # Database Credentials  
session = cluster.connect()
```

- **Step-2:**

To execute the CQL used the following Python code.

```
session.execute(("SELECT password  
                FROM keyspace-name.Table-name  
                WHERE email = % s, ('name@datastax.com'))
```

- **Step-3:**

To close the Session used the following Python code.

```
session.shutdown()
```

# MONGODB —DOCUMENT DB

## ■ MongoDB Atlas

🔒 mongodb.com/cloud/atlas/lp/try4?utm\_source=google&utm\_campaign=search\_gs\_pl\_evergreen\_atlas\_core\_prosp-brand\_gic-null\_apac-in\_ps-all\_desktop\_eng\_lead&utm...



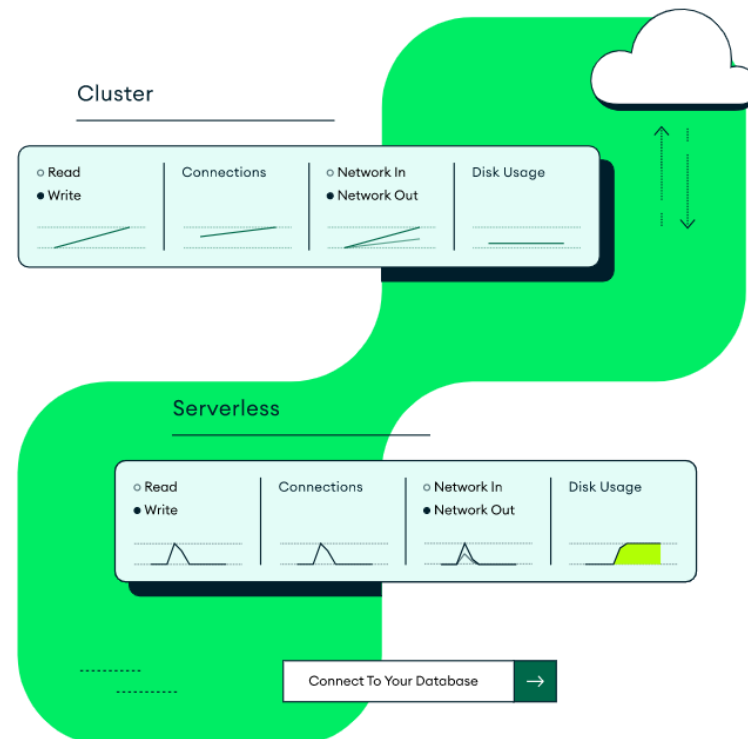
MONGODB ATLAS

**MongoDB Atlas.** Fully managed MongoDB in the cloud.

Harness the power of your data by building and managing your data in the cloud.

Start Free

View pricing →



# MongoDB Atlas

## ✓ Work with your data as code

Documents in MongoDB map directly to objects in your programming language. Modify your schema as your apps grow over time.

## ✓ Focus on building, not managing

Let MongoDB Atlas take care of the infrastructure operations you need for performance at scale, from always-on security to point-in-time recovery.

## ✓ Simplify your data dependencies

Leverage application data for full-text search, real-time analytics, rich visualizations and more with a single API and minimal data movement.

## Sign up


See what Atlas is capable of for free



☐ I agree to the [Terms of Service](#) and [Privacy Policy](#).

Create your Atlas account

or

 Sign up with Google

Already have an account? [Sign in](#)



## Accept Privacy Policy & Terms of Service

Please acknowledge the following terms and conditions to finish creating your account.

☒ I accept the [Privacy Policy](#) and the [Terms of Service](#)

Cancel Signup

Submit

# Deploy a cloud database

Experience the best of MongoDB on AWS, Azure, and Google Cloud. Choose a deployment option to get started.

NEW



## Serverless

For application development and testing, or workloads with variable traffic. Minimal configuration required.

- ✓ Pay only for the operations you run
- ✓ Resources scale seamlessly to meet your workload
- ✓ Always-on security and backups

Create

Starting at  
**\$0.10/1M reads**

ADVANCED



## Dedicated

For production applications with sophisticated workload requirements. Advanced configuration controls.

- ✓ Network isolation and fine-grained access controls
- ✓ On-demand performance advice
- ✓ Multi-region and multi-cloud options available

Create

Starting at  
**\$0.08/hr\***  
\*estimated cost \$56.94/month

FREE



## Shared

For learning and exploring MongoDB in a cloud environment. Basic configuration options.

- ✓ No credit card required to start
- ✓ Explore with sample datasets
- ✓ Upgrade to dedicated clusters for full functionality

Create

Starting at  
**FREE**

## Create a Shared Cluster

Welcome to MongoDB Atlas! We've recommended some of our most popular options, but feel free to customize your cluster to your needs. For more information, check our [documentation](#).

Serverless

Dedicated

FREE Shared

For learning and exploring MongoDB in a sandbox environment. Basic configuration controls.

No credit card required to start. Upgrade to dedicated clusters for full functionality.

Explore with sample datasets. Limit of one free cluster per project.

### Cloud Provider & Region

AWS, Mumbai (ap-south-1) ^

aws

Google Cloud

Azure

★ Recommended region ⓘ

🏷️ Dedicated tier region ⓘ

FREE

**Free forever!** Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

[Back](#)

Create Cluster





## DEPLOYMENT

Database

Data Lake **PREVIEW**

## DATA SERVICES

Triggers

Data API

Data Federation

## SECURITY

### Quickstart

Database Access

Network Access

Advanced

New On Atlas **2**

To access data stored in Atlas, you'll need to create users and set up network security controls. [Learn more about security setup](#)

### 1 How would you like to authenticate your connection?

Your first user will have permission to read and write any data in your project.

Username and Password

Certificate

Create a database user using a username and password. Users will be given the *read and write to any database privilege* by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.

#### Username

#### Password

Autogenerate Secure Password

Copy

Create User

New On Atlas **2**

+ Create

...

SHARED



VERSION	REGION	CLUSTER TIER	TYPE	BACKUPS	LINKED APP SERVICES	ATLAS SEARCH
5.0.13	AWS / Mumbai (ap-south-1)	M0 Sandbox (General)	Replica Set - 3 nodes	Inactive	<a href="#">data</a>	<a href="#">Create Index</a>

# VIEW MONITORING

Project 0

Atlas

App Services

Charts



Cluster0

VERSION

5.0.13

REGION

AWS Mumbai (ap-south-1)

DEPLOYMENT

Database

PREVIEW

DATA SERVICES

Triggers

Data API

Data Federation

SECURITY

Database Access

Network Access

Advanced

Overview

Real Time

Metrics

Collections

Search

Profiler

Performance Advisor

Online Archive

Cmd Line Tools

DATABASES: 1 COLLECTIONS: 1

VISUALIZE YOUR DATA

REFRESH

+ Create Database

Search Namespaces

selva-BMS

Demo

## selva-BMS.Demo

STORAGE SIZE: 20KB

LOGICAL DATA SIZE: 50B

TOTAL DOCUMENTS: 1

INDEXES TOTAL SIZE: 20KB

Find

Indexes

Schema Anti-Patterns 0

Aggregation

Search Indexes

INSERT DOCUMENT

FILTER { field: 'value' }

OPTIONS

Apply

Reset

QUERY RESULTS: 1-2 OF 2

