

Contents

- Server In Memory Storage VS Disk Storage
- Connecting Spring Boot App Server with Database Server
- Request Classes and Validation
- JDBC

Server In Memory Storage VS Disk Storage : The term "memory" usually means RAM (Random Access Memory); RAM is hardware that allows the computer to efficiently perform more than one task at a time (i.e., multi-task).

The terms "disk space" and "storage" usually refer to hard drive storage. Hard drive storage is typically used for long-term storage of various types of files. Higher capacity hard drives can store larger amounts and sizes of files, such as videos, music, pictures, and documents.

Challenge 1 : Why storing data in `HashMap<Integer,Person>` is not scalable for companies like Amazon , Microsoft ? OR Why do we need Persistent Storage for our applications ?

HINT : <https://www.geeksforgeeks.org/need-for-dbms/>

All servers will be communicating with a single source of truth where data won't be lost on stopping the server .

Think , what is RAM memory and Disk memory in your computer ?

HINT : This shows that RAM memory of the server is very less and thus not feasible for data storage of your system .

Challenge 2 : What is the con of using a database ?

HINT : Speed as when we used `HashMap` for storage , we were fetching from the main memory of your server which is very fast when compared to network calls from IP1 to IP2 .

But still we use databases for storage and not server main memory because in the latter case , Correctness will be compromised for optimisation which is not desirable .

To overcome slowness in case of disk (database) , we can use cache .

References :

<https://aws.amazon.com/caching/database-caching/#:~:text=A%20database%20cache%20supplements%20your,or%20as%20a%20standalone%20layer.>

<https://stackoverflow.com/questions/12227752/what-is-a-database-cache-and-how-does-one-use-it>

Connecting Spring Boot App Server with Database Server : The responsibility of connecting and writing data to the database server lies with the application server .

DDL - Data definition language (index, alter, create a new table)

DML - Data manipulation language (insert into, update, delete)

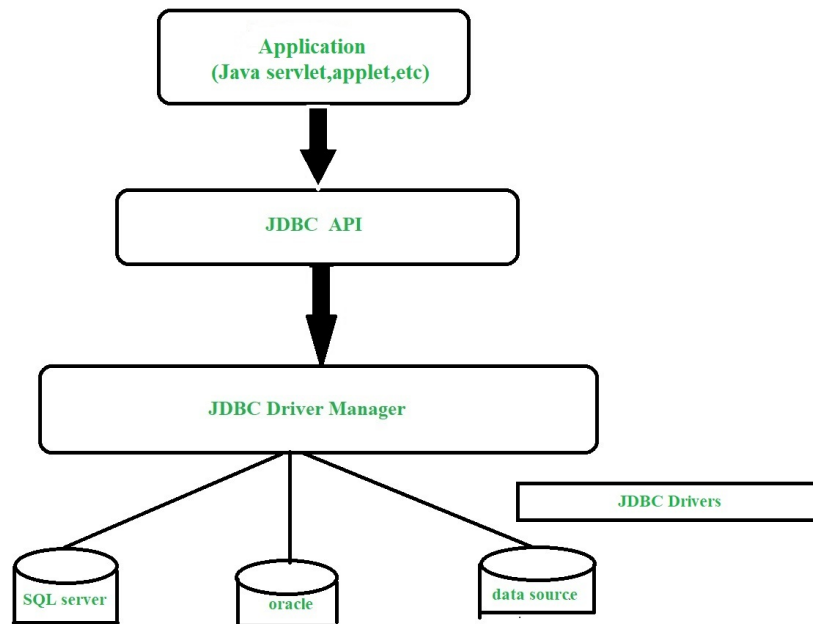
CRUD - Create, Read, Update and Delete

Challenge 1 : What is JDBC ?

HINT : JDBC (Java Database Connectivity) is a standard Java interface for connecting from Java to relational databases. The JDBC standard was defined by Sun

Microsystems, allowing individual providers to implement and extend the standard with their own JDBC drivers. <https://stackoverflow.com/questions/2716733/how-does-jdbc-work>

<https://www.geeksforgeeks.org/establishing-jdbc-connection-in-java/>



Challenge 2 : What are types of Statements in JDBC ?

<https://www.geeksforgeeks.org/types-of-statements-in-jdbc/>

<https://www.geeksforgeeks.org/how-to-use-preparedstatement-in-java/>

Challenge 3 : What is the difference between execute, executeQuery and executeUpdate in JDBC ?

HINT : <https://www.javapedia.net/JDBC/1793>

We will use .execute() for creating the table and it will be successful if it returns false .

