**=========================================**

**-------------Core Java Related Questions --------------**

**=========================================**

1. **What is Java?**
2. **What is JDK, JVM and JRE?**

**JDK is used by developers to create Java applications.**

**JVM is a runtime environment that executes Java bytecode.**

**JRE provides the runtime environment for Java applications without development tools.**

1. **Difference between Static or Global**
2. **Difference between private, public, protected?**
3. **Difference between final, finally and finalize?**
4. **Types of data Type uses and what’s the purpose?**
5. **What is mutable or immutable?**
6. **String is immutable or not?**
7. **Difference between String, String builder or String buffer?**

**StringBuffer (Mean gape) so it’s thread safe once it realese then other proceed.**

**StringBuilder(Mean building steps) not thread safe many thread use it’s once.**

1. **How may type of method?**
2. **Ternary operators in java?**
3. **How can we handle null type exception in java?**
4. **Types of loops in java like for, while, do-while? Foreach?**
5. **How many types of exception?**
6. **What is run-time exception or compile-time exception?**
7. **How can we handle exception? And how many ways to handle exception?**
8. **Is Java purely Object-oriented Programming?**
9. **Type of Collection framework in java?**
10. **What is array?**
11. **What is list?**
12. **Difference between array or Array List in java?**
13. **In which version of java, you are working? And what’s features?**
    1. **Java 8**
       1. **Lamda expression**
       2. **Optional**
       3. **Steams api**
       4. **DateApi**
14. **Difference between Hash-Table or Set?**
15. **Difference between HashMap or Set?**
16. **Internal working of HashMap?**
17. **What is OOP?**
18. **Difference between abstract class or interface?**
19. **Where we use abstract class and where we use interface?**
20. **Difference between encapsulation and abstraction?**
21. **What is polymorphism in java?**
22. **What is tight coupling or lose coupling in java?**
23. **How can we achieve diamond problem in java?**
24. **How can we stop inheritance? E.g. I have a class human I want no one can inherit this human class so what can I do?**
25. **What are some important features of java8?**
26. **Functional interfaces in java?**
27. **Difference between Collection framework or Streams Api?**
28. **What’s the benefit of stream Api?**
29. **How many types of Design pattern in java?**
30. **Singelton Design pattern in java?**
31. **Threads in java**
32. **Multithreading in java?**
33. **Difference between thread.run or thread.start ?**
34. **How can we implement Logging?**
35. **Wrapper classes?**
36. **Difference between hasphmap or hashset?**
37. **What’s the concept of serialization or deserialization in java?**
38. **Transient and volatile keyword why we use? And where we use?**
39. **java 8 map & flat map  / map & tomap?**
40. **out of memeory handling in java 8**
41. **All objects in java inherited from?**
42. **What are the two ways to create a thread?**
43. **What are the different types of garbage collectors in Java?**
44. **What is a finally block? Is there a case when finally will execute?**
45. **What are the differences between processes and threads?**
46. **join/fork/future in java?**
47. **peak in arrays?**
48. **repitions in string?**
49. **Cascading & its types?**
50. **Method overloading and overriding in java in detail?**
51. **Method Hiding?**
52. **Predicate in java?**
53. **Static Block in java**
54. **Thread Pool in java**
55. **Which part will create java .byte file ? JDK,JVM or JRE**
56. **Where we use primitive and where we use non primitive**
57. **How we make over method thread safe**
58. **Finally block**
59. **Where we use jar file or where we use war file**
60. **Class Loader**
61. **Which type of functional interface you are using in stream?**

**=========================================**

**-------------Experience --------------**

**=========================================**

* **Explain the concept of immutability in Java. Provide examples of immutable classes in Java standard library.**

* **What are the differences between checked and unchecked exceptions in Java? Can you provide examples of each?**

* **Explain the concept of Java generics and how they improve type safety. Provide examples of using generics in collections.**

* **What is the difference between ArrayList and LinkedList? In what scenarios would you prefer one over the other?**

