**Introduction to NoSQL Databases:**

Scalability and flexibility is the main issue of modern applications.

Model1: concepts of ACID and BASE consistency models, distributed databases, the CAP theorem, and discuss the challenges of moving from a relational database world to one that utilizes non-relational NoSQL databases.

Model2: introduction to MongoDB databases, CRUD operations, indexing, aggregation framework, replication and sharding, and how to access MongoDB from Python.

Model3: introduction to Apache Cassandra databases, architecture, key features, and data model. Introduction to the Cassandra Query Language and its shell (cqlsh). You will also learn about CQL data types and learn how to perform keyspace, table, and CRUD operations in Cassandra.

Model4: an overview of IBM Cloudant databases, how to use dashboards in Cloudant, use the HTTP API in Cloudant to create and query databases and documents.

**Over View of NoSQL:**

NoSQL stands for not only sql. NoSQL refers to a type of databases that are non-relational in type. Implementation of NoSQL databases are technically deferent from each other’s.

Relational (historical) databases more popular between 1997 and 2000, but since 2000 NoSQL databases have become more popular in the marketplace, due to the scale demands of big data

**Characteristics of NoSQL Databases:**

Four Categories: Key-Value, Document, Column, Graph.

Why to use NoSQL DB?

-Built to scale horizontally (scalability)

-Performance (& scalability)

-Availability (more copies)

-Cloud Architecture

-Cost

-Flexible schema

-Varied Data Structure

-developer use it easily

**NoSQL Database Categories**

**Key-Value:**

-Key-Value databases are the least complex architecturally speaking.

-data is stored with a key and corresponding value blob.

-data represented by a hash-map.

-scale will and sherd easily.

-not use with complex query that connect mutable pieces of data.

Use Cases:

->quick CRUD operations, ex(storing and retrieving session information, storing in-app user profiles, and storing shopping cart data in online stores.)

Unsuitable Use Cases:

->query with data interconnected with many to many relationships, ex (social networks)

->high level consistency (ACID operations), this case need database.

->query based on value not key need Document based database.

**Document (MongoDB):**

-Document-based NoSQL databases use documents to make values visible and able to be queried.

-Each piece of data is considered a document and typically stored in either JSON or XML format.

-Each document offers a flexible schema.

-Content of Doc-DB can be indexed, queried, and do some analytical queries with MapReduce paradigm.

-Allow sharding between multiple nodes.

-Apply atomic operations on single document.

Use Cases:

Event Logging, Online blobs and operational datasets.

Doc-DB designed with internet in mined (Jason, XML, unstructured data)

Unsuitable Use Cases:

-ACID transactions need RDB

-Aggregated-Oriented designed DB (Naturally Normalized Model) need RDB

**Column (Cassandra):**

-Column-based db spawned from the architecture of Google’s Bigtable storage system.

-Column-based databases store data in columns or groups of columns.

-Column ‘families’ are several rows, with unique keys, belonging to one or more columns.

**Graph:**

**MongoDB:**

-MongoDB is a document and a NoSQL database.

-MongoDB supports various data types Documents provide a flexible way of storing data.

-MongoDB documents of a similar type are grouped into collections.

-MongoDB models data as you read/write, brings structured or unstructured data, and provides high availability.

-MongoDB can be used for a variety of purposes because of the flexibility of storing structured or unstructured data.

-MongoDB uses a code-first approach, instead of a design then code approach.

-MongoDB also utilizes an evolving schema.

-Complex data analysis can be done on the server using Aggregation Pipelines.

-MongoDB provides native high availability.

**Use Cases**=> many sources to one source view, IOT, E-Commerce, Real-Time, Finance

**Shell**=> The Mongo shell is an interactive command line use JavaScript interpreter to interact with the databases.

-To use the Mongo shell, you first need to make a connection to cluster via a connection string.

-You use ‘show dbs’ to list databases, ‘use databasename’ to select a database, and ‘show collections’ to list collections in a database.

-CRUD operations consist of Create, Read, Update, and Delete. Useful functions include insertOne, insertMany, findOne, find, count, replace, updateOne,

updateMany, deleteOne, and deleteMany.

**Indexes**

-Indexes help quickly locate data without looking for it everywhere.

-Indexes should be created for the most frequent queries. A compound index indexes more than one field

-MongoDB stores data being indexed on the index entry and a location of the document on disk.

-MongoDB stores an index as a tree to make finding documents more efficient