```
_id: ObjectId('63692389d96e35e7f60b7f85')
name: "selva"
mobile: 87767676
```



## Connect to Cluster0

✓ Setup connection security

Choose a connection method

Connect

Choose a connection method [View documentation](#)

Get your pre-formatted connection string by selecting your tool below.



**Connect with the MongoDB Shell**

Interact with your cluster using MongoDB's interactive Javascript interface



**Connect your application**

Connect your application to your cluster using MongoDB's native drivers



**Connect using MongoDB Compass**

Explore, modify, and visualize your data with MongoDB's GUI



**Connect using VS Code**

Connect to a MongoDB host in Visual Studio Code



Go Back

Close

## Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

I do not have the MongoDB Shell installed

I have the MongoDB Shell installed

1 Select your operating system and download the `mongosh`

Windows

Download mongosh (6.1.0)

or

Copy download URL

2 Add <your mongosh's download directory>/bin to your \$PATH variable. [How to](#)

3 Run your connection string in your command line

Use this connection string in your application:

```
mongosh "mongodb+srv://cluster0.eli2h0j.mongodb.net/myFirstDatabase" --apiVersion  
1 --username selvaBMS
```

Replace **myFirstDatabase** with the name of the database that connections will use by default. You will be prompted for the password for the Database User, **selvaBMS**. When entering your password, make sure all special characters are [URL encoded](#).

## Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

### 1 Select your driver and version

DRIVER

PHP

VERSION

PHPLIB 1.11 + mongodb-1.10 or later

### 2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://selvaBMS:<password>@cluster0.eli2h0j.mongodb.net/?  
retryWrites=true&w=majority
```



Replace **<password>** with the password for the **selvaBMS** user. Ensure any option params are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

# Network Access

- IP Access List
- Peering
- Private Endpoint

+ ADD IP ADDRESS

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
49.37.243.251/32 (includes your current IP address)	My IP Address	<div></div> Active	<div>EDIT</div> <div>DELETE</div>