* **How does Java handle memory management? Explain the roles of the stack and heap in memory allocation.**

* **What is the difference between final, finally, and finalize in Java?**

* **Explain the concept of Java annotations. Provide examples of built-in annotations and explain how custom annotations can be created.**

* **What is polymorphism in Java? Explain the difference between compile-time polymorphism and runtime polymorphism.**

* **What is the volatile keyword in Java? When and why would you use it?**

* **Explain the Java synchronization mechanisms. Compare synchronized blocks and java.util.concurrent locks.**

* **What is the Java Memory Model (JMM)? How does it ensure thread safety in Java?**

* **Explain the difference between String, StringBuffer, and StringBuilder in Java. When would you use each of them?**

* **What are lambda expressions in Java 8? How do they simplify code and what are the best practices for using them?**

* **What are functional interfaces in Java? Provide examples and explain their significance in the context of lambda expressions.**

* **Explain the principles of Object-Oriented Programming (OOP) in Java. How do encapsulation, inheritance, and polymorphism apply in real-world scenarios?**

* **What are the different types of class loaders in Java? How do they work and what is their significance?**

* **Explain the concept of serialization and deserialization in Java. How can objects be serialized and deserialized?**

* **What is reflection in Java? Provide examples of how reflection can be used to inspect and modify classes and objects at runtime.**

* **What are the advantages and disadvantages of using an interface versus an abstract class in Java? When would you prefer one over the other?**

* **Explain the concept of Java 8 Streams API. How does it facilitate functional-style operations on collections?**

**=========================================**

**-------------Multithreading --------------**

**=========================================**

1. **What is multithreading?**
2. **What is the difference between a process and a thread?**
3. **What are the advantages of using multithreading?**
4. **What are race conditions and how can they be avoided?**
5. **What is synchronization in multithreading?**
6. **What are mutexes and semaphores?**
7. **What is deadlock and how can it be prevented?**
8. **What is thread pooling?**
9. **What is thread safety?**
10. **What are some common multithreading issues and how can they be debugged?**
11. **Implement a thread-safe singleton class using double-checked locking. Explain why double-checked locking is necessary and how it ensures thread safety.**
12. **Write a program to demonstrate the "producer-consumer" problem using multithreading. Implement it using low-level synchronization constructs like wait() and notify().**
13. **Create a deadlock scenario involving multiple threads and demonstrate how it can be detected and resolved programmatically.**
14. **Implement a custom thread pool with a fixed number of threads and a blocking queue for task management. Ensure proper synchronization and handling of tasks.**
15. **Write a program to calculate the factorial of a number using multiple threads, where each thread calculates a portion of the factorial. Ensure efficient synchronization and coordination between threads.**
16. **Develop a program to simulate a dining philosophers problem with deadlock prevention using techniques like resource allocation hierarchy or timeouts.**
17. **Implement a concurrent version of a hash map in Java. Ensure thread safety while maintaining performance and scalability.**
18. **Write a program to demonstrate thread starvation and propose solutions to mitigate it.**
19. **Create a program that performs matrix multiplication using multithreading. Optimize the program for performance while ensuring correct results and minimal contention.**
20. **Develop a program to implement a read-write lock, allowing multiple readers or a single writer to access a shared resource concurrently. Ensure fairness and proper synchronization.**

**If two user update a table at the same time in an api how can we validate ? like two peoples do a transaction at the same time.**

**=========================================**

**-------------Web Services--------------**

**=========================================**

1. What is Webservices?

* webservices are type of internet software that use standardized messaging protocol
* user can access some webservice from peer-to-peer arrangement rather than by central server.
* some services can communicate with other services.

