NoSQL Databases

Overview

NoSQL

- Not only SQL
- SQL
  - Most popular
  - Widely used
  - De facto standard
  - Based on relational algebra
    - Relational algebra does not always work for certain database types

Querying Without SQL

- Typically involves a lot of procedural programming
- A lot more work than SQL
- Higher skill set than SQL
- Longer development times than SQL
- Result
  - NoSQL databases try to adapt to allow SQL
  - SQL layer on top of the non-SQL query layer
  - SQL-like query languages

NoSQL Database Attributes

NoSQL databases typically have one or more of these attributes:

- Key-value
- Document
- In-memory
- Wide column
- Graph
- Time series
- Ledger

NoSQL Key-Value

- Similar to a Python dictionary
- Keys must be unique
- Value can be anything
- Tends to be in-memory with backing store
- Must query using the key or search the entire database

NoSQL Document

- Collection of documents
- Typically each document is JSON or JSON-like format
- Tends to be in storage with caching to memory
- Query based on anything in the document
- Can use indices to speed up queries
- Queries return entire documents
- Advantage is that entire documents can be the equivalent of multiple tables

NoSQL In-Memory

- Entire database fits into memory
- Often used with key-value
- Could be used with any type of database
- Major advantages
  - Fast
  - Do not need indices
- Major disadvantage
  - Memory is more expensive than storage

NoSQL Wide Column

- Wide column tables are similar to relational database table
- Storage back end is different
  - Horizontal partitioning
  - Vertical partitioning—column headers—columnar
  - Immutable model
    - Bulk inserts and deletes
    - No individual inserts, no individual deletes, no updates
  - Eventual consistency (BASE)
- SQL on top of wide column SQL makes it very similar to querying a relational database

NoSQL Graph

- Graph
  - Nodes (vertices)
  - Relationships (edges)
- Many problems are naturally represented as graphs
- SQL-like query language

NoSQL Time Series

- Time-based (or temporal) data
- Measures how things change over time
- Especially good for high volumes of streaming data coming in with timestamps
- SQL-like query language

NoSQL Ledger

- Central authority ledger with blockchain verification
- Compare to cryptocurrencies which lack central authority
- SQL-like query language

Most Common

- We are going to take a deeper look at three of the most common types of NoSQL databases in use:
  - NoSQL graph
  - NoSQL document
  - NoSQL key-value
- NoSQL wide column is also very common
  - Uses SQL on top
  - User level: very similar to querying SQL against relational table

The End

NoSQL Graph Databases

Overview

NoSQL Graph

- Basic graph
  - Nodes (vertices)
  - Relationships (edges)
- Many problems are naturally represented as graphs
- SQL-like query language

Nodes and Relationships

- Nodes (vertices)
  - Can have labels for classification purposes
  - Attributes
    - Key-value pairs
- Relationships (edges)
  - Type
  - Direction
  - Attributes
    - Key-value pairs

Graph Types and Structures

- Shapes
  - Random, small-world, scale-free
- Characteristics
  - Connected, disconnected, weighted, unweighted, directed, undirected, acyclic, cyclic, trees
- Density
  - Calculation, sparse, dense
- k-partite
  - Monopartite, bipartite, k-partite

Graph Algorithms

- Path algorithms
- Centrality algorithms
- Community detection algorithms

Graph Feature Engineering

- Types of graph-related features
  - Graphy features
  - Graph algorithm features
- Feature engineering is for:
  - AI (artificial intelligence)
  - ML (machine learning)
  - DL (deep learning)

Graph-Based Model Evaluation

Use graphs to evaluate models

- AI (artificial intelligence)
- ML (machine learning)
- DL (deep learning)

The End

NoSQL Graph Types and Structures

Concepts

Graph Shapes

- Random
- Small-world
- Scale-free

Random

- Flat
- No patterns
- All nodes have the same probability of being attached to each other

Small-World

- High degree of local clustering
- Short average path lengths
- Hub and spoke
- No node more than a few relationships away from any other node

Scale-Free

- Hub and spoke in multiple scales
- Power-law distribution
  - Change in one quantity results in relatively proportional change in another quantity

Graph Characteristics

- Connected, disconnected
- Weighted, unweighted
- Directed, undirected
- Acyclic, cyclic
- Trees

Connected, Disconnected

- Connection
  - Path between any two nodes
  - Regardless of distance between the nodes
- Issues
  - Disconnected nodes
    - May not be analyzed in most graph algorithms
  - An island of connected nodes disconnected from the main graph
    - May not be analyzed in most graph algorithms
    - Can cause algorithms to get stuck with no way out

Weighted, Unweighted

- Weight
  - Numeric values placed on a relationship
- Issues
  - If an algorithm requires weights, unweighted relationships would not be considered
  - If entire graph is unweighted, weighted algorithms are not appropriate

Directed, Undirected

- Directed: relationship has a direction
- Undirected: relationship does not have a direction
- Issues
  - If an algorithm requires direction, undirected graphs would not be appropriate
  - If an algorithm does not consider direction, and we have a directed graph, reconsider