System Status: All Good



SELVA KUMAR'S ORG - 2022-11-02 > PROJECT 0

# Data API

ENABLED

Create API Key

</> Test Your API

Data API

API Keys

Logs

Settings

i

By default, your Data API returns data as JSON which will cast Timestamp, ObjectId, Binary, and Decimal128 data types into strings. Visit the [Settings page](#) or add the appropriate header to change this return type to Extended JSON. [Learn More](#)

x

URL Endpoint:

Version: V1

https://data.mongodb-api.com/app/data-kaybt/endpoint/data/v1

Copy

Advanced Settings

View our API documentation

Data Source	Provider/Region	Tier	Data API Access
<div></div>			

System Status: All Good

Connect View Collection Help

selva

3 DBS 1 COLLECTIONS

★ FAVORITE

HOSTS

ac-7z57wy5-shard-00-02....

ac-7z57wy5-shard-00-00....

ac-7z57wy5-shard-00-01....

CLUSTER

Replica Set (atlas-6tn8d0-...

3 Nodes

EDITION

MongoDB 5.0.13 Enterprise

{ } My Queries

Databases

Filter your data

▶ admin

▶ local

▼ selva-BMS

Demo

Documents

selva-BMS.Demo



## selva-BMS.Demo

1  
DOCUMENTS1  
INDEXES

Documents

Aggregations

Schema

Explain Plan

Indexes

Validation

FILTER { field: 'value' }

▼ OPTIONS

PROJECT { field: 0 }

SORT { field: -1 } or [['field', -1]]

MAX TIME MS 60000

COLLATION { locale: 'simple' }

SKIP 0

LIMIT 0

ADD DATA



VIEW



Displaying documents 1 - 1 of 1



REFRESH

```
_id: ObjectId('63692389d96e35e7f60b7f85')
name: "selva"
mobile: 87767676
```

# DEMO

- Show dbs;
- Use database\_name;
- Db; \\ shows current database in
- db.createCollection('democollection')
- Or
- db.slstudent.insert({name:'selva'})
- Show collections;
- db.slstudent.find()
- db.createCollection('emp',{capped:true, size:5242880,max:5})
- db.emp.insert({empid:1, name:"selva", exp:{company:'TCS', exp:4}})
- db.emp.find()

# DEMO CONTD..

- `Var myemp = [ {empid:1, name:"selva"},{empid:2, name:"kumar"}]`
- `db.emp.insert(myemp) //bulk insert`
- `db.emp.find()`
  
- `db.emp.update({name:'selva'},{$set:{name:"skumar"}})`
- `db.emp.find()`
  
- `db.emp.update({name:'selva'},{$set:{name:"skumar"}}, {multi:true})`
- `db.emp.update({name:'selva'},{$set:{name:"skumar"}}, {upsert:true})`

# DEMO CONTD..

- `db.emp.remove({"name":"selva"})`
- `db.emp.find({"name":"selva"})`
- `db.emp.find().limit(2)`
- `db.emp.find().sort({name:-1})` // -1: descending order / 1: Ascending order
- `db.emp.find({empid:{$gt:1}})`
- `db.emp.find({empid : {$in:[3,6]}})`
- `db.emp.getIndexes()`
- `db.emp.createIndex({"name":1})`

# DEMO CONTD..

- `db.emp.aggregate([{"$match":{"section":"A"}}])`
- `db.emp.aggregate([{"$match":{"$and":[{"section":"A"}, {"Marks":{"$gt":70}}]})`
- `db.emp.aggregate([{"$project":{"name":1,"section":1}}])` \ \ shows id , name and section
- `db.emp.aggregate([{"$project":{"_id":0,"name":1,"section":1}}])` \ \ shows only name and section id is not visible
- `db.emp.aggregate([{"$match":{"section":"A"}}, {"$project":{"_id":0,"name":1,"section":1}}])`
- `db.emp.aggregate([{"$group":{"_id":{"section":"$section"}, "Totalmarks":{"$sum":"$Marks"}}}])`

