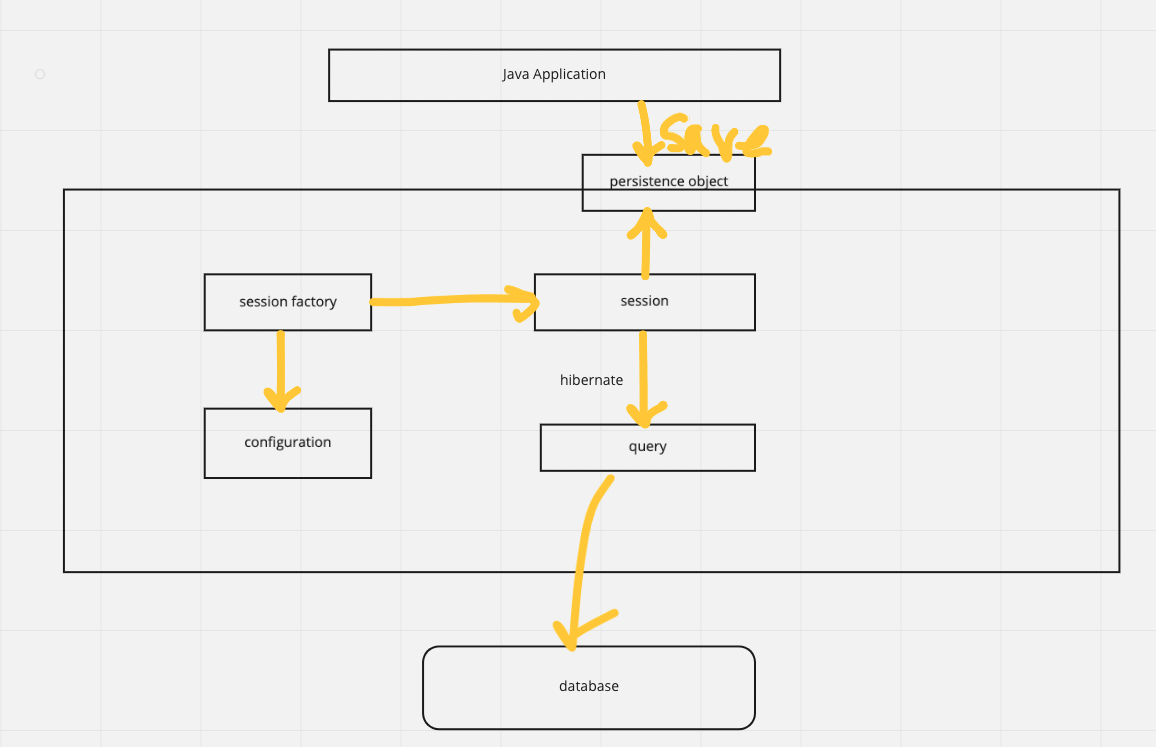
1. 8 basic data types
   1. Primitive type
      1. byte, short, int, long, float, double, char, Boolean
   2. wrapper type
      1. Byte, Short, Integer, Long, Float, Double, Character, Boolean
   3. autoboxing and unboxing
2. String related
   1. String, immutable thread safe
   2. StringBuilder, mutable not thread safe
   3. StringBuffer, mutable thread safe
   4. When use stringBuilder, when use stringBuffer
   5. String pool
3. equals/hashcode
   1. diff between == and equals()
   2. why equals and hashcode have to overwrite together
4. collection
   1. List
      1. ArrayList
      2. LinkedList
      3. Vector
      4. Stack
   2. Queue
      1. Deque
      2. PriorityQueue
   3. Set
      1. HashSet
      2. TreeSet
      3. LinkedHashSet
   4. Map
      1. HashMap
      2. HashTable
      3. LinkedHashMap
      4. TreeMap
      5. ConcurrentHashMap
   5. list vs set
   6. HashMap vs HashTable vs ConcurrentHashMap
   7. HashSet and HashMap
5. Comparator vs Comparable
   1. if sorting of objects needs to be based on natural order then use Comparable whereas if you sorting needs to be done on attributes of different objects, then use Comparator in Java
6. JVM
   1. Class Loader
   2. Runtime Memory/Data area
   3. Execution Engine
7. Garbage Collector
   1. Finalize keyword
8. Keywords
   1. Data types
   2. Flow control
   3. Modifiers
      1. Public
      2. Private
      3. Protected
      4. Static
      5. Final
      6. Abstract
      7. Synchronized
   4. Except handling
      1. Throw
      2. Throws
      3. Finally
   5. Class related
   6. Object related
   7. Final vs finally vs finalize
9. OOP
   1. Abstraction
   2. Encapsulation
   3. Inheritance
   4. Polymorphism
10. Exception
    1. Try catch
    2. Throws vs throw
       1. Throws use in the statement of method, will handle by the user
       2. Throw use in the method block handle by the method
    3. Checked exception vs unchecked exception
    4. Multiple exception
11. Generics
    1. parameterized types
    2. <? extends E>
    3. <? super T>
12. IO Streams
    1. ByteStream
       1. InputStream,outputStream
       2. 1 byte = 8 bits
    2. CharacterSream
       1. Reader writer
       2. 2 byte = 16 bits
    3. File
       1. File class in java.io
       2. Give access to file system
13. Serialization and deserialization
    1. ObjectOutputStream
    2. ObjectInputStream
    3. Object -> byte
    4. Byte -> object
    5. Implement Serializable (no method inside this interface, just like a tag)
14. Java 8 new features
    1. Lambda expression: (parameter1, parameter2) -> expression
