Chapter:NoSQL

Q1) What is NoSQL? Explain types of NoSQL databases in detail.
Ans:
NoSql:

- NoSQL is literally a combination of two words: No and SQL. The implication is that NoSQL is a technology or product that counters SQL.
- The creators and early adopters of the buzzword *NoSQL* probably wanted to say *No RDBMS* or *No relational* but were infatuated by the nicer sounding NoSQL and stuck to it.
- Whatever the literal meaning, NoSQL is used today as an umbrella term for all databases and data stores that don't follow the popular and well established RDBMS principles and often relate to large data sets accessed and manipulated on a Web scale.
- This means NoSQL is not a single product or even a single technology. It represents a class of products and a collection of diverse, and sometimes related, concepts about data storage and manipulation.

Types Of NoSql:

1. Sorted ordered column-oriented stores
2. Key/value stores
3. Document databases
4. Graph databases

Sorted ordered column-oriented stores:
Google's Bigtable espouses a model where data in stored in a column-oriented way. This contrasts with the row-oriented format in RDBMS. The column-oriented storage allows data to be stored effectively.
It avoids consuming space when storing nulls by simply not storing a column when a value doesn't exist for that column.
Each unit of data can be thought of as a set of key/value pairs, where the unit itself is identified with the help of a primary identifier, often referred to as the primary key.Bigtable and its clones tend to call this primary key the row-key. Also, as the title of this subsection suggests, units are stored in an ordered-sorted manner. The units of data are sorted and ordered on the basis of the row-key.
To explain sorted ordered column-oriented stores, an example serves better than a lot of text, so let me present an example to you. Consider a simple table of values that keeps information about a set of people.
Such a table could have columns like first_name, last_name, occupation, zip_code, and gender. A person's information in this table could be as follows:

first_name: John
last_name: Doe
zip_code: 10001
gender: male

Key/value stores:
A HashMap or an associative array is the simplest data structure that can hold a set of key/value pairs. Such data structures are extremely popular because they provide a very effi cient, big O(1) average algorithm running time for accessing data. The key of a key/value pair is a unique value in the set and can be easily looked up to access the data.

Key/value pairs are of varied types: some keep the data in memory and some provide the capability to persist the data to disk. Key/value pairs can be distributed and held in a cluster of nodes.

A simple, yet powerful, key/value store is Oracle's Berkeley DB. Berkeley DB is a pure storage engine where both key and value are an array of bytes. The core storage engine of Berkeley DB doesn't attach meaning to the key or the value. It takes byte array pairs in and returns the same back to the calling client.

Berkeley DB allows data to be cached in memory and fl ushed to disk as it grows. There is also a notion of indexing the keys for faster lookup and access. Berkeley DB has existed since the mid-1990s. It was created to replace AT&T's NDBM as a part of migrating from BSD 4.3 to 4.4. In 1996, Sleepycat Software was formed to maintain and provide support for Berkeley DB.

Another type of key/value store in common use is a cache. A cache provides an in-memory snapshot of the most-used data in an application. The purpose of cache is to reduce disk I/O. Cache systems could be rudimentary map structures or robust systems with a cache expiration policy. Caching is a popular strategy employed at all levels of a computer software stack to boost performance. Operating systems, databases, middleware components, and applications use caching.

Many of these key/value pairs have APIs that allow get-and-set mechanisms to get and set values. A few, like Redis (http://redis.io/), provide richer abstractions and powerful APIs. Redis could be considered as a data structure server because it provides data structures like string (character sequences), lists, and sets, apart from maps. Also, Redis provides a very rich set of operations to access data from these different types of data structures.

Document databases:

Document databases are not document management systems. More often than not, developers starting out with NoSQL confuse document databases with document and content management systems. The word *document* in *document databases* connotes loosely structured sets of key/value pairs in documents, typically JSON (JavaScript Object Notation), and not *documents* or *spreadsheets* (though these could be stored too).

Document databases treat a document as a whole and avoid splitting a document into its constituent name/value pairs. At a collection level, this allows for putting together a diverse set of documents into a single collection. Document databases allow indexing of documents on the basis of not only its primary identifi er but also its properties. A few different open-source document databases are available today but the most prominent among the available options are MongoDB and CouchDB.

