**MongoDB:**

**\*How do you connect MongoDb in your application?**

We hav used mongo-java-driver 2.13.3(maven dependancy). We have connect mongodb through propertie file using MongoClientFactory.

//CODE WITH MONGO AUTHENTICATION

**if** (MongoClientFactory.*mongoClient* == **null**) {

Properties configs = **new** Properties();

configs = SystemUtil.*loadItegrationConfigPropertyFile*();

String user = configs.getProperty(ConfigUtil.***MONGO\_DATABASE\_USER\_NAME***);

String database = configs.getProperty(ConfigUtil.***MONGO\_DATABASE***);

**char**[] password = configs.getProperty(ConfigUtil.***MONGO\_DATABASE\_USER\_PASSWORD***).toCharArray();

MongoCredential credential = MongoCredential.*createCredential*(user, database, password);

MongoClientOptions options = MongoClientOptions.*builder*().minHeartbeatFrequency(20000).readPreference(ReadPreference.*primary*()).build();

**int** mongoPort= Integer.*parseInt*(configs.getProperty(ConfigUtil.***MONGOCLIENT\_PORT***));

MongoClientFactory.*mongoClient* = **new** MongoClient(Arrays.*asList*(**new** ServerAddress(configs.getProperty(ConfigUtil.***MONGOCLIENT\_HOST\_1***), mongoPort),**new** ServerAddress(configs.getProperty(ConfigUtil.***MONGOCLIENT\_HOST\_2***), mongoPort),**new** ServerAddress(configs.getProperty(ConfigUtil.***MONGOCLIENT\_HOST\_3***), mongoPort)),

Arrays.*asList*(credential), options);

}

**return** MongoClientFactory.*mongoClient*;

**\*What is latency of db?**

**Latency** is the time it takes to send a TCP packet between the app and the SQL Server. You incur **latency** on the way up to the **DB** and on the way down. People generally talk about **latency** in terms of round trip times: i.e. the time to get there and back.

**\*Can we create foreign key in mongo db?**

Normally, **MongoDB** doesn't work with the concept of **foreign keys**. The **foreign keys can** be used in the Relational Databases for visualizing data from multiple collections simultaneously

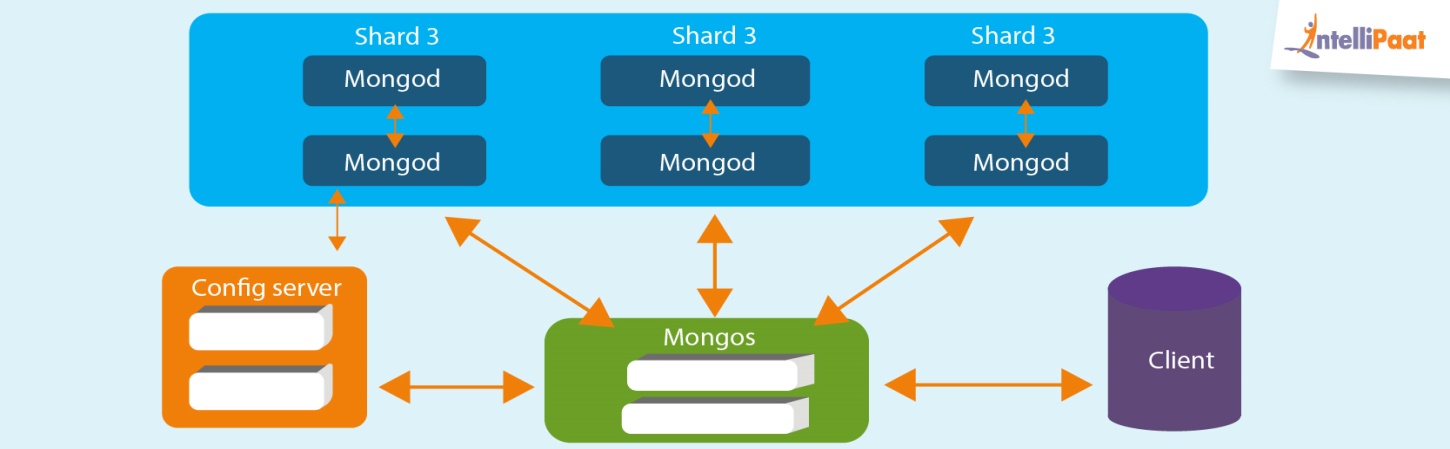
**\*Which language is used by mongo db?**

**MongoDB** is an Open Source database written in C++. Drivers and client libraries are typically written in their respective **languages**, although some drivers **use** C extensions for better performance.

**\*Architecture of mongo db?**

Being a **NoSQL tool** means that it does not use the usual rows and columns that you so much associate with the relational database management. **It is an architecture that is built on collections and documents. The basic unit of data in this database consists of a set of key–value pairs.**It allows documents to have different fields and structures. **This database uses a document storage format called BSON which is a binary style of**[**JSON**](https://intellipaat.com/blog/processing-json-data-in-real-time-streaming-using-storm-kafka/)**documents.**The data model that MongoDB follows is a highly elastic one that lets you combine and store data of multivariate types without having to compromise on the powerful indexing options, data access, and validation rules. There is no downtime when you want to dynamically modify the schemas.