Cyclic, Acyclic, Trees

- Cyclic: has cycles—path from a node back to itself
- Acyclic: no cycles—no path from any node back to itself
- Issues
  - If we have cycles, some algorithms can get stuck
  - Lots of common algorithms require acyclic

Trees, Spanning Trees

- Tree: acyclic graph
  - Computer science: directed or undirected
  - Formal discrete mathematics: undirected only
- Spanning tree
  - All nodes in a graph
  - Relationships removed to remove cycles
- Minimum spanning tree
  - Connects all nodes with minimum hops
- Numerous spanning tree algorithms

Graph Density

- Maximum density = (nodes [nodes – 1]) / 2
- Actual density = (2 * relationships) / (nodes * [nodes – 1])
- Sparse = low density
- Issues
  - High level of density
    - Identify and peel off layers
  - High level of sparsity
    - See if we can add relationships

k-Partite Graphs

- Monopartite
  - One node type, one relationship type
- Bipartite
  - Two sets, nodes from one set only connect to nodes in the other set
- k-partite
  - Number of node types
  - In real world, most graphs have a high k value

The End

NoSQL Graph Types and Structures

Business Cases

Random Network

- Interactions involving Social Security numbers
- All demographics across the USA
- Births
- Deaths
- Paychecks
- Government assistance programs
- Retirement
- Disabled

Small-World Network

- Social network
- Lots of small local networks
  - Most people connected to fewer than 500 other people
- Despite the small local networks
  - Most people are a few hops from anyone on the planet
  - Six degrees of separation

Scale-Free Network

- World Wide Web
- Hub and spoke in multiple scales
- Each scale up of hub and spoke
  - Power law
  - Proportional scale up

All Roads in the USA, Part I

- Connected, disconnected
  - Highly connected over 48 states and DC
  - Disconnected nodes: Alaska, Hawaii, Puerto Rico, Guam, etc.
- Weighted, unweighted
  - Weight freeways and interstate highways highest
  - Weight gravel roads lowest
  - Weight traffic congestion periods, construction, special events

All Roads in the USA, Part II

- Directed, undirected
  - Two-way streets
  - One-way streets, freeways, entrance and exit ramps
- Cyclic, acyclic
  - Highly cyclic: many places, numerous ways to get from A to B
- Trees
  - Create spanning trees of major roads
  - Use the full graph to get you to and from the spanning tree

All Roads in the USA, Part III

- Density
  - High density in cities
  - Low density in rural areas with small towns
  - High sparsity in remote rural areas
- k-partite
  - High k value—numerous node and relationship types

The End

NoSQL Graph Path Algorithms

Concepts

Path Algorithms

- Breadth-first search/depth-first search
- Eulerian circuits/Hamiltonian circuits
- Shortest path (with A* and Yen’s variations)
- All pairs shortest path
- Single source shortest path
- Minimum spanning tree
- Random walk

Breadth-First vs. Depth-First

- Searching trees
- Breadth-first search
  - Visit sibling nodes first before visiting child nodes
- Depth-first search
  - Visit child nodes first before visiting sibling nodes

Eulerian vs. Hamiltonian Circuits

- Eulerian circuit
  - Visit every relationship only one time
  - Visit nodes one or more times
- Hamiltonian circuit
  - Visit every node only one time
  - Visit relationships one or more times

Shortest Path

- Find the shortest path between two nodes
- Variations
  - A* (A star)
  - Yen’s

All Pairs Shortest Path

Find the shortest path between all pairs of nodes

Single Source Shortest Path

Find the shortest paths from one node pairwise with all other nodes

Minimum Spanning Tree

Find a tree structure path that will visit all nodes in the graph with smallest cost

Random Walk

Given a path size, randomly walk through nodes until we hit the path size

The End

NoSQL Graph Path Algorithms

Business Cases

Map App

- Find nearby places of interest
- Solution
  - Use a breadth-first search

Game Simulation

- Simulate moves several levels deep to find the best move
- Solution
  - Use a depth-first search

Driving Directions

- Find driving directions between point A and B
- Solution
  - Use a shortest path algorithm with weights for road size, traffic lights, traffic, construction, etc.

Unexpected Traffic Jam

- We are a GPS routing company
- A traffic jam occurs
- We want to route our customers around the traffic jam
- Solution
  - All pairs shortest path to find all routes around the traffic jam

Home to Freeway

- We are a GPS routing company
- Our customers want to get from their home to the freeway as quickly as possible to get to work
- Numerous ways to get to the freeway
- Solution
  - Single source shortest path with weights for road size, traffic lights, traffic, construction, etc. to find all routes from home to freeway

Laying Fiber

- We are a telecom company wanting to lay new fiber cabling to businesses and homes around the city
- Minimize cost for laying fiber
  - Implies laying the smallest amount of fiber
- Maximize internet speed for our customers
  - Implies each customer has the shortest path to data center
- Solution
  - Minimum spanning tree

Explore Graph Structure

- We have a dataset we are using for AI (or ML or DL)
- We want to enhance our AI training by also creating a graph for some elements
- Since a graph is a secondary structure, not a primary structure, we need to statistically explore it
- Solution
  - Random walks

The End