MongoDB

Offi cial Online Resources — www.mongodb.org.
History — Created at 10gen.
Technologies and Language — Implemented in C++.
Access Methods — A JavaScript command-line interface. Drivers exist for a number of languages including C, C#, C++, Erlang. Haskell, Java, JavaScript, Perl, PHP, Python, Ruby, and Scala.
Query Language — SQL-like query language.
Open-Source License — GNU Affero GPL (http://gnu.org/licenses/agpl-3.0.html).
Who Uses It — FourSquare, Shutterfl y, Intuit, Github, and more.

Graph databases:

A graph database focuses on the relationship between data elements. Each element is stored as a node (such as a person in a social media graph). The connections between elements are called links or relationships. In a graph database, connections are first-class elements of the database, stored directly. In relational databases, links are implied, using data to express the relationships.

A graph database is optimized to capture and search the connections between data elements, overcoming the overhead associated with JOINing multiple tables in SQL.

Very few real-world business systems can survive solely on graph queries. As a result graph databases are usually run alongside other more traditional databases.

Use cases include fraud detection, social networks, and knowledge graphs.

As you can see, despite a common umbrella, NoSQL databases are diverse in their data structures and their applications.

Q2)Explain sorted ordered column oriented stores. Also explain HBase and Hypertable.

Ans:

Sorted ordered column-oriented stores:

Google's Bigtable espouses a model where data in stored in a column-oriented way. This contrasts with the row-oriented format in RDBMS. The column-oriented storage allows data to be stored effectively.

It avoids consuming space when storing nulls by simply not storing a column when a value doesn't exist for that column.

Each unit of data can be thought of as a set of key/value pairs, where the unit itself is identified with the help of a primary identifier, often referred to as the primary key.Bigtable and its clones tend to call this primary key the row-key. Also, as the title of this subsection suggests, units are stored in an ordered-sorted manner. The units of data are sorted and ordered on the basis of the row-key.

To explain sorted ordered column-oriented stores, an example serves better than a lot of text, so let me present an example to you. Consider a simple table of values that keeps information about a set of people.

Such a table could have columns like first_name, last_name, occupation, zip_code, and gender. A person's information in this table could be as follows:

first_name: John
last_name: Doe
zip_code: 10001
gender: male

The sorted ordered structure makes data seek by row-key extremely effi cient. Data access is less random and ad-hoc and lookup is as simple as fi nding the node in the sequence that holds the data. Data is inserted at the end of the list. Updates are in-place but often imply adding a newer version of data to the specific cell rather than in-place overwrites. This means a few versions of each cell are maintained at all times. The versioning property is usually configurable.

HBase

Official Online Resources — http://hbase.apache.org.
History — Created at Powerset (now part of Microsoft) in 2007. Donated to the Apache foundation before Powerset was acquired by Microsoft.
Technologies and Language — Implemented in Java.

Access Methods — A JRuby shell allows command-line access to the store. Thrift, Avro,REST, and protobuf clients exist. A few language bindings are also available. A Java API is available with the distribution.
Query Language — No native querying language. Hive (http://hive.apache.org) provides a SQL-like interface for HBase.
Open-Source License — Apache License version 2.
Who Uses It — Facebook, StumbleUpon, Hulu, Ning, Mahalo, Yahoo!, and others.

Hypertable

Offi cial Online Resources — www.hypertable.org.
History — Created at Zvents in 2007. Now an independent open-source project.
Technologies and Language — Implemented in C++, uses Google RE2 regular expression library. RE2 provides a fast and effi cient implementation. Hypertable promises performance boost over HBase, potentially serving to reduce time and cost when dealing with large amounts of data.
Access Methods — A command-line shell is available. In addition, a Thrift interface is supported. Language bindings have been created based on the Thrift interface. A creative developer has even created a JDBC-compliant interface for Hypertable.
Query Language — HQL (Hypertable Query Language) is a SQL-like abstraction for querying Hypertable data. Hypertable also has an adapter for Hive.
Open-Source License — GNU GPL version 2.
Who Uses It — Zvents, Baidu (China's biggest search engine), Rediff (India's biggest portal).