1. Two types of Webservices.

* 1-SOAP webservice
* 2-Rest full webservice

1. How webservice works?

* webservice are build using standardized protocol in order to integrate various application
* XML (Extensible Markup language) This is used for tag, code and decode.
* SOAP (Simple Object Access Protocol) This is used to transfer data. SOAP protocol was developed so different programming language could communicate with minimal effort.
* WSDL (Webservices Description Language) This is used to telling the client application what is include in the webservice and how to connect.
* UDDI (Universal Description, Discovery Integration) This is used to list what webservice are available with in one application. it allows webservice to discoverable to other services.
* Rest (Representation State Transfer protocol) while not all webservice use the Rest protocol. Rest Api are more lightful manageable and saleable.

1. which language is used in web services?

* You can use language according to your stack like, java, JavaScript, phyton, c#

1. What is RESTFUL API?

* Restful Api is an interface that two-computer system use to exchange information securely over the internet. most business application have to communicate with other
* internal and third-party application to perform various task.

1. Embedded Server for Tomcat?
   * Tomcat
   * Jetty
   * Undertow

1. how can we use central tomcat server?

                             You can use central tomcat

* First add in a pom tomcat dependency
* Change packaging JAR to WAR
* Place this war file in to install tomcat WEBAPPS folder and restart it it will run java application

1. What is servlet?

                             servlet is basically java base Api that handle various things.

* HTTP Handling
* Extensibility
* State Management
* Request-Response management life cycle.

1. What's the role of servlet in spring boot?

* Servlet contain an important role in spring boot, embedded servlet container (Like tomcat, jetty, undertow etc) support like a servlet. that work like a servlet to host.
* In our Spring boot application controller act like a servlet that can take request manipulate data and return a response.

1. What are microservices, and what are the key advantages and challenges associated with implementing a microservices architecture?

* Microservices is an architectural style that structures an application as a collection of small, loosely coupled, and independently deployable services. Each service is designed to perform a specific business function and communicates with others through well-defined APIs

1. How do Microservices communicate with each other?

* Microservices communicate through well-defined APIs. This communication can happen via HTTP/REST, messaging protocols like RabbitMQ or Kafka, or other mechanisms. The choice depends on the specific needs of the application.

1. Explain the concept of API Gateway in Microservices?

* An API Gateway is a service that acts as a single-entry point for managing and routing requests to various microservices. It handles tasks such as authentication, load balancing, and routing, providing a unified interface for clients to interact with multiple services

**=========================================**

**----------  SPRING CLOUD ----------**

**==========================================**

1. What is Spring cloud?

* Spring cloud is a set of tools that can used by developer to build some common patterns in distributed system and microservices such as service discovery, configuration management, routing, circuit breaker pattern etc.

1. What are the common features of Spring Cloud?

* Service registry and discovery
* Routing
* Service to Service call
* Versioned configuration
* Load balancing

1. Difference between service discovery and service registry?

* Service Discovery

Service discovery is the process of automatically detecting and locating service with in the network. It allows service to find and

Communicate with each other dynamically without the need of

Hardcode configuration.

* Service Registry

A service registry is a centralized directory or database that keep track the information of about available service in their network location such as their IP address, port number.

1. What is load balancing?

* Load balancing is a technique used to improve the distribution of workload

Across the several computing resource, such as computer cluster CPU, network lines and disk drive.in microservice architecture just not for api gateway patter but you can use authorization, authentication load balancing or security.

1. How load balancing is implemented in spring cloud?

* In spring cloud, you can use Netflix Ribbon. Which is responsible for distributing the incoming request among available resources of microservice.

Ribbon use round robin algorithm but it also supports other load balancing algo. Such as random and weight loaded.

1. What is the use of Spring cloud bus?

* The spring cloud bus provides a helpful feature to refresh configuration across multiple instances.

1. Name the service that provide service registry and provided?

* Eureka
* Zookeper

1. Difference between spring cloud or spring boot?

* Spring boot is a framework that help us to develop stand alone, production grade spring boot application on the other hand spring cloud is a framework that help us to develop distributed microservice base application.