We will use .executeQuery() for select query .

We will use .executeUpdate() for insert query .

Challenge 4 : Let us create a project to showcase database connection to Spring Boot app now . This will be a spring boot application which implements the crud functionality for a User management system . We can create , update , read and delete users . Data of users needs to be persisted to a MySQL database using JDBC protocol only .

HINT : We will be following the Spring MVC framework .

<https://www.geeksforgeeks.org/spring-mvc-framework/>

<1>Create a Spring Boot Maven project using start.spring.io . We will be creating CRUD API and persisting our data to a MySQL database using JDBC protocol . Which dependencies will we need ?

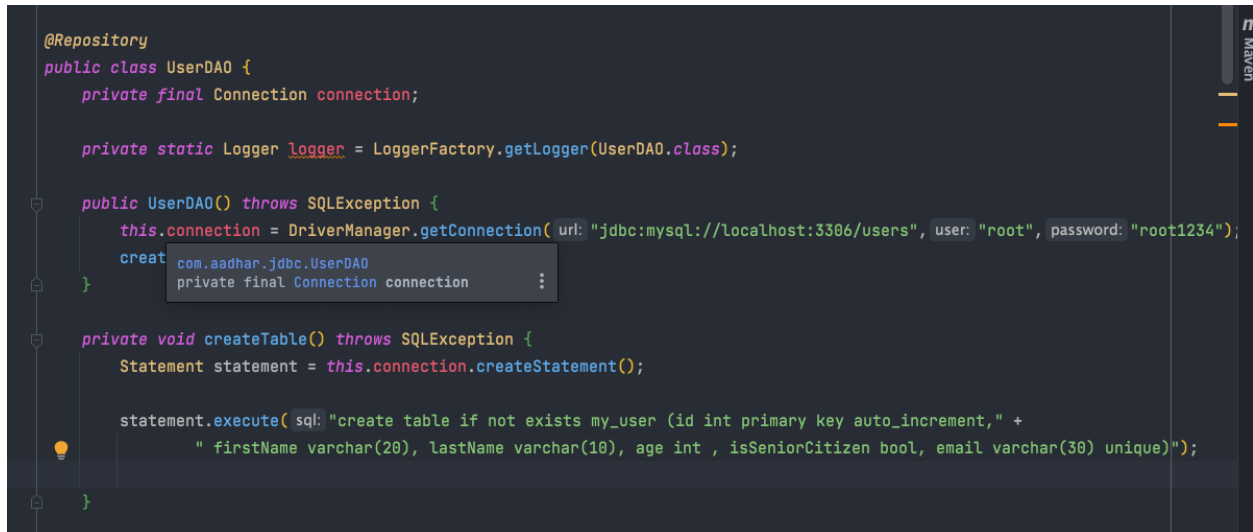
Note : Don't forget to add the MySQL Connector Java and Spring Boot Starter Validation maven dependency to POM.xml .This is the driver we need to establish the connection between our Spring Boot application and the MySQL database .

<2> Now we need to have mysql installed in our system . Post installation , create a new database called users .

<3>Now we will create our model class (entity class) ie User.java with following properties . Here how can we use lombok to get constructors and getter setter methods .

```
public class User {  
    private int id;  
    private String firstName;  
    private String lastName;  
    private boolean isSeniorCitizen;  
    private int age;  
    private String email;  
}
```

<4> Now we will work on UserDao.java . We will have a connection object of Connection class as a property . In the constructor , firstly we will get the connection from the DriverManager class and store it in our connection object property . Secondly , Then in DAO class , we will create a createTable method with logic to create our table named "my_user". We call this method from the DAO constructor .



```
@Repository
public class UserDao {
    private final Connection connection;

    private static Logger logger = LoggerFactory.getLogger(UserDao.class);

    public UserDao() throws SQLException {
        this.connection = DriverManager.getConnection( url: "jdbc:mysql://localhost:3306/users", user: "root", password: "root1234");
        createTable();
    }

    private void createTable() throws SQLException {
        Statement statement = this.connection.createStatement();

        statement.execute(sql: "create table if not exists my_user (id int primary key auto_increment," +
            " firstName varchar(20), lastName varchar(10), age int , isSeniorCitizen bool, email varchar(30) unique)");
    }
}
```