Q3)Explain key/value stores in detail. Also explain cassendra.

Ans:

Key/value stores:

A HashMap or an associative array is the simplest data structure that can hold a set of key/value pairs. Such data structures are extremely popular because they provide a very effi cient, big O(1) average algorithm running time for accessing data. The key of a key/value pair is a unique value in the set and can be easily looked up to access the data. Key/value pairs are of varied types: some keep the data in memory and some provide the capability to persist the data to disk. Key/value pairs can be distributed and held in a cluster of nodes.

A simple, yet powerful, key/value store is Oracle's Berkeley DB. Berkeley DB is a pure storage engine where both key and value are an array of bytes. The core storage engine of Berkeley DB doesn't attach meaning to the key or the value. It takes byte array pairs in and returns the same back to the calling client.

Berkeley DB allows data to be cached in memory and fl ushed to disk as it grows. There is also a notion of indexing the keys for faster lookup and access. Berkeley DB has existed since the mid-1990s. It was created to replace AT&T's NDBM as a part of migrating from BSD 4.3 to 4.4. In 1996, Sleepycat Software was formed to maintain and provide support for Berkeley DB.

Another type of key/value store in common use is a cache. A cache provides an in-memory snapshot of the most-used data in an application. The purpose of cache is to reduce disk I/O. Cache systems could be rudimentary map structures or robust systems with a cache expiration policy. Caching is a popular strategy employed at all levels of a computer software stack to boost performance. Operating systems, databases, middleware components, and applications use caching.

Many of these key/value pairs have APIs that allow get-and-set mechanisms to get and set values. A few, like Redis (http://redis.io/), provide richer abstractions and powerful APIs. Redis could be considered as a data structure server because it provides data structures like string (character sequences), lists, and sets, apart from maps. Also, Redis provides a very rich set of operations to access data from these different types of data structures.

Cassandra

Offi cial Online Resources — http://cassandra.apache.org/.
History — Developed at Facebook and open sourced in 2008, Apache Cassandra was donated to the Apache foundation.
Technologies and Language — Implemented in Java.
Access Methods — A command-line access to the store. Thrift interface and an internal Java API exist. Clients for multiple languages including Java, Python, Grails, PHP, .NET. and Ruby are available. Hadoop integration is also supported.
Query Language — A query language specifi cation is in the making.

Open-Source License — Apache License version 2.
Who Uses It — Facebook, Digg, Reddit, Twitter, and others.

Q4)Explain document databases in detail. Also explain MongoDB and CouchDB.
Ans:
Document databases:

Document databases are not document management systems. More often than not, developers starting out with NoSQL confuse document databases with document and content management systems. The word *document* in *document databases* connotes loosely structured sets of key/value pairs in documents, typically JSON (JavaScript Object Notation), and not *documents* or *spreadsheets* (though these could be stored too).