1. What are some common spring cloud annotations?

* @EnableEurekaServer
* @EnableConfigServer
* @EnableServerDiscovery
* @EnableCircuitBroker
* @HystricCommand
  1. What is the combination of Spring boot 3 annotation

**=========================================**

**----------  SPRING BOOT ----------**

**==========================================**

1. dependency injection
2. Spring IOC
3. Bean Factory
4. Spring Application context
5. Spring maven/gradle
6. Spring Annotations
7. PA vs Crud repo
8. CRUD OR JPA
9. Spring security
10. native queries
11. hibernate query language
12. Join mapping in hibernate
13. transient
14. @Lob/@Basic(fetch = FetchType.LAZY)/@joincolumn
15. Spingboot path variable
16. swagger
17. lazy vs eager loading
18. OOP -> multiple inheritance
19. OOP-> Create parent object with reference of child
20. <https://qxf2.com/blog/mysql-query-execution/>

**=========================================**

**----------  APACHE KAFKA ----------**

**==========================================**

1. What is Kafka?

Kafka is a distributed message streaming platform that is used to produce and subscribe mechanism to stream the record. Stream means flow and record mean data so that's mean dataflow and it's an open source.

1. What is distributed system?

* In earlier we have a centralized message system. but when this centralized system crashes all data is loss. so overcome this demerit. they introduce a distributed system.
* Distributed system mean that make the multiple copies of entity and that we called replication. if we lose one entity remanning help us to restore the data. but it cons is that it used a lot of space. and it is configurable. it makes loosely couple. it reduces the connection.