# CONNECT USING PYTHON CODE

Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

## 1 Select your driver and version

DRIVER

Python

VERSION

3.4 or later

## 2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb://selvaBMS:<password>@ac-7z57wy5-shard-00-00.eli2h0j.mongodb.net:27017,ac-7z57wy5-shard-00-01.eli2h0j.mongodb.net:27017,ac-7z57wy5-shard-00-02.eli2h0j.mongodb.net:27017/?ssl=true&replicaSet=atlas-6tn8d0-shard-0&authSource=admin&retryWrites=true&w=majority
```



Replace **<password>** with the password for the **selvaBMS** user. Ensure any option params are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

# CODE

- `import pymongo`
- `from pymongo import MongoClient`
- `cluster = MongoClient("mongodb://selvaBMS:<password>@ac-7z57wy5-shard-00-00.eli2h0j.mongodb.net:27017,ac-7z57wy5-shard-00-01.eli2h0j.mongodb.net:27017,ac-7z57wy5-shard-00-02.eli2h0j.mongodb.net:27017/?ssl=true&replicaSet=atlas-6tn8d0-shard-0&authSource=admin&retryWrites=true&w=majority")`
- `db = cluster["selva-test"]`
- `collection=db["emp"]`
- `post={"name":"selva", "email":"selva.cse@bmsce.ac.in"}`
- `collection.insert_one(post)`

# REDIS — KEY VALUE DATABASE

- Redis stands for **REmote DIctionary Server**.
- Redis is a No SQL database which works on the concept of key-value pair.
- Redis is a flexible, open-source (BSD licensed), in-memory data structure store, used as database, cache, and message broker.
- Redis supports various types of data structures like strings, hashes, lists, sets, sorted sets and bitmaps.
- Redis is an advanced key-value store to improve performance when serving data that is stored in system memory.

# ADVANTAGES OF REDIS

- Very flexible
- No schemas and column names
- Very fast : Can perform around 110,000 SETs per second and about 81,000 GETs per second.
- Rich Datatype support
- Caching and Disk persistence

# APPLICATIONS OF REDIS

- Login and cookie caching
- Shopping cart
- Web page caching
- Database row caching
- Web page Analytics

# REDIS STRUCTURES

Structure type	What it contains	Structure read/write ability
STRING	Strings, integers, or floating-point values	Operate on the whole string, parts, increment/decrement the integers and floats
LIST	Linked list of strings	Push or pop items from both ends, trim based on offsets, read individual or multiple items, find or remove items by value
SET	Unordered collection of unique strings	Add, fetch, or remove individual items, check membership, intersect, union, difference, fetch random items
HASH	Unordered hash table of keys to values	Add, fetch, or remove individual items, fetch the whole hash
ZSET (sorted set)	Ordered mapping of string members to floating-point scores, ordered by score	Add, fetch, or remove individual values, fetch items based on score ranges or member value

# STRINGS IN REDIS

## Command

## What it does

**GET**

Fetches the data stored at the given key

**SET**

Sets the value stored at the given key