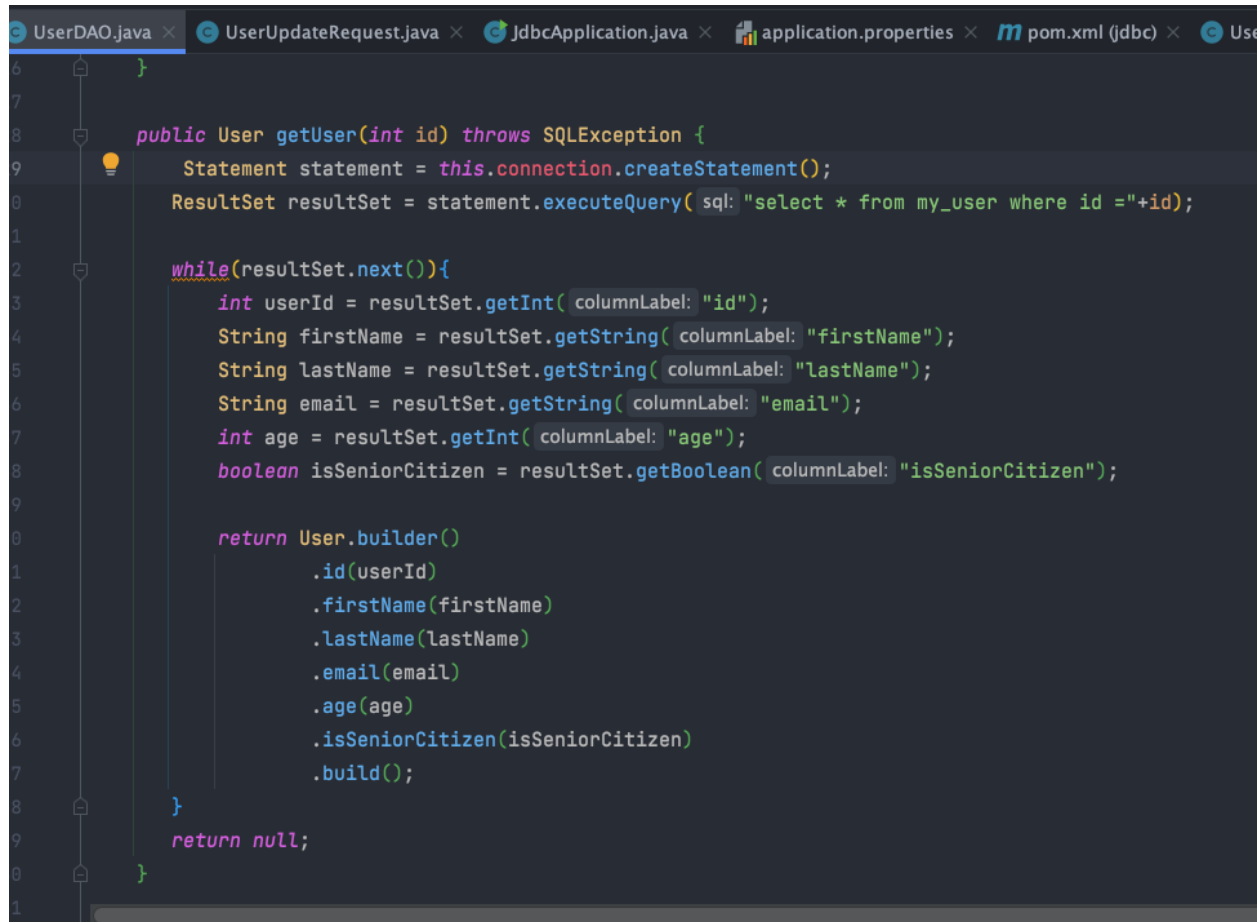
<5> Now your task is to create API in UserController , write the business logic for each API in UserService.java and finally communicate with database using the UserDao.java Class .

Following API need to be developed :

- create user (use UserCreateRequest object with Validations as JSON RequestBody object instead of User object)
- get all users
- get user by id (we will have id as primary key.We will not be taking id as input in json requestbody object because it is primary key and will be auto incremented .)
- put (update a user . HINT : Here use UserUpdateRequest object with Validations as JSON RequestBody object instead of User object and make all properties in UserUpdateRequest.java with datatype of Wrapper class instead of using primitive data types)
- delete a user by id

Below are some examples of UserDao methods :