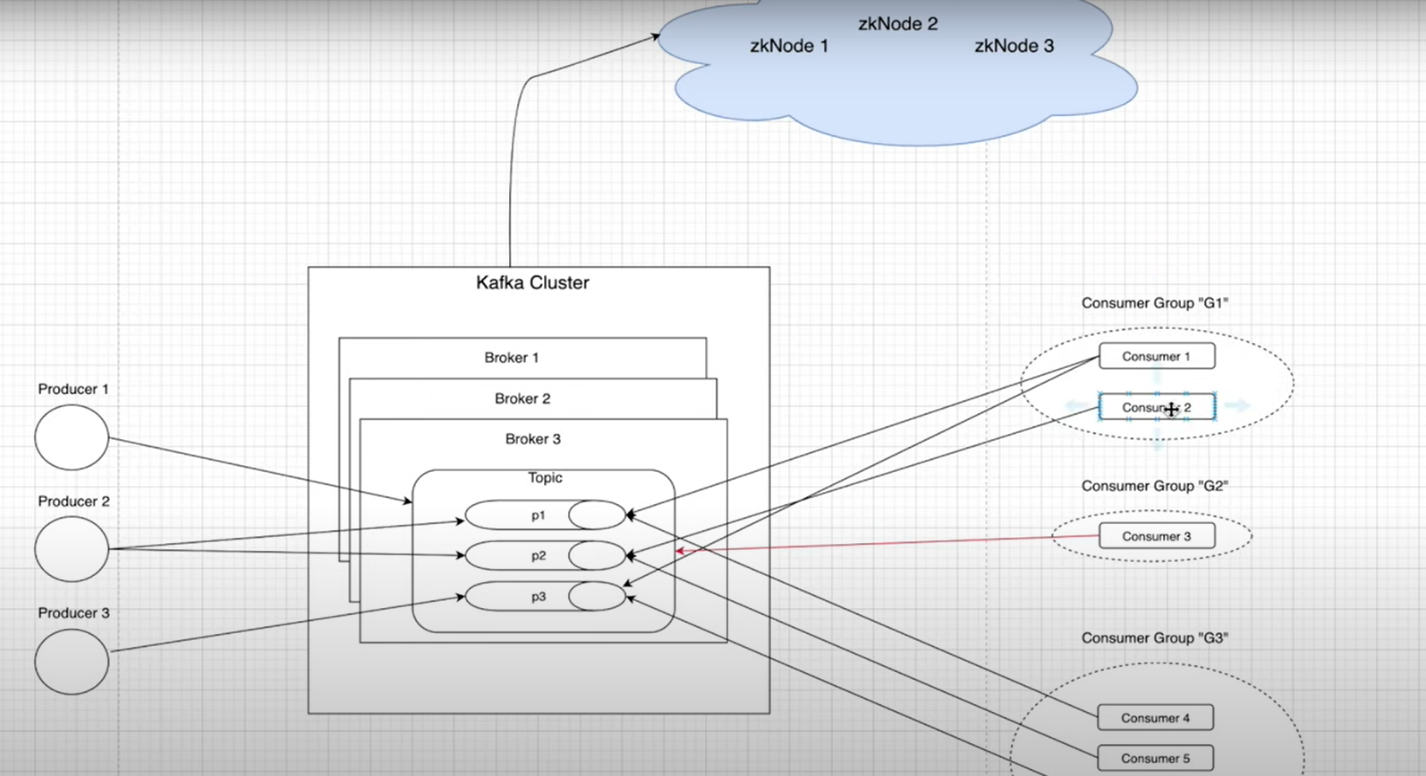
1. how many types of messaging system

                             Two type of messaging system

* Point to Point
* Publish Subscribe message
  + Point to Point:
    - Message are persisted in a queue. one is sender side and the other one is receiver side. once sender sent the data and consumer consume that data the data will be delete.
    - automatically. There is no time limit to read the data in to the queue. once receiver receive the data it acknowledges to the sender
    - Publish Subscribe message:
      * In which one is producer who produce the data they sent the data on a topic and multiple consumers subscribe the same topic. but there is a time limit on it for subscriber. but that is configurable you can change according to your need

              And Kafka is published subscribe messaging system. And it’s is **parallel processing** system.

**Introduction Or Features**

1. 

**Kafka architecture:**

**Topic:**

A stream of messages belonging to a specific category is called topic. For example

We have an employee table in that table we just store only employee information. Just like if we have a finance topic so that mean this topic contain info of finance.

You can create n number of topic but name should be unique. Topic has two properties partition and replication.

**Partition:**

Topic is split in to partitions.

              All message with in the partition is immutable.

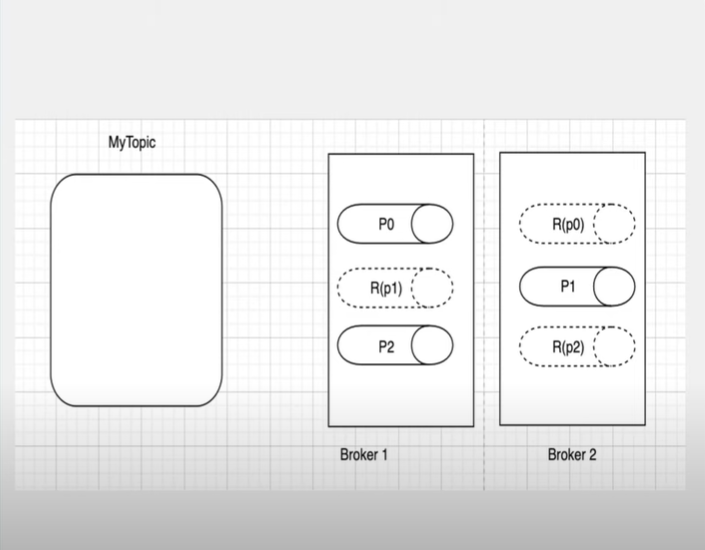
              Each message with in a partition has a unique id associated know as **offset.**

**Replication:**

              Replica is the backup of partition.

              Replicas never read or write data. (Consumer or producer can read or write)

              They are used to prevent data loss.



**Producer:**

* Producer are the application that produce data on to the topic with in the cluster
* Producer produce the data on to the topic either or a specific partition.

**Consumer:**

* Consumer consume the data to the topic or the specific partition.