**DEL**

Deletes the value stored at the given key (works for all types)

# LIST IN REDIS

Command	What it does
<b>RPUSH</b>	Pushes the value onto the right end of the list
<b>LRANGE</b>	Fetches a range of values from the list
<b>LINDEX</b>	Fetches an item at a given position in the list
<b>LPOP</b>	Pops the value from the left end of the list and returns it

# SETS IN REDIS

Command	What it does
SADD	Adds the item to the set
SMEMBERS	Returns the entire set of items
SISMEMBER	Checks if an item is in the set
SREM	Removes the item from the set, if it exists

# HASHES IN REDIS

Command	What it does
HSET	Stores the value at the key in the hash
HGET	Fetches the value at the given hash key
HGETALL	Fetches the entire hash
HDEL	Removes a key from the hash, if it exists

# SORTED SETS IN REDIS

Command	What it does
ZADD	Adds member with the given score to the ZSET
ZRANGE	Fetches the items in the ZSET from their positions in sorted order
ZRANGEBYSCORE	Fetches items in the ZSET based on a range of scores
ZREM	Removes the item from the ZSET, if it exists

# REDIS.COM

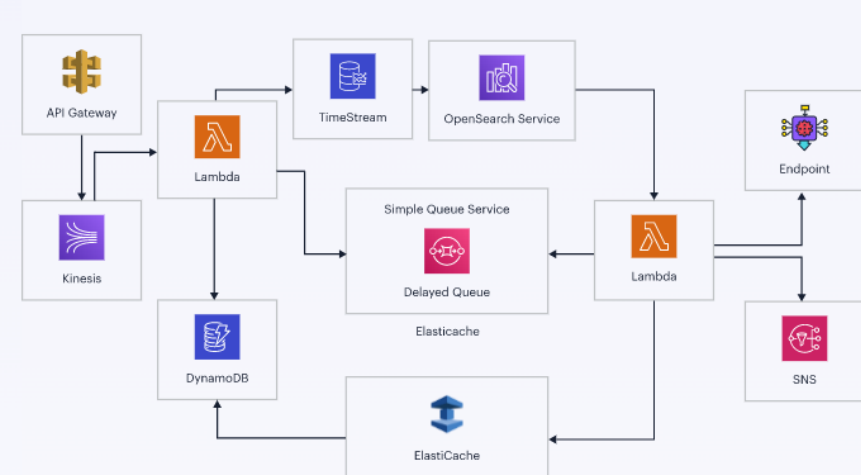


## Real-time speed and simplicity

Build blazing fast distributed apps that your customers will love and your DevOps teams can easily scale from code to production. From the makers of Redis, only Redis Enterprise unlocks the full potential of Redis.

Start Building

Talk to us



# *Get started with Redis Cloud, for free*

the creators of Redis, Redis Enterprise is the **most powerful, fully-featured Redis** you'll ever see. With this you can do the following:

- Redis as an **all-in-one database *and* cache** like never before
- Store and search JSON documents with complex relationships
- Enable real-time analytics, streaming, and microservices

No credit card required



Sign up with Google



Sign up with Github

\_\_\_\_\_ or \_\_\_\_\_

First Name

Last Name

Email

Continuing to use this site, you consent to our updated [privacy agreement](#). You can change your cookie settings at any time but parts of our site will not function correctly



Subscriptions



Databases



Data Access Control



Access Management



Logs



Account Settings



Usage Report



Billing & Payments

New subscription



Download Center

## Welcome, Selva Kumar

Let's create your **FREE** database



30MB RAM



1 dedicated database



30 connections



Seamless upgrade to Fixed plans



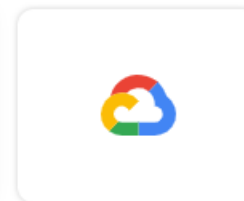
Redis Stack

Includes recommended database capabilities

Select cloud vendor



Amazon Web Services



Google Cloud Platform



Microsoft Azure

Region



US East (N. Virginia) us-east-1



Let's start free

or

Create a custom database >

## ✓ Selva Kumar free subscription

Creation date: 11-Nov-2022 | ID: #1896314

Databases

Overview

Status	Name	Endpoint	Memory
●	SelvaKumar-free-db	<a href="#">Connect</a>	6.3MB

Items per page 10

### Connect to SelvaKumar-free-db



#### > Redis CLI



Copy the following command to your command line