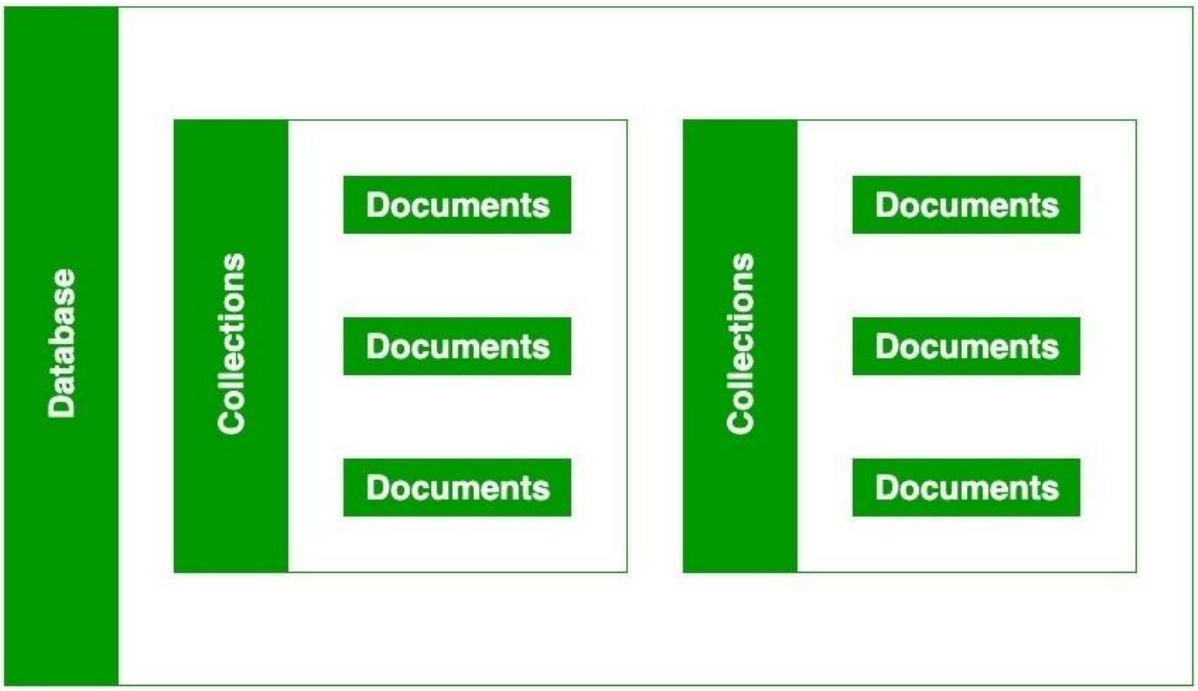
Document databases treat a document as a whole and avoid splitting a document into its constituent name/value pairs. At a collection level, this allows for putting together a diverse set of documents into a single collection. Document databases allow indexing of documents on the basis of not only its primary identifi er but also its properties. A few different open-source document databases are available today but the most prominent among the available options are MongoDB and CouchDB.

MongoDB
Offi cial Online Resources — www.mongodb.org.
History — Created at 10gen.
Technologies and Language — Implemented in C++.
Access Methods — A JavaScript command-line interface. Drivers exist for a number of languages including C, C#, C++, Erlang. Haskell, Java, JavaScript, Perl, PHP, Python, Ruby, and Scala.
Query Language — SQL-like query language.
Open-Source License — GNU Affero GPL (http://gnu.org/licenses/agpl-3.0.html).
Who Uses It — FourSquare, Shutterfl y, Intuit, Github, and more.

CouchDB
Offi cial Online Resources — http://couchdb.apache.org and www.couchbase.com. Most of the authors are part of Couchbase, Inc.
History — Work started in 2005 and it was incubated into Apache in 2008.
Technologies and Language — Implemented in Erlang with some C and a JavaScript execution environment.
Access Methods — Upholds REST above every other mechanism. Use standard web tools and clients to access the database, the same way as you access web resources.
Open-Source License — Apache License version 2.
Who Uses It — Apple, BBC, Canonical, Cern, and more at http://wiki.apache.org/couchdb/CouchDB_in_the_wild .

Q5)What is MongoDB? Explain its features, advantages and disadvantages.
Ans:
MongoDB:
MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009. It also provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, that you can create an application using any of these languages. Nowadays there are so many companies that used MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amount of data.



The MongoDB database contains collections just like the MYSQL database contains tables. You are allowed to create multiple databases and multiple collections.

Now inside of the collection we have documents. These documents contain the data we want to store in the MongoDB database and a single collection can contain multiple documents and you are schema-less means it is not necessary that one document is similar to another.

The documents are created using the fields. Fields are key-value pairs in the documents, it is just like columns in the relation database. The value of the fields can be of any BSON data types like double, string, boolean, etc.

The data stored in the MongoDB is in the format of BSON documents. Here, BSON stands for Binary representation of JSON documents. Or in other words, in the backend, the MongoDB server converts the JSON data into a binary form that is known as BSON and this BSON is stored and queried more efficiently.

In MongoDB documents, you are allowed to store nested data. This nesting of data allows you to create complex relations between data and store them in the same document which makes the working and fetching of data extremely efficient as compared to SQL. In SQL, you need to write complex joins to get the data from table 1 and table 2. The maximum size of the BSON document is 16MB

Features of MongoDB –

Schema-less Database: It is the great feature provided by the MongoDB. A Schema-less database means one collection can hold different types of documents in it. Or in other words, in the MongoDB database, a single collection can hold multiple documents and these documents may consist of the different numbers of fields, content, and size. It is not necessary that the one document is similar to another document like in the relational databases. Due to this cool feature, MongoDB provides great flexibility to databases.