**Zookeeper**

              Zookeeper is used to monitor the Kafka cluster and co-ordinate with each broker. Keep all the

Metadata information in the form of key value pair.

Metadata includes

1. Configuration information
2. Health status of each broker

A set of zookeeper nodes working together to manage other distributed system known as zookeeper cluster.

**KAFKA FEATURES:**

1. Scalable (Horizontal scaling)
2. Fault tolerance
3. Durable (Throw put)
4. Performance
5. No Data loss
6. Zero down time
7. Reliability

**=========================================**

**---------- Object Oriented Programming ----------**

**==========================================**

1. **What is Multithreading?**

* In Java you can achieve multithreading by using Thread class or Runnable interface. by using thread class, you can override the run method in to class but that time your class itself become a thread.
* If you are using runnable interface your class not become itself thread and by using this you can achieve loss coupling.

1. **Different between abstract class or interface?**

* In Java both are using to achieve abstraction but they have some key difference.

Abstract Class:

* **Nature:**

An abstract class can have both abstract (methods without a body) and concrete methods (methods with an implementation).It can have instance variables (fields).

* **Constructors:**

An abstract class can have constructors, and they are invoked when a subclass object is created.

* **Access Modifiers:**

Abstract classes can have different access modifiers for their members (methods and fields).

* **Single Inheritance:**

A class can extend only one abstract class.

* **Partial Abstraction:**

An abstract class provides partial abstraction, as it can have a mix of abstract and non-abstract methods.

* **Default Method Implementation:**

Abstract classes can have method implementations using the default keyword (introduced in Java 8).

* **Interface:**
* **Nature:**

An interface contains only abstract methods (methods without a body) and constants (public static final fields).

* **Constructors:**

Interfaces cannot have constructors because they cannot be instantiated on their own.

* **Access Modifiers:**

All members of an interface are implicitly public and abstract. Fields are implicitly public, static, and final.

* **Multiple Inheritance:**

A class can implement multiple interfaces.

* **Complete Abstraction:**

Interfaces provide complete abstraction, as all methods are abstract by default.

* **Default and Static Methods:**

Interfaces can have default and static methods with implementations (introduced in Java 8).

* **When to Use Which:**

Use an abstract class when you want to provide a common base class for multiple related classes and share code among them.

Use an interface when you want to define a contract that multiple classes can adhere to, irrespective of their class hierarchy.

In Java, a class can extend only one class (abstract or not), but it can implement multiple interfaces. If you need to define common behavior and structure for multiple classes, and those classes may belong to different class hierarchies, interfaces are often a more flexible choice. If you want to provide a common base with some shared functionality, an abstract class might be appropriate. In many cases, a combination of abstract classes and interfaces is used to achieve the desired design.

1. **Exception Handling:**

What is the difference between **checked** and **unchecked** exceptions?

* **Checked** exceptions are checked at compile-time, and the programmer must handle them using try-catch or declare them using throws.
* **Unchecked** exceptions are not checked at compile-time, and the programmer is not required to handle them.

1. **What is the purpose of the transient keyword?**

The transient keyword is used in object serialization to indicate that a field should not be serialized.

1. **Explain the difference between predicate and function.**

Although they are both functional interfaces, Predicate<T> is a single argument function that returns either true or false. Function<T,R> is also a single argument function, although it returns an object instead. In this case, the “**T**” represents the type of function input, and the “**R**” denotes the type of result.

