Table of Contents

Introduction

The purpose of building this repository is to understand the concepts of database, and to practice the skills of using database. There are some goals wanted to achieve in this repository, with handling databases in several programming languages being the major one.

To develop a large project, we may use multiple databases to process and store information. These database are run by different programming languages. Observing queries and operations of databases have similar syntax and features. We can group the syntax and features into a set of common functions. For example, a SQL database must have commands to store, insert, update, and delete data. The common functions help us to understand and use databases quickly.

Catalog

MongoDB

PostgreDB

MySQL

Introduction

MongoDB is a popular open-source `document-oriented database system`. It is designed to store and manage `unstructured data` in the form of documents, which are similar to `JSON objects`. MongoDB is known for its flexible data model, horizontal scalability, and ease of use.

MongoDB VS SQL Database

Advantages of MongoDB

1. Data Model: MongoDB is a document-oriented database, which means that data is stored in a more flexible and dynamic way compared to traditional SQL databases, which are based on a rigid table structure. In MongoDB, data is stored in collections of documents, which are similar to JSON objects and can have varying structures. In contrast, SQL databases use tables with a `fixed schema`, where each row has the `same set of columns`.
2. Query Language: MongoDB uses a query language called MongoDB Query Language `(MQL)`, which is based on `JSON` and allows for more flexible and expressive queries. SQL databases, on the other hand, use SQL (Structured Query Language), which is more rigid and requires a predefined schema.
3. Scalability: MongoDB is designed to `scale horizontally`, which means that it can easily handle large datasets by distributing them across multiple servers. Traditional SQL databases, on the other hand, are designed to scale vertically, which means that they rely on adding more resources to a single server.
4. Performance: MongoDB can provide `faster performance` for certain types of queries, especially when dealing with `large datasets`. This is because it can use `indexing and other optimization techniques` to speed up queries. In contrast, traditional SQL databases may experience performance issues when dealing with complex queries or large datasets.

Disadvantages of MongoDB

1. Transactions: MongoDB supports transactions, but with some limitations. Traditional SQL databases, on the other hand, have strong support for transactions and provide advanced features suchas ACID (Atomicity, Consistency, Isolation, and Durability) compliance, which ensures that transactions are executed reliably and with data integrity.
2. Data Relationships: MongoDB supports data relationships between documents using a feature called document embedding or referencing. This allows for more flexible and efficient data modeling compared to traditional SQL databases, which use foreign keys to establish relationships between tables.

Conclusion

In my points of view, MongoDB is a good choice for big data analysis and data mining due to its flexibility and easy migration. However, it is not a good choice for transactional data due to its lack of ACID compliance. e.g. databse of Salvio Environmental Monitor.

ACID Compliance

- Atomicity: This property ensures that a transaction is treated as a single, indivisible unit of work. It means that either all the operations in a transaction are completed successfully, or none of them are completed at all. If any part of a transaction fails, the entire transaction is rolled back to its original state.
- Consistency: This property ensures that a transaction brings the database from one valid state to another. It means that a transaction should not violate any integrity constraints or business rules defined by the database schema. If a transaction violates any constraints or rules, it is rolled back to its original state.
- Isolation: This property ensures that multiple transactions can execute concurrently without interfering with each other. It means that each transaction should be executed in isolation from other transactions, and the results of a transaction should not be visible to other transactions until it is committed.
- Durability: This property ensures that the results of a transaction are permanent and survive system failures. It means that once a transaction is committed, its changes are saved to non-volatile storage, such as a hard disk or SSD, and will still be available even if the system crashes.

Catalog

1. Install MongoDB on Linux

- 1.2 Follow Instructions below this link
- 1.2 References: https://linuxize.com/post/how-to-install-mongodb-on-debian-10/
- 1.4 Input this command to activate command prompt of MongoDB

```
ubuntu:~$ mongo
```

- 1.3 Create Database Admin

```
db.createUser({
    user: "user1",
    pwd: "user1password",
    roles: [
              { role: "userAdmin", db: "sampledb" },
              { role: "dbAdmin",   db: "sampledb" },
              { role: "readWrite", db: "sampledb" }
            ]
});
```

- 1.4 Create SuperUser for specific Database

```
db.createUser({
    user: "user1",
    pwd: "user1password",
    roles: [
              { role: "userAdminAnyDatabase", db: "admin" },
              { role: "readWriteAnyDatabase", db: "admin" },
              { role: "dbAdminAnyDatabase",   db: "admin" }
            ]
});
```

- 1.5 Create User for specific Database ``` db.createUser({ user: 'username', pwd: 'password', roles: [{role: 'readWrite', db: 'safio'}] });

```
## 2. Enable MongoDB Auth
- 2.1 Modify Configuration of Database
```

sudo vim /etc/mongod.conf

```
- 2.2 Change authorization to 'enabled'
```

security: authorization: 'enabled'

```
- 2.3 Enable all Ip address to access the database
```

network interfaces

net: port: 27017 bindIp: 0.0.0.0 #default value is 127.0.0.1

```
- 2.4 Restart and Reset Severice
```

sudo service mongod restart

```
- 2.5 Test Connection
```

to access the admin database

ubuntu:~$ mongo -u admin -p myadminpassword 127.0.0.1/admin

to access the other databases

ubuntu:~$ mongo -u user1 -p user1password 127.0.0.1/sampledb ```

3. Open up network port on the EC2 instance

- Different Cloud platforms may have different configurations
- Please set up your configurations depending on your platforms

4. Mongo Compass Connections

- mongodb://username:password@ip_address:port

References:

- Install MongoDB on Debian:

- https://linuxize.com/post/how-to-install-mongodb-on-debian-10/
- Create account and network connection
- https://medium.com/founding-ithaka/setting-up-and-connecting-to-a-remote-mongodb-database-5df754a4da89
- MongoDB Open network tutorial
- https://docs.mongodb.com/manual/reference/connection-string/

Commands

1. MongoDB Indexing

```
db.stores.insertMany(
   [
     { _id: 1, name: "Java Hut", description: "Coffee and cakes" },
     { _id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },
     { _id: 3, name: "Coffee Shop", description: "Just coffee" },
     { _id: 4, name: "Clothes Clothes Clothes", description: "Discount clothing" },
     { _id: 5, name: "Java Shopping", description: "Indonesian goods" }
   ]
)
db.stores.createIndex( { name: "text", description: "text" } ) //both in text
db.stores.find({ $text: {$search: "Coffee" } })
db.stores.find({ $text: {$search: "Java Hut Coffee" } })
db.stores.find(
   { $text: { $search: "java hut coffee" } },
   { score: { $meta: "textScore" } }
).sort( { score: { $meta: "textScore" } } )
```

2. Aggregation

```
db.purchase_orders.insertMany(
     [
          {product: "toothbrush", total: 4.75, customer: "Mike"},
          {product: "guitar", total: 199.99, customer: "Tom"},
          {product: "milk", total: 11.33, customer: "Mike"},
          {product: "pizza", total: 8.50, customer: "Karen"},
          {product: "toothbrush", total: 4.75, customer: "Karen"},
          {product: "pizza", total: 4.75, customer: "Dave"}
          {product: "toothbrush", total: 4.75, customer: "Mike"},
     ]
)
// find out how many toothbrushes were sold
db.purchase_orders.count({product: "toothbrush"})
// Find list of all products sold
db.purchase_orders.distinct("product")
// Find the total amount of money spent by each customer
db.purchase_orders.aggregate(
     [
          {$match: {} },
          {$group: {_id: "$customer", total: { $sum: "$total"} } }
     ]
)
// Find how much has been spent on each product and sort it by price
db.purchase_orders.aggregate(
     [
          {$match: {} },
          {$group: {_id: "$product", total: { $sum: "$total"} } },
          {$sort: {total: -1}}
     ]
)
// Find how much money each customer has spent on toothbrushes and pizza
db.purchase_orders.aggregate(
     [
          {$match: {product: {$in: ["toothbrush", "pizza"]} } },
          {$group: {_id: "$product", total: { $sum: "$total"} } },
     ]
)
// https://docs.mongodb.com/manual/reference/operator/aggregation/
// Show the list of all pipeline operators
```

3. MongoDB bulkWrite

```
db.students.bulkWrite(
      [
         { insertOne :
            {
               "document" :
               {
                  name: "Andrew", major: "Architecture", gpa: 3.2
               }
            }
         },
         { insertOne :
            {
               "document" :
               {
                  name: "Terry", major: "Math", gpa: 3.8
               }
            }
         },
         { updateOne :
            {
               filter : { name : "Terry" },
               update : { $set : { gpa : 4.0 } }
            }
         },
         { deleteOne :
            { filter : { name : "Kate"} }
         },
         { replaceOne :
            {
               filter : { name : "Claire" },
               replacement : { name: "Genny", major: "Counsling", gpa: 2.4 }
            }
         }
    ],
    {ordered: false}
   );
```

Problems

Unable to start MongoDB services

Descrpition

- The file ownership is changed to root or admin. The current permission isn't high enough to enable the service

Solutions

- 1. deleting the mongod sock file

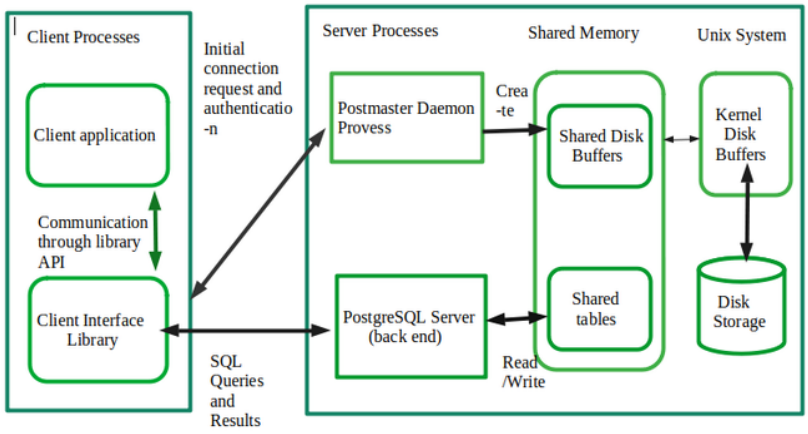
```
sudo rm -rf /tmp/mongodb-27017.sock
sudo systemctl start mongod
```

- 1. Change file ownership

```
chown -R mongodb:mongodb /var/lib/mongodb
chown mongodb:mongodb /tmp/mongodb-27017.sock
sudo service mongod restart
```

A. Architectural Fundamentals

1. PostgreSQL uses user and client model. The whole model can be explained with the graph below



2. Client Process:
    - Client application: The interface that allows user control / send database command. I.e. Pgadmin, windows shell
    - Client Interface Library: Transfer the commands from application to SQL commands
3. Sever Prcess:
    -
4. Internet connection via TCP/IP
5. PostgreSQL can handel multiple connections from clients which start ("forks") in server. The details of "folk" can be explained by C language

SQL to Go command

1. Create a Table

2. Insert a Row or Rows to Table

3.