How to make java application communicate with Permanent storage systems like
      a. File(java.io.*)
            a. FileInputStream
            b. FileOutputStream
            c. ObjectInputStream
            d. ObjectOutputStream
            e. FileReader
            f. FileWriter
            g. BufferedReader
            h. BufferedWriter
            i. PrintWriter
---------------apache community api(To copy content from one file to another file)---------------
                  eg:: IOutils.copy(src,dest);

      b. database
              java ------> API ----> for developers

               java   ----> Pgm language[Architecture of JDK software]
                              JDBC-API(Application Programming Interface]
               database ----> SQL language[Architecture of SQL Software]

MySQL is installed local in our system. After deployment any webapps how our data(MySQL) goes on internet to use for clients,
and how the client access his data.can you give short summary for about that.
Is our data go on the internet. You have bulid STUDENT CRUD APP if we want to deploy how any user can access data.
How to things will work in real life project.
I always thought about that and I search on YouTube but I didn't answer.

MySQL ---> locally means (jdbc url :: jdbc:mysql://localhost:3306:databaseName[actual db information present in cloud])
 Project build for webapp will be deployed as "war" file in the machine configured with cloud details

      [configured in devops environment]server(ipadress) -----> DNS(domain name server) -----> unique name

++++++++
JDBC API
++++++++
 => To communicate with database, we need an api from java community and that api is "JDBC API".
 => SUNMS gave an SRS for all db vendors to implement the specification requirement for db requirement in java language
 => SUNMS gave the interfaces in the package called "java.sql.* and javax.sql.*"
 => These interfaces will be used by java developers,where as the db vendors implementation code will be binded at the runtime
    depending upon the type of database we use in the project.

java developer =====> interfaces ===dbspecific====> mysql
java developer =====> interfaces ===dbspecific====> oracle
java developer =====> interfaces ===dbspecific====> postgresql

      writing code using interfaces(java.sql.*,javax.sql.*) =======>(any database as per project specification)
                  WO   =======> RA(any database)

WORA(Write Once RunAnyWhere)

List commonly used interfaces used in JDBC API(Application program interfaces)?
API -> It consists of set of classes/interfaces through which the coding would be made simple.
a. Connection
b. Statement,PreparedStatement,CallableStatement
c. ResultSet
d. DataSource
e. RowSet

Steps followed to write java communicate with database
a. Load and register the driver
b. Establish the connection
c. Create Statement/PreparedStatement/CallableStatement to send the query
d. Execute the query based on the nature of query(DDL,DML,TCL)
e. Process the result based on the output from Database
f. Close the connection
g. Handle exceptions.

Spring
a. SpringCore(expert)
b. SpringDataJPA
c. SpringMVC
d. SpringAOP

---

d. SpringRest
e. SpringCloud
f. Microservices

JRE -----> will be loaded only with .class files
=> We need to infrom JVM that hey set up the environment to communicate with database

Frameworks :: Springcloud and microservices
c. cloud
d. git repositories

In java langauge data can be stored in multiple forms
a. stack(local variable)
b. heap(object data/instance variable)
c. methodarea(class data/static variable)
All these memory location would be on RAM(volatile memory)

Q>
if we have multiple methods in DaoImpl , in every method we call both the methods? is it good practice or there is any better approach? Sir please answer

```
class DaoImpl
{
      Connection connection = getConnection();
      pubic void save(){
            //connection
      }
      public void read(){
            //connection
```

```
      }
      public void update(){
            //connection
      }
      public void display(){
            //connection
      }
}
we took 12 class each of 3 hours long to digest these.

++++
FAQ>
++++
 1. Differnce b/w Statement vs PreparedStatement
 2. When to use PreparedStatement and When to use CallableStatement
 3. Difference b/w Physical connection and Logical Connection
 4. What is DataSource and how to implement in JDBC
 5. Working with Date and time insertion
 6. Working with images(LOB's operation)
 7. Working with types of ResultSet
 8. Explain SQLInjection.
 9. Difference b/w execute(),executeUpdate(),executeQuery()?

            boolean execute()        -> for both select and non-select operation
                                    a. true (select)
                                    b. false(non-select)
            int executeUpdate()      -> for execute non-select query
            ReslutSet executeQuery() -> for execute select query

 10. What are the limitations of JDBC and what is the alternative to jdbc?

++++++++++++++++++++++++++++++
ORM(Object Relational Mapping)
++++++++++++++++++++++++++++++
 JPA => specification vendor
      implementation vendors are
            a. hibernate(commonly used orm tool)
            b. eclipselink
            c. ibatis
            d. openjpa
            e. datajpa(uses hibernate internally and reduces lot of boiler code)

++++++++++++
Hibernate FAQ
++++++++++++
 1. Explain the interfaces associated with hibernate?
 2. Explain the features of hibernate?
 3. Explain the caching in hibernate?
 4. Explain the type of operations supported by hibernate?
      a. SRO[save(),saveOrUpdate(),get(),load(),delete()]
      b. Bulkoperations[JPQL---> HQL]
 5. Difference b/w get() and load()?
 6. How to execute NativeQuery in Hibernate?
 7. Explain locking mechanism in hibernate?
 8. How to perform date and time insertion in hibernate?
 10.How to perform Lob's operation in hibernate?
 11. Explain the different types of algorithm to generate primary key for the
table[generators]
 12. Explain the relationship in hibernate?
```

13. Connection pooling in hibernate.
14. How to execute StoredProcedure in hibernate.
15. Pagination in hibernate[important in webapplications]
16. Annotations in hibernate[Practise only using annotations]
17. Versioning and TimeStamping in hiberante
18. Hibernate communicating with multiple databases