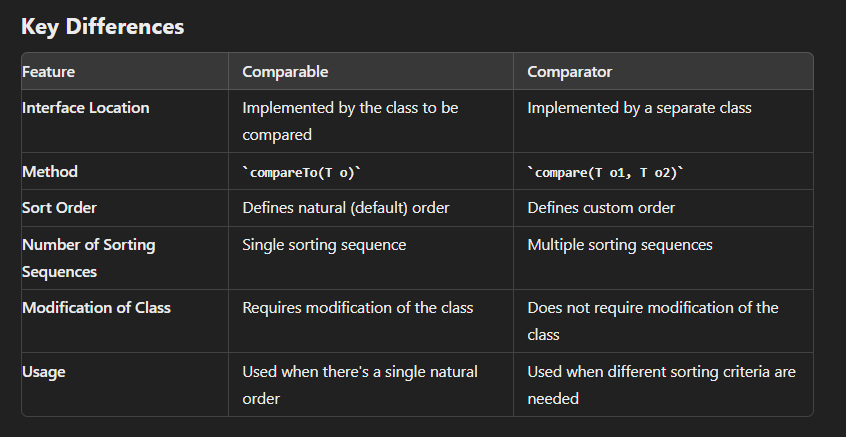
1. **What is the purpose of volatile keyword?**

The volatile keyword ensures that any thread reading the variable sees the most recent modification made by any other thread. It helps to address the visibility problem in a multithreaded environment.

1. **Difference between Comparator, and Comparable?**

They both are java interface in java and that are used for sorting a object.

**Comparable** is used for the natural ordering of objects, while **Comparator** provides a way to define custom ordering without modifying the class itself. The choice between them depends on whether you want to define the natural order within the class or provide multiple ordering strategies externally.



1. **.class or .jar file difference?**

.class files contain compiled Java bytecode for individual classes.

.jar files are archives that can contain multiple .class files, resources

1. **HashSet or HashMap difference?**
2. **Linkedlist or Arraylist?**
3. **Types of exceptions?**
4. **When garbage collector invokes?**

Hibernates Annotations or its working

Abstract or encapsulation difference

Interface or abstract difference

Beanfactory or applicationContext difference

disadvantages of indexing in sql?

JDK, JVM, aur JRE mein farq Java programming language mein hota hai:

JDK (Java Development Kit):

JDK Java ka development kit hai jo programmers ko Java applications banane aur unko run karne ki suvidha pradan karta hai. Is mein Java compiler, Java runtime environment (JRE), aur aurat samagri hoti hai jo Java applications ko banane aur execute karne mein madad karti hai.

JVM (Java Virtual Machine):

JVM Java Virtual Machine hai jo Java bytecode ko machine code mein translate karta hai aur Java applications ko run karta hai. Ye platform-independent hota hai, matlab ek JVM ke under likhe gaye code ko kisi bhi platform par run kiya ja sakta hai. Har ek Java application ke liye alag-alag JVM instance hota hai.

JRE (Java Runtime Environment):

JRE Java Runtime Environment hai jo Java applications ko run karne ke liye zaroori hota hai. Is mein JVM aur Java class libraries shamil hote hain jo Java applications ki runtime environment ko provide karte hain. JRE mein Java Virtual Machine (JVM) bhi shamil hoti hai, lekin ismein koi development tools nahi hote.

How hand shake works? (HTTP and HTTPS)

Question 2

Hashing and hashCode and equals method. Why to override these methods? What is collision?

Advance Javascript, HTML, CSS, Vue JS

Explain about the collections and streaming in java8. Difference between JPA and Hibernate

Why choose this company?

some technical questions related to java springboot nosql and all

What kind of vulnerabilities are there with off site servers

======================= PROGRAMMING QUESTIONS================================

1. Find the first repeated Character in a String?
2. Find the second most repeated Character in a String?
3. Find Factorial
4. Remove the all-repeated Character in a String?
5. Remove 3 or more consecutive characters from a String. e.g.: ABCCCCBBCA -> ABBBCA -> ACA
6. Singlton class
7. Find the max value in to array
8. Find min or max value in to array

**How is a query executed in MySQL?**

This post is for the testers who have some basic idea about MySQL database. I want to put up few things I came across when I dove deeper into MySQL. Here’s a brief overview of the order in which a query is executed inside a MySQL server.