```
redis-cli -u redis://<username>:<password>@redis-15597.c264.ap-south-1-1.ec2.cloud.redislab.s.com:15597
```

Replace <username> and <password>

[Copy](#)

[↗ Install Redis CLI](#)



Redis Client



RedisInsight Desktop



## Connect to SelvaKumar-free-db



 Redis CLI



 Redis Client



Select your client

Python



Copy the following code snippet to your application

```
import redis

r = redis.Redis(
    host='redis-15597.c264.ap-south-1-1.ec2.clou
d.redislabs.com',
    port=15597,
    password='<password>')
```

Replace <password>

 Copy

 RedisInsight Desktop



Show all



## Connect to SelvaKumar-free-db



>\_ Redis CLI



Redis Client



RedisInsight Desktop



RedisInsight is a desktop manager that provides an intuitive and efficient GUI for Redis, allowing you to interact with your databases and manage your data.

[↗ Read more](#)

Copy the URI below to RedisInsight connection box

```
redis://<username>:<password>@redis-15597.c26  
4.ap-south-1-1.ec2.cloud.redislabs.com:15597
```

Replace **<username>** and **<password>**

Copy

Download RedisInsight

For Windows



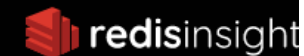
Download

[Show all](#)





## My Redis databases

[+ ADD REDIS DATABASE](#)**Limited offer: Up to 6 months free with \$200 credits.**

Try Redis Cloud with enhanced database capabilities.



Or follow the guides:

[Build from source](#)[Docker](#)[Homebrew](#)

### Discover and Add Redis Databases



#### Add Database Manually

Use Host and Port to connect to your Redis Database



#### Autodiscover Databases

Use discovery tools to automatically discover and add your Redis Databases

You can manually add your Redis databases. Enter Host and Port of your Redis database to add it to RedisInsight. [Learn more here.](#)

Host\*



Port\*

Should not exceed 65535.

Database Alias\*

Username

Password

[Cancel](#)[Add Redis Database](#)



# My Redis databases

+ ADD REDIS DATABASE

Limited offer: Up to 6 months free with \$200 credits.  
Try Redis Cloud with enhanced database capabilities. >

Or follow the guides:  
[Build from source](#) [Docker](#) [Homebrew](#)

Database List Search

<input type="checkbox"/>	Database Alias	Host:Port	Connection Type	Modules	Last connection <span>↑</span>
<input type="checkbox"/>	<a href="#">redis-15597.c264.ap-south-1...</a>	redis-15597.c264.ap-south-1-...	Standalone		less than a minute ago

WindowViewHelp

redis-15597.c264.ap-south-1-1.ec2.cloud.redislabs.com:15597

0

2 MB

2

2

mobile

Q

≡

+

Key

Results: 1 key. Scanned 2 / 2 keys

< 1 min

STRINGmobileNo limit48 B

STRINGmobile

48 BLength: 10TTL: No limit< 1 min

2323343445

81

# DEMO

- SET name “selva”
- GET name
- SET age 40
- GET age
- DEL age
- GET age
- EXISTS name
- KEY \*
- flushall

# DEMO CONTD..

- Ttl name
- Expire name 10
- Setex name 10 selva //set string for time period
- Lpush friends kumar //List
- Lrange friends 0 -1
- Rpush friends ram
- Lpop friends
- SADD name "kumar ram" //sets
- SMEMBERS name

# DEMO CONTD..

- HMSET DemoHash name "redis tutorial" // Hashes  
description "redis basic commands for caching" likes 20 visitors 23000
- HGETALL DemoHash
- ZADD tutorials 1 redis // Sorted Sets
- ZADD tutorials 2 mongodb
- ZRANGE tutorials 0 10 WITHSCORES

# PHP CODE TO CONNECT REDIS SERVER

- `<?php`
- `//Connecting to Redis server on localhost`
- `$redis = new Redis();`
- `$redis->connect('127.0.0.1', 6379);`
- `echo "Connection to server sucessfully";`
- `//check whether server is running or not`
- `echo "Server is running: ".$redis->ping();`
- `?>`

# STRING EXAMPLE

- `<?php`
- `//Connecting to Redis server on localhost`
- `$redis = new Redis();`
- `$redis->connect('127.0.0.1', 6379);`
- `echo "Connection to server sucessfully";`
- `//set the data in redis string`
- `$redis->set("tutorial-name", "Redis tutorial");`
- `// Get the stored data and print it`
- `echo "Stored string in redis:: " . $redis->get("tutorial-name");`
- `?>`

# LIST EXAMPLE

- `<?php`
- `$redis = new Redis();`
- `$redis->connect('127.0.0.1', 6379); //Connecting to Redis server on localhost`
- `echo "Connection to server sucessfully";`
- `//store data in redis list`
- `$redis->lpush("tutorial-list", "Redis");`
- `$redis->lpush("tutorial-list", "Mongodb");`
