Read-After-Write Consistency

- One method for implementing read-after-write consistency is that modifiable data (from the client's perspective) is always read from the leader
- Another method for implementing read-after-write consistency is to dynamically switch to reading from the leader for "recently updated" data
- The challenge these methods create is that followers were supposed to be proximal to users, but now you have to route requests to distant leaders

Key-Value Stores

- Key-value stores are simple, especially in comparison to RDBMS
- Retrieving a value in a key-value store given its key is typically a O(1) operation because hash tables or similar data structures are used under the hood
- There are no concepts of complex queries or joins that slow things down
- Horizontal scaling is simple in key-value stores, just add more nodes
- Key-value stores are typically concerned with eventual consistency, meaning that in a distributed environment, the only guarantee is that all nodes will eventually converge on the same value

Redis

- Redis stands for Remote Directory Server
- Redis is an open source, in-memory database, sometimes called a data structure store
- It is primarily a KV store, but can be used with other models such as Graph, Spatial, Full Text Search, Vector, Time Series
- Redis supports the durability of data by essentially saving snapshots to disk at specific intervals or append-only file which is a journal of changes that can be used for roll-forward if there is a failure
- Redis only supports lookup by key
- Redis keys are usually strings (but can be any binary sequence) and its values can be strings, lists, linked lists, sets, sorted sets, hashes, and geospatial data
- Redis provides 16 databases by default, numbered 0 to 15
- Value of KV entry is a collection of *field-value* pairs
- Can be used to represent basic objects/structures
    - number of field/value pairs per hash is 2^32-1
    - practical limit: available system resources (e.g. memory)
- Stack: Last In, First Out
- Queue: First In, First Out
- Redis fully supports JSON standard, internally it is stored in a binary tree structure for fast access to sub-elements

BSON

- BSON is a binary-encoded serialization of a JSON-like document structure
- BSON supports extended types not part of basic JSON (e.g. Date, BinaryData, etc)
- BSON is lightweight and keeps space overhead to a minimum
- BSON is traversable, designed to be easily traversed, which is vitally important to a

document database
- BSON is efficient - encoding and decoding *must* be efficient
- Supported by many modern programming languages

XML

- Xquery is the SQL of XML
- **XSLT** - eXtensible Stylesheet Language Transformation - tool to transform XML into other formats, including non-XML formats such as HTML

Document Databases

- Document databases address the *impedance mismatch* problem between object persistence in OO systems and how relational DBs structure data.
- Impedance mismatch is when two computers/systems are supposed to work together but have different data models or structures.
- The structure of a document is self-describing.
- Document databases are well aligned with apps that use JSON/XML as a transport layer.

MongoDB Commands

- SELECT * FROM users = db.users.find()
- SELECT * FROM users WHERE name = “Dave” = db.users.find({“name”: “Dave”})
- Movies released in Mexico, IMDB rating at least 7: db.movies.find({“countries”: “Mexico”, “imdb.rating”: {$gte: 7}})
- How many movies from the movies collection were released in 2010 and either won at least 5 awards or have a genre of Drama: db.movies.countDocuments({“year”: 2010, $or: [{ "awards.wins": { $gte: 5 } }, { “genres”: "Drama" }]})
- Return the names of all movies from the movies collection that were released in 2010 and either won at least 5 awards or have a genre of Drama: db.movies.countDocuments({“year”: 2010, $or: [{ "awards.wins": { $gte: 5 } }, { “genres”: "Drama" }]}, {“name”: 1, “_id”: 0}) where 1 = return, 0 = don’t return
- SELECT count(*) FROM collection = db.collection.count_documents({})

Graph Data Model

- Examples of graphs are social networks, the web, and chemical/biological data
- In a labeled property graph, labels are used to mark a node as part of a group
- In a labeled property graph, nodes with no associated relationships are okay, but edges not connected to nodes are not allowed

Trees

- A binary tree has up to two child nodes and no cycles.
- A spanning tree is a subgraph of all nodes, but not all relationships of the original graph

(no cycles)

Pathfinding

- Pathfinding is finding the shortest path between two nodes, if one exists, and is probably the most common operation for graphs
- The “shortest” path means fewest edges or lowest weight
- Average Shortest Path can be used to monitor efficiency and resiliency of networks.
- Minimum spanning tree, cycle detection, max/min flow are other types of pathfinding
- Breadth-First Search (BFS) is a graph traversal algorithm that explores all nodes at the current level before moving on to the next level, starting from a designated root node and systematically visiting neighbors
- Depth-First Search (DFS) is a graph traversal algorithm that explores as far as possible along each branch before backtracking, often implemented using a stack data structure.
- Shortest path is the shortest path between two nodes
- All-pairs shortest paths is optimized calculations for shortest paths from all nodes to all other nodes
- Single source shortest path is the shortest path from a root node to all other nodes
- Minimum spanning tree is the shortest path connecting all nodes
- Features of centrality:
    - Degree refers to the number of connections. If a node is connected to many other nodes, it has a high degree.
    - Betweenness refers to whether a node has control over the flow between nodes and groups.
    - Closeness refers to nodes that can easily reach all other nodes in a graph or subgraph.
- Dijkstra's Algorithm is a single-source shortest path algorithm for positively weighted graphs, where the shortest distance node is always chosen next.
- A* Algorithm is similar to Dijkstra’s algorithm with the added feature of using a heuristic to guide traversal.
- PageRank measures the importance of each node within a graph based on the number of incoming relationships and the importance of the nodes from those incoming relationships

Neo4j

- Neo4j is a Graph Database System that supports both transactional and analytical processing of graph-based data, and it is a relatively new class of no-sql DBs
- Neo4j is considered schema optional (one can be imposed) and supports various types of indexing
- Neo4j is ACID compliant and supports distributed computing