       1. It is a short block of code which takes in parameters and returns a value. Lambda expressions are similar to methods, but they do not need a name and they can be implemented right in the body of a method.
    2. Stream API
       1. a special iterator class that allows us to process collections of objects in a functional manner. It computes data, while Collections stores data
    3. Functional interface
       1. an interface with at most one abstract method; implementation can be provided using a Lambda Expression
15. Mutli-threading
    1. Thread vs process
       1. Process: independent memory space, heap, OS resources
       2. Thread: shared memory space, private stack, program counter, register
    2. Thread states
       1. new
          1. thread create, not yet start
       2. runnable
          1. executing in JVM
       3. blocked
          1. wait for a monitor lock to enter synchronized block or method
       4. waiting
          1. Object.wait with no timeout
          2. Thread.join() with no timeout
          3. park()
       5. timed\_waiting
          1. thread sleep
          2. Object.wait() with timeout
          3. thread.join with timeout
          4. park
       6. terminated
          1. thread has completed
    3. thread creation
       1. extends Thread
       2. implements Runnable
       3. implements Callable
    4. Diagram

       Description automatically generatedthread pool
       1. Customize thread pool
       2. In-build thread pool
    5. runnable vs callable
       1. no return / has
       2. no exception / has
       3. run() / call()
    6. Lock
       1. synchronized
          1. block
          2. method
          3. static method
          4. class
       2. Lock interface
          1. lock(), unlock(), newCondition(), tryLock(), lockInterruptibly()
          2. ReentrantLock class
    7. ReadWriteLock interface
       1. method
          1. Lock readLock();
          2. Lock writeLock();
       2. class
          1. reentrantReadWriteLock
16. Database
    1. DBMS(database management system)
       1. MySQL, SQL Server, Oracle
    2. SQL Structured Query Language
       1. Different system have a little diff
17. Normalization
    1. 1NF
       1. Each table cell should contain a single value
       2. Each record needs to be unique
    2. 2NF
       1. Be in 1NF
       2. Single column primary key
    3. 3NF
       1. Be in 2NF
       2. Has no transitive functional dependencies
    4. De-normalization
18. Non-relational Database and no SQL
    1. Document data store: mongoDB(CP)
    2. Key-value data store: redis(CP)
    3. Column family data store: Cassandra(AP)
19. CAP
    1. Consistency
    2. Availability
    3. Partition tolerance
20. Sharding and replica
    1. Sharding
       1. distribute a single logical database across a cluster of machine.