/user/{id} GET



```
6 }
7
8 public User getUser(int id) throws SQLException {
9     Statement statement = this.connection.createStatement();
10    ResultSet resultSet = statement.executeQuery( sql: "select * from my_user where id ="+id);
11
12    while(resultSet.next()){
13        int userId = resultSet.getInt( columnLabel: "id");
14        String firstName = resultSet.getString( columnLabel: "firstName");
15        String lastName = resultSet.getString( columnLabel: "lastName");
16        String email = resultSet.getString( columnLabel: "email");
17        int age = resultSet.getInt( columnLabel: "age");
18        boolean isSeniorCitizen = resultSet.getBoolean( columnLabel: "isSeniorCitizen");
19
20        return User.builder()
21            .id(userId)
22            .firstName(firstName)
23            .lastName(lastName)
24            .email(email)
25            .age(age)
26            .isSeniorCitizen(isSeniorCitizen)
27            .build();
28    }
29    return null;
30 }
```

/user POST

```

public void create(User user) throws SQLException {
    PreparedStatement preparedStatement = this.connection.prepareStatement( sql: "insert into my_user " +
        "(firstName, lastName, age, email, isSeniorCitizen) VALUES(?,?,?,?,?)" );
    preparedStatement.setString( parameterIndex: 1, user.getFirstName());
    preparedStatement.setString( parameterIndex: 2, user.getLastName());
    preparedStatement.setInt( parameterIndex: 3, user.getAge());
    preparedStatement.setString( parameterIndex: 4, user.getEmail());
    preparedStatement.setBoolean( parameterIndex: 5, user.isSeniorCitizen());

    int result = preparedStatement.executeUpdate();

    logger.info("No of records inserted = "+result);
}

```

/user/{id} PUT

<https://ide.geeksforgeeks.org/S9MQjQpYHZ>

Challenge 5 : What is DDL vs DML ? <https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/>

Challenge 6 : Will we use the JDBC protocol to connect to NoSQL databases also ?
HINT : No . They have their different driver dependencies .

Challenge 7 : We have a DriverManager class inside rt.jar which is not part of MySQL connector java driver dependency . Then why do we need the driver ?

HINT : The implementation of Statement.java is provided in the driver . Statement.java is just an interface .

Challenge 8 : Why can't we use our model class User during creating a new user . Why did we have to create a new class UserCreateRequest.java and accept it as JSON ?

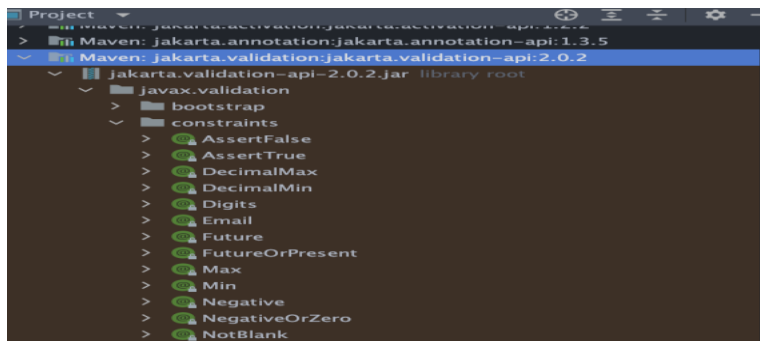
```

3
4
5      @PostMapping("/user")
6      public void createUser(@Valid @RequestBody UserCreateRequest userCreateRequest) throws S
7          //TODO: Creating a new user with given details
8          userService.createUser(userCreateRequest);
9      }
10

```

HINT : For applying Validations . Also not all properties of the model class need to be sent as JSON from Postman to create a new User .

You can find all the available validations in the validation dependency jar inside the External Libraries.



Challenge 9 : Can we not apply the validations directly on the model class User.java ?
HINT : Yes , we can but it's not a recommended approach as it will be saved in the database .

Challenge 10 : While creating a new user , suppose we are sending json from postman which has mistakes according to the validations put on the UserCreateRequest class properties .Will the PostMapping work fine ? Will it only show the first validation mistake or all mistakes ?

HINT : No , in the logs you would be able to see all the mistakes that are failing validation .

Challenge 11 : Checkout the below image and tell if the id for the second row Raj will be 2 or 34 ?

Also what will be id for the next row if I don't provide any id in the insert statement (will it be 35 or 2 or 3) ?

```
mysql> create table test(id int primary key auto_increment, name varchar(30));
Query OK, 0 rows affected (0.06 sec)

mysql> insert into test(name) VALUES ('Raj');
Query OK, 1 row affected (0.02 sec)

mysql> select * from test;
+-----+-----+
| id | name |
+-----+-----+
| 1 | Raj |
+-----+-----+
1 row in set (0.00 sec)

mysql> insert into test(id, name) VALUES (34, 'Raj');
```

HINT : Next id will always be GREATEST ID VALUE SO FAR + 1

Challenge 12 : Can NULL be stored in (i) Primary Key (ii) Foreign Key ?

