Terminologies

What is Persistence?
The process of Storing and managing the data for a long time is called "Persistency".
To Perform Persistencey we use secondary devices like HDD,CD,DVD,ThumbDrivers etc
From the application perspective,we store the information inside variables/objects,but these objects would vanish once the application stops its execution.
To achieve persistency we use
1. File Operations(Storing it is HDD)
   java.io.*
2. Database(Storing it in the form of Table)
   java.sql.*

Terminologies associated with Persistency
==========
1. Persistence store
   It is a place/store where data will be saved and managed for a long time.
   eg: Files,DB s/w(MySQL,Oracle,PostgreSQL,.....)
2. Persistence data
   The data of Persistence Store is called 'Persistence Data'.
   eg: File info,DB tables and their records.
3. Persistence operation
   Insert,update,delete,select there are the operations which are performed on "Persistence Data".

4. Persistence Logic
   The logic which is written to perform Persistence operation is called as "Persistence Logic".
   eg: IO Streams logic(Serialization,DeSerialization)
       JDBC code(Technology name is JDBCAPI)
       hibernate code
       Spring JDBC
       Spring ORM(hibernate)
       Spring DataJPA(hot cake)

5. Pesistence technology/Framework
   The technology/Framework through which we write persistence logic is called 'PersistenceTechnology/Framework'.
   eg: JDBC(Technology)
       Hibernate(Framework/tool)
       SpringJDBC/SpringORM/SpringDataJpa(framework)

When we already have JDBC Techology, What is the need to go for framework/tool called ORM?

Limitations of JDBC
---
1. If we use JDBC to develop persistence logic,we need to write sql queries by following the syntax of "Database".
   DBQueries are specific to Database,this makes JDBC not portable across mulitple databases.
2. JDBC technology if we use and write a code, there would be a boiler plate code in our application.
   Boiler plate code => A code which would repeat in multiple parts of the project with no change/small change is called

boiler plate code.

CRUD
=====

1. Load and register the driver(automatic from JDBC4.X)
2. Establish the connection
3. Create PrepareStatement
4. Execute the Query
5. Process the ResultSet
6. Handle the Exception
7. Closing the Resource

Step1,2,3,6,7 boiler plat code becoz it is a common logic.

3. JDBC technology throws only one Exception called "SQLException',but it is a CheckedException which means u should have
handling logic otherwise code would not compile.
a. try{}catch(SQLException e){}
b. public static void main(String... args) throws SQLException{}

4. JDBC technology has only Exception class called "SQLException",so we don't have detailed hierarchy of Exceptions related
to different problems.

5. JDBC ResultSet object is not serializable, so we can't send it over the network, we need to use Bean/POJO to send the data
over the network by writing our own logic.

6. While closing the jdbc connection object, we need to analyze the code allot otherwise it would result in
"NullPointerException".
eg: Connection con = DriverManager.getConnection(url,user,password)
if(con!=null){.....}
closing the connection object should take place in "finally" block only.
To make the usage of AutoCloseable, we need to know the syntax of "try with Resource".

7. Java ====> OOP's based language
Assume we need to send Student object to database, can we write a logic of Database query at Object level if we use JDBC?
No, Not possible becoz DBqueries always expectes the value,but not the object directly.

8. JDBC doen't have good support of Transaction Management
a. local
b. global(no support in JDBC)

9. JDBC supports only positional parameters,it is difficult for the user to inject the values,It doesnot support namedparamaters.
String sqlInsertQuery = "insert into student(`name`,`email`,`city`,`country`) values(?,?,?,?)";
String sqlInsertQuery = "insert into student(`name`,`email`,`city`,`country`) values(:name,:email,:city,:country)";

10. To use JDBC, Strong knowledge of SQL is required.

11. JDBC does not supports versioning ,timestamp as inbuilt features
versioning:: keeping track of how many times record got modified.
timestamp:: keep track of when record was inserted and when lastly it was modified.

12. While developing persitence logic using JDBC, we can't enjoy oops features like
        a. inheritance
        b. polymorphism
        c. composition
    because jdbc does not allow objects as input values in sqlqueries.

                Solution to all these problems is use "ORM".

ORM:- (ORM stands for): Object Relational Mapping.
It is a theory concept used at database programming to perform operations like insert. Update, delete and select in object
format only ie. JDBC converts object to primitive data and SQL Query shuld be written by programmer using primitives,
which is not following OOPs.
        ORM says “Do not convert object data, do operations in OOPs. Format only”.

For this concept programmer should follow mapping rule. Given as
    1. className- Must be mapped with – tableName
    2. VariableName- Must be mapped with – columnName
            ** should be done by programmer using XML/Annotations concept.
            ** Then ORM convert Object ⇄ ROW
            ** Here , ORM only generates SQL Query

What is Framework?
Initially when java was introduced, it has only java api to develop standalone applications.
later group of api's are released under then name jee for developing distrubuted applications.
Developers faced so many problems while creating projects using java and jee api's.
jee is a largest set of api's, it was difficult for developers to remember so many classes and interfaces.
Devleoper needs to write so many boiler plate code to do integration of api's.
To overcome these problem framework was introduced by "thirdparty" vendors.
A framework provides "framework-api" which is an abstraction on top of "java and jee" api's.
Framework is not a new technology, it is an abstraction which is built on top of technology.
With frameworks we have the following facility
    a. Developer burden will be reduced
    b. Project can be deliverd to the client easily
    c. Project maintainence would be easy.

How many types of framework are available?
   There are 2 types of framework
        a. invasive framework
                => Developer has to extend his class from a superclass or interface supplied by the framework-api.
                => The developer class would be tightly coupled, so that class can't be moved to new framework
                        for execution.
                            eg: Servlet, EJB's,Struts

        b. non-invasive framework
                => Developer need not extend his class from a superclass or interface supplied by the framework-api.
                => The developer class would be loosely coupled, so

that the class can be moved to new framework
for execution.
eg: Hibernate(ORM tool) and Spring.