Document Oriented: In MongoDB, all the data stored in the documents instead of tables like in RDBMS. In these documents, the data is stored in fields(key-value pair) instead of rows and columns which make the data much more flexible in comparison to RDBMS. And each document contains its unique object id.

Indexing: In MongoDB database, every field in the documents is indexed with primary and secondary indices this makes easier and takes less time to get or search data from the pool of the data. If the data is not indexed, then database search each document with the specified query which takes lots of time and not so efficient.

Scalability: MongoDB provides horizontal scalability with the help of sharding. Sharding means to distribute data on multiple servers, here a large amount of data is partitioned into data chunks using the shard key, and these data chunks are evenly distributed across shards that reside across many physical servers. It will also add new machines to a running database.

Replication: MongoDB provides high availability and redundancy with the help of replication, it creates multiple copies of the data and sends these copies to a different server so that if one server fails, then the data is retrieved from another server.
Aggregation: It allows to perform operations on the grouped data and get a single result or computed result. It is similar to the SQL GROUPBY clause. It provides three different aggregations i.e, aggregation pipeline, map-reduce function, and single-purpose aggregation methods
High Performance: The performance of MongoDB is very high and data persistence as compared to another database due to its features like scalability, indexing, replication, etc.

Advantages of MongoDB :
It is a schema-less NoSQL database. You need not to design the schema of the database when you are working with MongoDB.
It does not support join operation.
It provides great flexibility to the fields in the documents.
It contains heterogeneous data.
It provides high performance, availability, scalability.
It supports Geospatial efficiently.
It is a document oriented database and the data is stored in BSON documents.
It also supports multiple document ACID transition(string from MongoDB 4.0).
It does not require any SQL injection.
It is easily integrated with Big Data Hadoop

Disadvantages of MongoDB :
It uses high memory for data storage.
You are not allowed to store more than 16MB data in the documents.
The nesting of data in BSON is also limited you are not allowed to nest data more than 100 levels.

Q6) Describe MongoDB. Explain following various commands of MongoDB with syntax and example.

A. Use
B. Insert()
C. Find()
D. Save()
Ans:
For describe mongo DB refer Q5.

A. **Use**:

MongoDB use DATABASE_NAME is used to create database. The command will create a new database if it doesn't exist, otherwise it will return the existing database.
Syntax
Basic syntax of use DATABASE statement is as follows –

use DATABASE_NAME

Example
If you want to use a database with name <mydb>, then use DATABASE statement would be as follows −

>use mydb
switched to db mydb

B. Insert()

The insert command inserts one or more documents and returns a document containing the status of all inserts. The insert methods provided by the MongoDB drivers use this command internally.

Syntax
The command has the following syntax:

```
db.runCommand(
   {
      insert: <collection>,
      documents: [ <document>, <document>, <document>, ... ],
      ordered: <boolean>,
      writeConcern: { <write concern> },
      bypassDocumentValidation: <boolean>,
      comment: <any>
   }
)
```

Example:
Insert a Single Document
Insert a document into the users collection:

```
db.runCommand(
   {
      insert: "users",
      documents: [ { _id: 1, user: "abc123", status: "A" } ]
   }
)
```

C. Find()

find
Executes a query and returns the first batch of results and the cursor id, from which the client can construct a cursor.

Syntax:

```
db.Collection_name.find(selection_criteria, projection)
```

Examples
The following command runs the find command filtering on the rating field and the cuisine field. The command includes a projection to only return the following fields in the matching documents: _id, name, rating, and address fields.

The command sorts the documents in the result set by the name field and limits the result set to 5 documents.

```
db.runCommand(
   {
     find: "restaurants",
     filter: { rating: { $gte: 9 }, cuisine: "italian" },
     projection: { name: 1, rating: 1, address: 1 },
     sort: { name: 1 },
     limit: 5
   }
)
```

D. Save()

db.collection.save()

Updates an existing document or inserts a new document, depending on its document parameter.