Challenge 13 : What will happen if we remove the Auto_Increment from the java code while creating the User table . Will it throw an error when the table gets created or the first row gets inserted or the second row inserted ?

HINT : It will not throw an error due to null . Actually there will be no condition where null is getting stored in the Primary Key . Error will come due to duplicate values. Think !

Challenge 14 : What is the difference between Statement and Prepared Statement ?

HINT : <https://www.baeldung.com/java-statement-preparedstatement>

Challenge 15 : What are the benefits of using Prepared Statement ?

HINT : <http://javarevisited.blogspot.com/2012/03/why-use-preparedstatement-in-java-jdbc.html>

Challenge 16 : What's the difference between boolean and Boolean in Java?

HINT : <https://stackoverflow.com/questions/1295170/whats-the-difference-between-boolean-and-boolean-in-java>

Challenge 17 : At which MVC architecture layer do we convert the UserCreateRequest object to User class object ?

HINT : The layer at which we write our business logic i.e. Service Layer .

Challenge 18 : Let's go to the post api where we are creating new users . Here while giving the JSON object from postman , this time let's also have the id property too and give value as 1 .What will happen . If there is an error , what will be the status code?

HINT : No error , this id we provide from json will not affect anything and the id with which our object be stored in our db will be the autoincrement of previous id . Status code 200 Successful .

Challenge 19 :**(HomeWork)** Suppose in the previous case we are taking the json object of User class and not UserCreateRequest . What will be the outcome if we have 2 rows in our table with id 1 and 2 . This time , give id property as 1 . What will happen ?

Challenge 20 : What is the architecture of JDBC ? What are cons of JDBC ?

HINT : <https://www.geeksforgeeks.org/introduction-to-jdbc/>

Challenge 21 : What is ResultSet and how to retrieve data from ResultSet ?

HINT : <https://www.javatpoint.com/ResultSet-interface>

```
List<User> userList = new ArrayList<>();

Statement statement = this.connection.createStatement();
ResultSet resultSet = statement.executeQuery( sql: "select * from my_user");

while(resultSet.next()){
//    int id = resultSet.getInt(1);

    int id = resultSet.getInt( columnLabel: "id");
    String firstName = resultSet.getString( columnLabel: "firstName");
    String lastName = resultSet.getString( columnLabel: "lastName");
    String email = resultSet.getString( columnLabel: "email");
    int age = resultSet.getInt( columnLabel: "age");
    boolean isSeniorCitizen = resultSet.getBoolean( columnLabel: "isSeniorCitizen");

    User user = User.builder()
        .id(id)
        .firstName(firstName)
        .lastName(lastName)
        .email(email)
        .age(age)
        .isSeniorCitizen(isSeniorCitizen)
        .build();
    userList.add(user);
}
```

Challenge 22 : What is Hibernate (ORM) , JPA (contract) , JDBC (protocol) ?

HINT : JPA is a contract to communicate with Database which is implemented by Hibernate . Internally Hibernate also uses JDBC protocol but with Hibernate , we don't have to write any boiler plate code and don't have to ourselves communicate with the driver . Hibernate does it for us like a piece of cake . <https://hackernoon.com/the-difference-between-jdbc-jpa-hibernate-and-spring-data-jpa>

Challenge 23 : What will be the response status code if we try to fetch a user by id which does not exist ?

HINT : 200 Ok . But the response will be simply null . Same if you try to delete a user by id which does not exist .

References :

Indexing in SQL <https://www.geeksforgeeks.org/sql-indexes/>

<https://dzone.com/articles/database-btree-indexing-in-sqlite>

<https://www.geeksforgeeks.org/performing-database-operations-java-sql-create-insert-update-delete-select/> IMP

<https://howtodoinjava.com/java/jdbc/how-to-execute-preparedstatement-using-jdbc/> IMP

<https://www.geeksforgeeks.org/java-jpa-vs-hibernate/> IMP

<https://www.geeksforgeeks.org/difference-between-jdbc-and-hibernate-in-java/> IMP

<https://www.geeksforgeeks.org/spring-mvc-using-java-based-configuration/>

HOMEWORK PRACTICE :

Q1. Given a CSV file with User data in each row . Your task is to parse it and store each row from CSV into a row in my_user Table in your database .

```
// 1 csv file
```

```
/*
```

```
    Name,Age,Email
```

```
    Lin,12,lin@google.com
```

```
*/
```

Q2. If you had MongoDB instead of MySQL , at which layer would you make the changes ? Your task is to create a Spring Boot application with a similar use case as in Challenge 4 but this time with MongoDB as database .