- `$redis->lpush("tutorial-list", "Mysql");`
- `// Get the stored data and print it`
- `$arList = $redis->lrange("tutorial-list", 0 ,5);`
- `echo "Stored string in redis:: ";`
- `print_r($arList);`
- `?>`

# PYTHON EXAMPLE

- `import redis`
- `pool = redis.ConnectionPool(host='localhost', port=6379, db=0)`
- `redis = redis.Redis(connection_pool=pool)`
- `redis.set('mykey', 'Hello from Python!')`
- `value = redis.get('mykey')`
- `print(value)`
- `redis.zadd('vehicles', {'car' : 0})`
- `redis.zadd('vehicles', {'bike' : 0})`
- `vehicles = redis.zrange('vehicles', 0, -1)`
- `print(vehicles)`

# NEO4J DATABASE

- Neo4j is one of the popular Graph Databases and Cypher Query Language (CQL).
- Graph database is a database used to model the data in the form of graph.
- Other Graph Databases are Oracle NoSQL Database, OrientDB, HypherGraphDB, GraphBase, InfiniteGraph, and AllegroGraph.
- Graph databases store relationships and connections as first-class entities.

# RDBMS VS. GRAPH DATABASE

RDBMS	Graph Database
Tables	Graphs
Rows	Nodes
Columns and Data	Properties and its values
Constraints	Relationships
Joins	Traversal

# ADVANTAGES OF NEO4J

- Flexible data model
- Real-time insights
- High availability
- Connected and semi structured data
- Easy retrieval
- Cypher Query Language
- No Joins





## Introducing Neo4j 5: Unbounded Scale, Performance, and Agility | [Learn More](#)

NOVEMBER 16-17, 2022

# NODES 22

Neo4j Online Developer  
Education Summit

Join us for NODES, our free 24-hour virtual graph conference! Learn and engage with the top graph experts and practitioners through 100+ tracks around the globe.

Save my Spot

### Featured Speakers



Nicholas Christakis



Ashleigh Faith



Ward Cunningham

[Products](#)[Solutions](#)[Learn](#)[Developers](#)[Data Scientists](#)[Get Started](#)

## Introducing Neo4j 5: Unbounded Scale, Performance, and Agility

NOVEMBER 16-17, 2022

# NODES 22

Neo4j Online Developer  
Education Summit



**Neo4j AuraDB** [Start Free](#)

Start your fully managed Neo4j cloud database

**Neo4j Sandbox**

Learn and use Neo4j for data science & more

**Neo4j Desktop**

Manage multiple local or remote Neo4j projects

**Neo4j AuraDS** [New!](#)

Fully managed graph data science, starting at \$1/hour



Register a new account to continue



Register

Already have an account? [Log in](#)

OR



Continue with Google



Instances

## Instances

New Instance

Connect

Python

JavaScript

GraphQL

Java

Spring Boot

.NET

Go

Feedback

Instances

Connect

Python

JavaScript

GraphQL

Java

Spring Boot

.NET

Go

Feedback

## Get started by picking a dataset or an empty instance



Free

### Movies

Small example graph of popular movies and actors

BEGINNER



Free

### Graph based Recommendations

Generate personalized real-time recommendations using a dataset of movie reviews



Free

### Graphs for Cybersecurity

Cybersecurity, Active Directory environment auditing and analysis of possible attack paths using graph



Free

### StackOverflow Data

StackOverflow Analysis - Questions, Answers, Tags, Comments



Free

### Empty instance

Load or create your own data in a blank

## Credentials for Instance01

Username: neo4j

Generated password

MJIHKL3HIWK0nCaKafIMReAJVnmMyGk1/



 [Download](#)

We strongly advise changing this initial password.

- ☒ I confirm I have have copied or downloaded the above credentials, as this password will not be available after this point

[Continue](#)



Instances

## Instances

New Instance

Connect

Python

JavaScript

GraphQL

Java

Spring Boot

.NET

Go

Feedback

### Instance01 FREE

● Setting up your instance  
(this will take a few minutes)

Neo4j version 5

Nodes 0 / 200000 (0%)

Relationships 0 / 400000 (0%)

Connection URI -

⚙ Explore

➤ Query

≡ Import





< Back to Instances

Instances

Connect

Python

JavaScript

GraphQL

Java

Spring Boot

.NET

Go

Feedback

## Instance01

Running



Explore

Query

Import

Neo4j version 5 Region Singapore (asia-southeast1), GCP

Nodes 28863 / 200000 (14%)

Relationships 166261 / 400000 (42%)

Connection URI neo4j+s://501c1d18.databases.neo4j.io

Connect

Snapshots

Import Database

Logs

Python

JavaScript

GraphQL

Java

Spring Boot

.NET

Go

