1. Database intro
   1. Database – physical place where you store data
   2. Data base management system (BDMS) – middleware that helps the user to manage the database, you can do CRUD operations on database with BDMS
      1. Ex: mySQL, PotsgreSQL, SQL Server, Oracle…
   3. SQL – Structured Query Language, used to view or change the database
      1. Ex: to get 21 – 30

mySQL: select \*

from tableName

order by name

limit 20, 10;

Oracle: select \*

from tableName

order by name

offset 20 rows fetch next 10 rows only;

* 1. Difference between file system and database

|  |  |
| --- | --- |
| File System | DBMS |
| Manage and organize the files in storage medium | Manage the database |
| Redundant data | No redundant data |
| No efficient query processing | Efficient query processing |
| Less data consistency | More data consistency |
| Less security | More security |
| Less expensive | Higher cost |

1. Database normalization – there are 6 normal forms, but we usually only use the first 3
   1. Definition: eliminate redundant data and ensure data is stored logically
   2. 1NF
      1. Each table cell should contain a single value
      2. Each record needs to be unique
      3. Example:

|  |  |  |  |
| --- | --- | --- | --- |
| id | name | Phone | gender |
| 1 | Tom | 123 | F |
| 2 | Tom | 123 | F |

* 1. 2NF
     1. Be in 1NF
     2. Single column primary key
     3. Just intro: composite primary key (not 2NF) – use combined columns as a primary key
     4. Example:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | firstName | lastName | age | gender |
| 1 | Tom | Zhang | 1 | F |
| 2 | Tom | Wang | 2 | M |
| 3 | Tom | Wang | 3 | F |

* 1. 3NF
     1. Be in 2NF
     2. Has no transitive functional dependencies
     3. Example:

City and zipcode has transitive functional dependent – if change city, zipcode will be changed accordingly.

|  |  |  |
| --- | --- | --- |
| id | city | zipcode |
| 1 | NY | 123 |
| 2 | DC | 321 |

Fixed: city can be the primary key

|  |  |
| --- | --- |
| city | zipcode |
| NY | 123 |
| DC | 321 |

* 1. De-normalization – lower the normal form

1. Non-relational database and no SQL
   1. No pre-defined rows or columns (prime difference between relational and non-relational)
   2. Major categories of non-relational database
      1. Document based data store
         1. Text

            Description automatically generated
         2. **MangoDB**, couchDB
      2. Key-value based data store – like a super big hashMap
         1. Table

            Description automatically generated
         2. **Redis**, riak
      3. Column family data store / columnar data store
         1. Table

            Description automatically generated
         2. **Cassandra**, Hbase
      4. Graph data store
         1. Diagram

            Description automatically generated
         2. Neo4j, GraphDB
   3. CAP principle – pick two of the three features, usually pick one from A and C
      1. Consistency: all clients always have the same view of the data
      2. Availability: each client can always read and write
      3. Partition tolerance: the system works despite physical network partitions
      4. AP: Dynamo, Cassandra, SimpleDB, CouchDB…
      5. CP: BigTable, MongoDB, Hbase, Redis…
      6. Diagram

         Description automatically generated
2. Sharding and replica
   1. Sharding: distribute a single logical database across a cluster of machines

Diagram, table

Description automatically generated

* 1. Replica
     1. Redundancy
     2. Failover
     3. Diagram, icon

        Description automatically generated

1. SQL vs no-SQL

|  |  |
| --- | --- |
| 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
2. Redis
4. Index – a way to optimize the performance of database by minimizing the number fo the disk access required when a query is processed.
   1. Clustered index – primary index
      1. Defines the order in which data is physically stored
      2. Only one clustered index per table
   2. Non-clustered index – secondary index
      1. As many as non-cluster index
      2. Doesn’t sort the physical data inside the table
   3. Data structure
      1. B+ tree (default)

Diagram

Description automatically generated

* + 1. Bitmap
    2. Hashtable
    3. R tree

1. 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 peocedure
   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
2. Transaction – A transaction is an action, or a series of actions, carried out by a single user or an application program.
   1. ACID Principle
      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
3. 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 transaction
   4. Isolation level:

Table

Description automatically generated

1. Lock
   1. Binary lock – 1 or 0, lock or unlock
   2. Shared and exclusive lock
      1. Shared lock: read lock
      2. Exclusive lock: write lock
   3. Optimistic lock and pessimistic lock
   4. DeadLock
      1. How to detect? Wait for graph
   5. LiveLock
2. Distributed Transaction
   1. Sage
   2. 2PC (two phase commit)
3. SQL
   1. DDL (data definition language)
      1. create, drop, alter, truncate,
   2. DQL(data query language)
      1. select …
   3. DML (data manipulation language)
      1. insert, update, delete
   4. DCL (data control language)
      1. grant, revoke
   5. DTL (data transaction language)
      1. commit, rollback
   6. queries