Syntax:

```
db.collection.save(
   <document>,
   {
     writeConcern: <document>
   }
)
```

Example:

In the following example, save() method performs an insert since the document passed to the method does not contain the _id field:

```
db.products.save( { item: "book", qty: 40 } )
```

Q7)What is MongoDB? What is collection and document used in MongoDB ?

Consider the following collection

student (rollno, name, marks).

Solve following Queries using MongoDB

1. **Create collection student**
2. **Insert 2 documents in student collection**
3. **Display all documents of student collection.**
4. **Display students having marks greater than 70.**
5. **Display only one document of student collection.**

Ans:

Collections:

A collection is a grouping of MongoDB documents. Documents within a collection can have different fields. A collection is the equivalent of a table in a relational database system. A collection exists within a single database.

Collection Information:
The Collections screen displays the following information for each collection in the selected database:

1. Collection name
2. Number of documents in the collection.
3. Average size of documents in the collection
4. Total size of all documents in the collection
5. Number of indexes on the collection
6. Total size of all indexes on the collection
7. Collation properties for the collection. Hover over a Collation banner to view the properties for that collection.

Documents:
In MongoDB, the data records are stored as BSON documents. Here, BSON stands for binary representation of JSON documents, although BSON contains more data types as compared to JSON. The document is created using field-value pairs or key-value pairs and the value of the field can be of any BSON type.

Syntax:

```
{
field1: value1
field2: value2
....
fieldN: valueN
}
```

Naming restriction of fields:
Before moving further first you should learn about the naming restrictions for fields:
The field names are of strings.
The _id field name is reserved to use as a primary key. And the value of this field must be unique, immutable, and can be of any type other than an array.
The field name cannot contain null characters.
The top-level field names should not start with a dollar sign ($).

Document Size: The maximum size of the BSON document is 16MB. It ensures that the single document does not use too much amount of RAM or bandwidth(during transmission). If a document contains more data than the specified size, then MongoDB provides a GridFS API to store such type of documents.

8. What is CouchDB? Give difference between MongoDB and CouchDB.

Ans:

What is CouchDB?

CouchDB is an open source database developed by Apache software foundation. The focus is on the ease of use, embracing the web. It is a NoSQL document store database.

It uses JSON, to store data (documents), java script as its query language to transform the documents, http protocol for api to access the documents, query the indices with the web browser. It is a multi master application released in 2005 and it became an apache project in 2008.

Why CouchDB?

CouchDB have an HTTP-based REST API, which helps to communicate with the database easily. And the simple structure of HTTP resources and methods (GET, PUT, DELETE) are easy to understand and use.

As we store data in the flexible document-based structure, there is no need to worry about the structure of the data.

Users are provided with powerful data mapping, which allows querying, combining, and filtering the information.

CouchDB provides easy-to-use replication, using which you can copy, share, and synchronize the data between databases and machines.

Data Model:

Database is the outermost data structure/container in CouchDB.

Each database is a collection of independent documents.

Each document maintains its own data and self-contained schema.

Document metadata contains revision information, which makes it possible to merge the differences occurred while the databases were disconnected.

CouchDB implements multi version concurrency control, to avoid the need to lock the database field during writes.

S.NO	CouchDB	MongoDB
1	Datastores in JSON format.	Datastores in BSON format.
2	The database contains documents.	The database contains collections.
3	It favors availability.	It favors consistency.
4	It is written in Erlang.	It is written in C++.
5	It is eventually consistent.	It is strongly consistent.
6	MongoDB is faster than CouchDB.	MongoDB provides faster read speeds.
7	It follows the Map/Reduce query method.	It follows Map/Reduce creating a collection and object-based query language.
8	It uses HTTP/REST-based interface.	It uses a TCP/IP based interface.
9	CouchDB provides support for Mobile Devices. It can run on Apple iOS and Android devices.	MongoDB provides no mobile support.
10	CouchDB offers master-master and master-slave replication.	MongoDB offers master-slave replication.
11	CouchDB is not suitable for a rapidly growing database where the structure is not clearly defined from the beginning.	MongoDB is an apt choice for a rapidly growing database.