**\*Architecture of MongoDB NoSQL Database**



**Database:** In simple words, it can be called the physical container for data. Each of the databases has its own set of files on the file system with multiple databases existing on a single MongoDB server.

**Collection:** A group of database documents can be called a collection. The RDBMS equivalent to a collection is a table. The entire collection exists within a single database. There are no schemas when it comes to collections. Inside the collection, various documents can have varied fields, but mostly the documents within a collection are meant for the same purpose or for serving the same end goal.

**Document:** A set of key–value pairs can be designated as a document. Documents are associated with dynamic schemas. The benefit of having dynamic schemas is that a document in a single collection does not have to possess the same structure or fields. Also, the common fields in a collection’s document can have varied types of data.

**\*Aggregation in mongo db?**

Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides three ways to perform aggregation: the [aggregation pipeline](https://docs.mongodb.com/manual/aggregation/#std-label-aggregation-framework)($match,$group), the [map-reduce function](https://docs.mongodb.com/manual/aggregation/#std-label-aggregation-map-reduce)($group,$merge), and [single purpose aggregation methods](https://docs.mongodb.com/manual/aggregation/#std-label-single-purpose-agg-operations)($count,$distinct).

**\*What is object\_id in mongo db?**

In **MongoDB**, \_id field as the primary key for the collection so that each document can be uniquely identified in the collection. The \_id field contains a unique **ObjectID** value. When explicitly creating an **id** field, it needs to be created with \_id in its name.

The 12-byte **ObjectId** value consists of: a 4-byte timestamp value, representing the **ObjectId's** creation, measured in seconds since the Unix epoch. a 5-byte random value. a 3-byte incrementing counter, initialized to a random value.

**\*Documents are stored in which form in mongo db?**

In MongoDB, data is stored as documents. These documents are stored in MongoDB in JSON (**JavaScript Object Notation**) format. JSON documents support embedded fields, so related data and lists of data can be stored with the document instead of an external table.

**\*Which indexes are used in mongo db?**

* **Single** Field.
* Compound Index.
* Multikey Index.
* Geospatial Index.
* Text Indexes.
* Hashed Indexes.

**\*Can we create views in mongo?**

**Yes.**

**Views** act as read-only collections, and are computed on demand during read operations. **You** must **create views** in the same database as the source collection. **MongoDB** executes read operations on **views** as part of the underlying aggregation pipeline. ... The name of the source collection or **view** from which to **create** the **view**.

**\*What is namespace in mongo?**

**MongoDB** stores BSON (Binary Interchange and Structure Object Notation) objects in the collection. The concatenation of the collection name and database name is called a **namespace**.

**\*How to define cache size in mongo?**

**MongoDB**, in its default configuration, will use the larger of either 256 MB or ½ of (ram – 1 GB) for its **cache size**. You can **limit** the **MongoDB cache size** by adding the cacheSizeGB argument to the /etc/mongod.

**\*Why you used mongo db in your application?**

Companies and development teams of all sizes **use MongoDB** because: The document data model is a powerful way to store and retrieve data that allows developers to move fast. **MongoDB's** horizontal, scale-out architecture can support huge volumes of both data and traffic.

**MongoDB** is a document-oriented database which stores data in JSON-like documents with dynamic schema. It means you can store your records without worrying about the data structure such as the number of fields or types of fields to store values.

**MongoDB** stores data in JSON-like documents that **can** vary in structure, offering a dynamic, flexible schema. **MongoDB** was also designed for high availability and scalability, with built-in replication and auto-sharding. **MongoDB is** a tool in the Databases category of a tech stack.

**\*Why we use MongoDB instead of MySQL?**

**MongoDB** is schema-free, allowing **you** to create documents without having to define the structure of the document first. These documents can be easily changed by adding or deleting fields. ... In some cases, **MongoDB** performance is improved over **MySQL** because **MongoDB** does not **use** joins to connect data, improving performance.

### \*What is sharding in MongoDB?

In MongoDB, Sharding is a procedure of storing data records across multiple machines. It is a MongoDB approach to meet the demands of data growth. It creates horizontal partition of data in a database or search engine. Each partition is referred as shard or database shard.

### \*What is replica set in MongoDB?

A replica can be specified as a group of mongo instances that host the same data set. In a replica set, one node is primary, and another is secondary. All data is replicated from primary to secondary nodes.

### \*What is primary and secondary replica set in MongoDB?

In MongoDB, primary nodes are the node that can accept write. These are also known as master nodes. The replication in MongoDB is single master so, only one node can accept write operations at a time.

Secondary nodes are known as slave nodes. These are read only nodes that replicate from the primary.

### \*By default, which replica sets are used to write data?

By default, MongoDB writes data only to the primary replica set.

### \*How does MongoDB provide concurrency?

MongoDB uses reader-writer locks for concurrency. Reader-writer locks allow concurrent readers shared access to a resource, such as a database or collection, but give exclusive access to a single write operation.