```
1 from neo4j import GraphDatabase
2 import logging
3 from neo4j.exceptions import ServiceUnavailable
4
5 class App:
6
```



## Credentials for Instance01

Username: neo4j

Generated password

LA1-q81EvQo7NIL8L7dLV5qUx6RT9Te-\_b



 [Download](#)

We strongly advise changing this initial password.

☐ I confirm I have have copied or downloaded the above credentials, as this password will not be available after this point

Continue

# QUERY

← **auradb/movies**

## What is Cypher?

Cypher is a graph query language that is used to query the Neo4j Database. Just like you use SQL to query a MySQL database, you would use Cypher to query the Neo4j Database.

A simple cypher query can look something like this

```
⌕ Match (m:Movie) where m.released > 2000 RETURN m limit 5
```

Hint: You can click on the query above to populate it in the editor.

**Expected Result:** The above query will return all the movies that were released after the year 2000 limiting the result to 5

neo4j\$

\$ :server connect

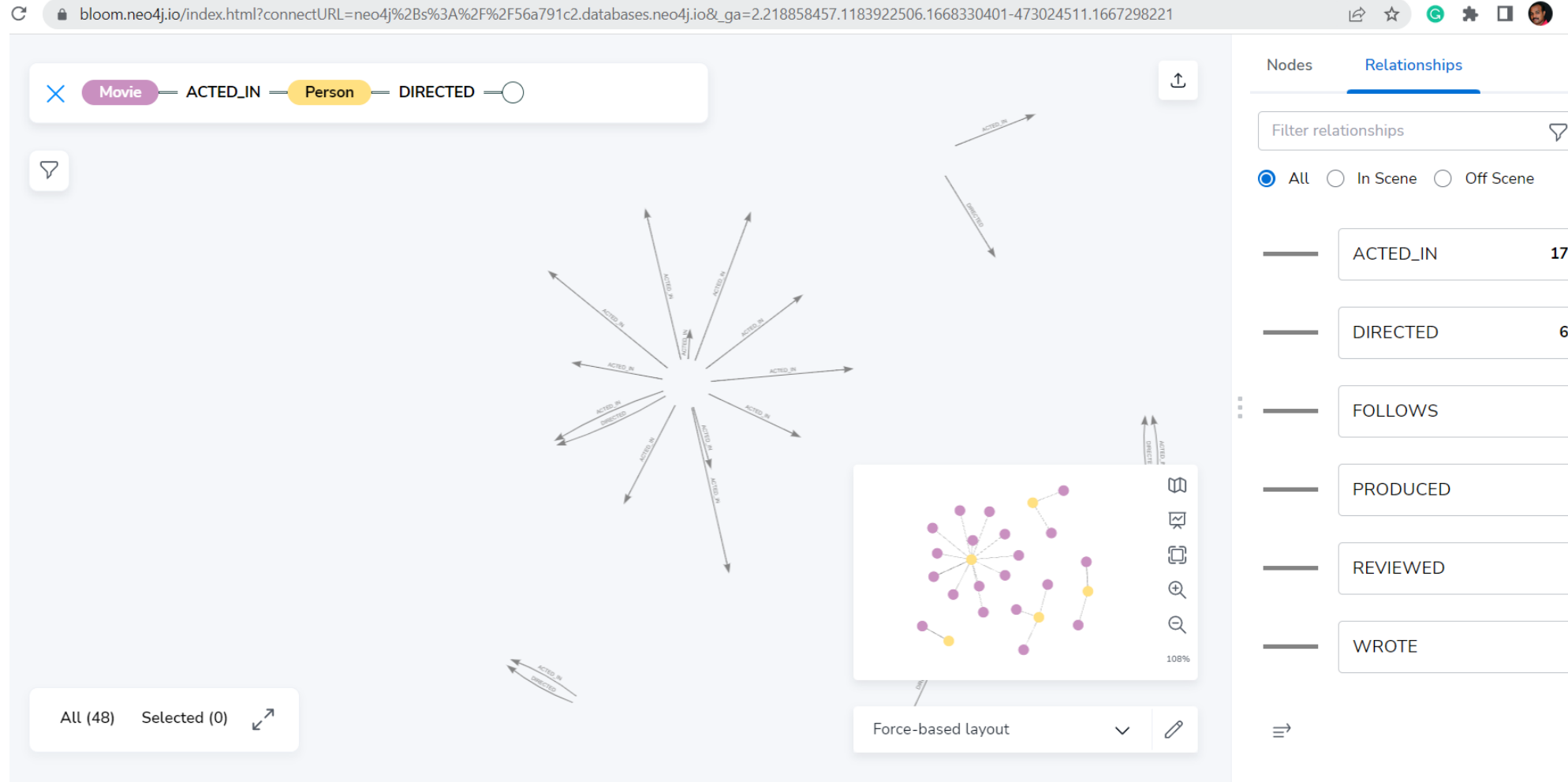
### Connected to Neo4j

Nice to meet you.

You are connected as user **neo4j** to **neo4j+s://56a791c2.databases.neo4j.io:7687**

Connection credentials are not stored in your web browser.

# EXPLORE



# DEMO

- `CREATE (n)`
- `MATCH (n) RETURN n`
- `CREATE (n),(m)`
- `MATCH(n) RETURN n limit 2`
- `MATCH (n) WHERE id(n)=1 RETURN n`
- `MATCH (n) WHERE id(n)<=6 RETURN n`
- `MATCH (n) WHERE id(n) IN [1,2,6] RETURN n`

# DEMO CONTD..

- MATCH (n) WHERE id(n)=1 DELETE n
- MATCH (n) RETURN n
- MATCH (n) WHERE id(n) IN[2,3] DELETE n
- MATCH (n) DELETE n

# DEMO CONTD..

- //WITH LABLE
- CREATE (n:Person)
- MATCH (n) WHERE n:Person RETURN n
- CREATE (n:Person:Indian) // 2 Label
- MATCH (n) WHERE n:Person:Indian RETURN n
- MATCH (n) WHERE n:Person OR n:Indian RETURN n
- MATCH (n) REMOVE n:Person RETURN n
- MATCH (n) WHERE ID(n) IN[2,3] REMOVE n:Employee RETURN n

# DEMO CONTD..

- //Update
- MATCH (n) WHERE ID(n)=0 REMOVE n:manager SET n:Director RETURN n
- //Create Node with property
- CREATE (x:Book{title:NoSQL}) RETURN x;
- CREATE (x:Book{title:"NoSQL",author:"abc",publisher:"wrox"}) RETURN x;
- MATCH (n:Book{author:"abc"}) RETURN n;
- MATCH (n:Book) WHERE n.price <1000 AND (n.author:"abc" OR n.author:"xyz") RETURN n;

# DEMO CONTD..

- //Create Relationship
  - CREATE (Dhawan:player{name: "Shikar Dhawan", YOB: 1985, POB: "Delhi"})
  - CREATE (Ind:Country {name: "India"})
  - CREATE (Dhawan)-[r:BATSMAN\_OF]->(Ind)
  - RETURN Dhawan, Ind
- 
- MATCH (a:player), (b:Country) WHERE a.name = "Shikar Dhawan" AND b.name = "India"
  - CREATE (a)-[r: BATSMAN\_OF]->(b)
  - RETURN a,b

**THANK YOU**