    2. Replica
       1. Redundancy
       2. Failover
    3. SQL vs NoSQL

|  |  |
| --- | --- |
| sql | no-sql |
| relational database | non-relational database |
| pre-defined schema | dynamic schema |
| vertical scaling | horizontal scaling |
| ACID | CAP |
| not suited for hierarchical data store | suited for hierarchical data store |

1. MongoDB
   1. Nosql
   2. Document datastore
   3. Support many languages
2. MongoDB architecture
   1. Mongod: database instance
   2. Mongos: sharding processes
      1. Analogous to a database router
      2. Process all the request based on the info from config servers
      3. Decide how many which mongods should receive the query
   3. Mongo: interactive shell
   4. Functionality of mongoDB
      1. Dynamic schema
      2. Document based database
      3. Secondary indexes
      4. Primary-second node with automated failover
      5. Built in horizontal scaling via automated ranged based partitioning of data(sharding)
      6. CP
3. Redis
   1. In memory
   2. Key/value data store
   3. Supports different kinds of data structure
      1. String
      2. List
      3. Sets
      4. Sorted sets
      5. Hashes
   4. Cache concept(miss and hit)
4. Index
   1. indexing is a way to optimize the performance of database by minimizing the number of the disk access required when a query is processed
   2. clustered index - primary index
   3. non-clustered index - secondary index
5. SQL/Application tuning
   1. Sql tuning
      1. using execution plan to identify the cause of slowness
      2. try to reduce joins, remove unused join and join conditions
      3. use the index to improve the performance
      4. union all instead of union
      5. limit
      6. view or stored procedure
   2. application tuning
      1. check the db query - do the sql tuning
      2. DB connection usage -> connection pool
      3. do JVM tuning -> Jstack, JMap, JConsole
      4. server side: CPU, Memory usage by using commands like top, ps
      5. code review
      6. check networking, firewall, load balancer
6. Transaction
   1. ACID
      1. Atomicity
         1. all transactions are atomic
         2. can’t be executed partially
         3. commit or rollback
      2. Consistency
         1. transactions take the database from one consistent state to another state
      3. Isolation
         1. a transaction is not visible to other transactions until it completes
      4. Durability
         1. once a transaction has completed, its changes are made permanent
7. Concurrency
   1. Dirty data: read uncommitted data from another transaction
   2. Non-repeatable read: read committed data from an update query from another transaction
   3. Phantom read: read committed data from an insert or delete query from another
8. Lock
   1. Binary lock
      1. 1,0
      2. Locked or unlocked
   2. Shard and exclusive locks
      1. Read lock
      2. Write lock
   3. Dead lock
9. Distributed transaction
   1. Saga
   2. 2pc(2 phase commit)
10. SQL
    1. Join
    2. Union
    3. Group by
    4. Having
    5. Where
    6. Intersect
11. JDBC
    1. JDBC Driver is a software component that enables java applications to interact with the database
    2. java application < — > JDBC driver < — > database
    3. A JDBC program comprises the following steps
       1. allocate a connection object, for connecting to the database
       2. allocate a statement object, based on the connection
       3. write sql query, execute query
       4. process the query result
       5. close the statement and connection object to free up the resources
    4. Connection, Statement, executeQuery()
    5. CRUD operation ->insert. read, update. Delete
    6. commit, rollback
12. Hibernate
    1. ORM(Object Relational Mapping)



* 1. Session factory
  2. Entity
  3. Session
  4. Fetch type
     1. Eager loading
     2. Lazy loading
  5. Mapping
  6. Cache
     1. first level cache: session level (open by default)
     2. second level cache: session factory level (close by default)
     3. EhCache
     4. OSCache