When you execute a SQL query, the order in which the SQL directives get executed is:

* FROM clause
* WHERE clause
* GROUP BY clause
* HAVING clause
* SELECT clause
* ORDER BY clause

However, HAVING and GROUP BY clauses can come after SELECT depending on the order it is specified in the query.

How are queries executed at the back-end by the database engine? Let’s take an example for each of the clauses and understand the sequence.

|  |
| --- |
| **SELECT** \* **FROM** order\_details **WHERE** category = 'produce'; |

The first clause which gets executed is the FROM clause, which is used to list the tables and any joins required for the query. It is through this clause we can narrow down possible record set sizes. The above query is straight forward without any joins.

Let’s take another example with JOIN in the query

|  |
| --- |
| **SELECT** order\_details.order\_id, customers.customer\_name  **FROM** customers  **INNER** **JOIN** order\_details  **ON** customers.customer\_id = order\_details.customer\_id; |

In the above query, JOIN condition is evaluated in the first step. The order of JOIN operation is determined dynamically by the query optimizer when it constructs its query plan. The ON condition is the criteria for deciding which rows to join from each table. The result of the FROM clause is a temporary result (like a temporary table), consisting of combined rows which satisfy all the join conditions. In the above example, MySQL would return all rows from the customers and order\_details tables where there is a matching customer\_id value in both the customers and order\_details tables.

Next, comes the WHERE clause. If you don’t specify the WHERE clause in the statement the optimizer retrieves all the rows from the temporary result. In a query with a WHERE clause, each row in the temporary result is evaluated according to the WHERE conditions, and either discarded or retained.

Next, comes the GROUP BY clause, which is an optional part of the SELECT statement. If there’s a GROUP BY clause, the temporary result is now split into groups, one group for every combination of values in the columns in the GROUP BY clause. When you perform GROUP BY on table it will retrieve the first row in that group. The below GROUP BY example uses the COUNT function to return the product and the number of orders (for that product) that are in the produce category.

|  |
| --- |
| **SELECT** product, **COUNT**(\*) **AS** "Number of orders"  **FROM** order\_details  **WHERE** category = 'produce'  **GROUP** **BY** product; |

Now comes the HAVING clause. The HAVING clause enables you to specify conditions that filter which group results appear in the final results. It operates once on each group, and all rows from groups which do not satisfy the HAVING clause are eliminated. In the below query, after having assembled an entire temporary result table, the optimizer will filter the results so that only products with more than 20 orders will be returned. After the HAVING clause has filtered the groups, a new temporary result set is produced, and in this new temporary result, there is only one row per group.

|  |
| --- |
| **SELECT** product, **COUNT**(\*) **AS** "Number of orders"  **FROM** order\_details  **WHERE** category = 'produce'  **GROUP** **BY** product  **HAVING** **COUNT**(\*) > 20; |

The MySQL HAVING clause is used in combination with the GROUP BY clause to restrict the groups of returned rows to only those whose the condition is TRUE.

Next, comes the SELECT clause. From the rows of the new temporary result produced by the GROUP BY and HAVING clauses, the SELECT now assembles the columns it needs.

Finally, the last step is the ORDER BY clause. The ORDER BY clause is used to sort the records in the result set. In queries with both a GROUP BY and ORDER BY clause, you can reference columns in the ORDER BY only if they are in the new temporary result produced by the grouping process, i.e. columns in the GROUP BY or aggregate functions.

|  |
| --- |
| **SELECT** \* **FROM** (**SELECT** product, **COUNT**(\*) **AS** "Number of orders"  **FROM** order\_details  **WHERE** supplier\_name = 'Microsoft'  **GROUP** **BY** product)  **AS** temp\_table **ORDER** **BY** supplier\_city **DESC**; |

In the above example, GROUP BY will be executed first and then ORDER BY Clause. Using non-aggregate columns in a SELECT with a GROUP BY clause is non-standard. MySQL will generally return the values of the first row it finds and discards the rest. Any ORDER BY clauses will only apply to the returned column value, not to the discarded ones.