CPSC 416 Distributed Systems

Winter 2023 Term 1 (November 2, 2023)

Tony Mason (fsgeek@cs.ubc.ca), Lecturer

Logistics

Teaching Assistants

Andy Hsu (andy.hsu@alumni.ubc.ca)

Hamid Ramezanikebrya (hamid@ece.ubc.ca)

Jonas Tai (jonastai@student.ubc.ca)

Cathy Yang (kaiqiany@student.ubc.ca)

Office Hours

Remember: Use Piazza for **all** official course-related communications

- Not on Piazza? Not official.
- Canvas “comments/messages” **are not monitored**

Office Hours:

Who	When	Where
Tony	Monday 14:00-15:00 Wednesday 16:00-17:00	Discord
Andy	Thursday 19:00-20:30	Discord
Hamid	Friday 16:30-18:00	Kaiser 4075
Jonas	Thursday 13:00-14:00	X241
Cathy	Friday 09:00-10:30	X237

Self-Assessment

Next week

- Usual self-assessment activity (Tue/Thu @ 17:00)
- Capstone Week 3 Report (Tue @ 17:00)
- DP3 Implementation Code (Thu @ 17:00)
- DP3 Implementation Report (Thu @ 23:59)

Note:

- You are strongly encouraged to collaborate with others on this
- You should use tools at your disposal to answer these questions
- **Do not forget to submit it.**

Today’s Failure

Catastrophic Software Failures

December 2022: Southwest Airlines IT melt-down

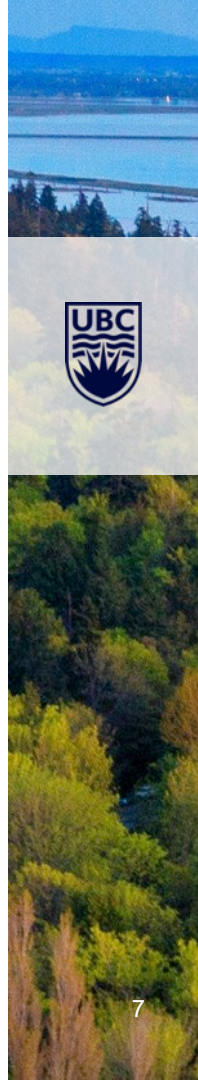
- Cause: IT infrastructure failures, reduced staffing, custom IT solutions
- Cost: $725 millon (direct) + $1.3 billion (“improved technology”)

January 11 2023: US FAA Outage

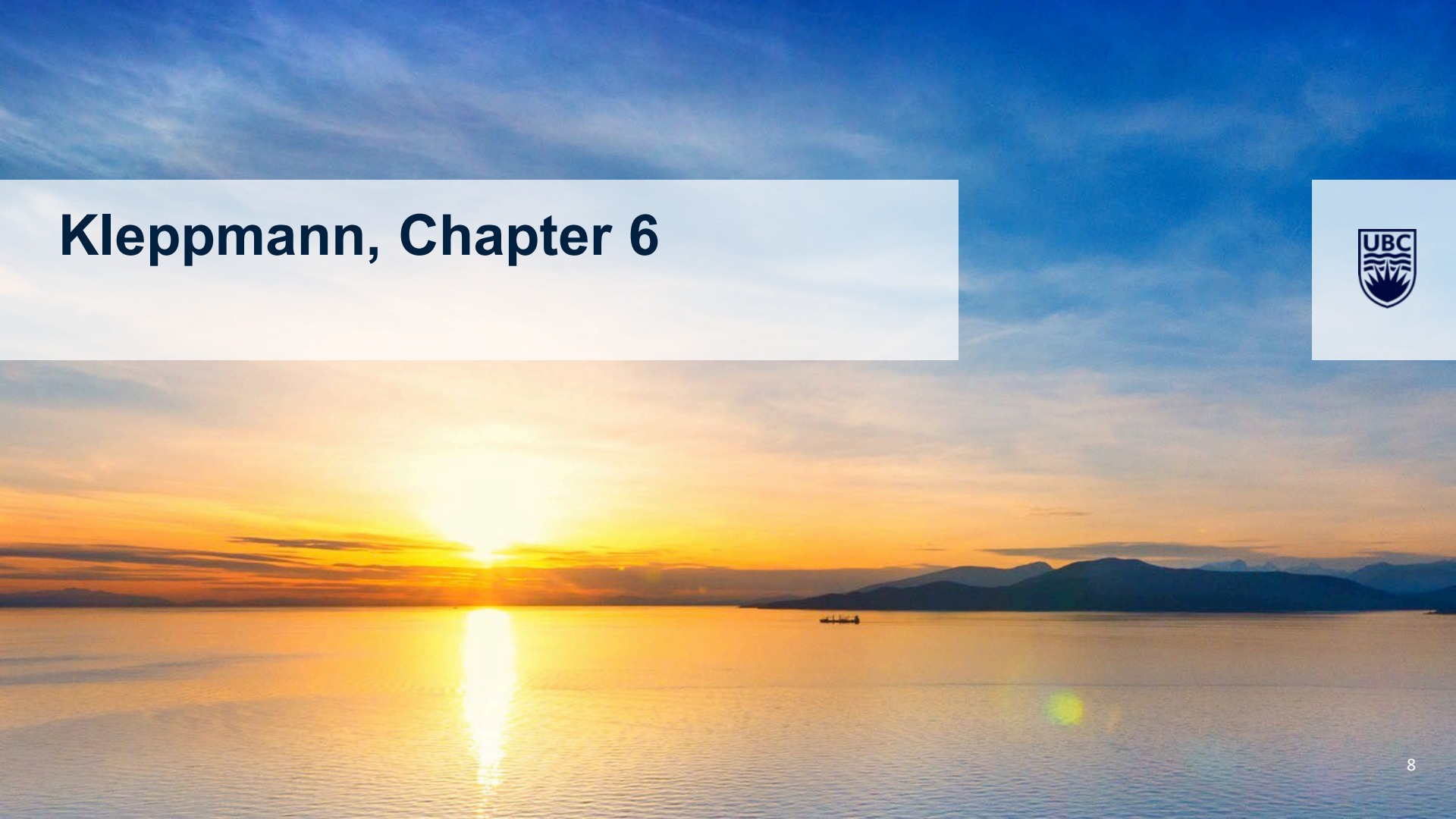
- Cause: a single corrupted (database) file
- Cost: 7,000 canceled flights

January 25 2023: NYSE trading issues

- Cause: Misconfiguration (of “Disaster Recovery” systems!)
- Cost: trading halt, reversed transactions



Kleppmann, Chapter 6



Learning Goals (Kleppmann, Chapter 6)

Partitioning as a concept for managing large datasets

Familiarity with the terminology and meaning of common terms

How partitioning is used to create scalability

Understanding common partitioning strategies

Exploring the impact of partitioning and indexing on performance

Delving into the need for rebalancing in dynamic partitioned systems

How replication and partitioning are used to provide fault tolerance

Handling request routing in partitioned environments

The evolution of partitioned databases

Workload impact on partitioning strategies

The Essence of Partitioning

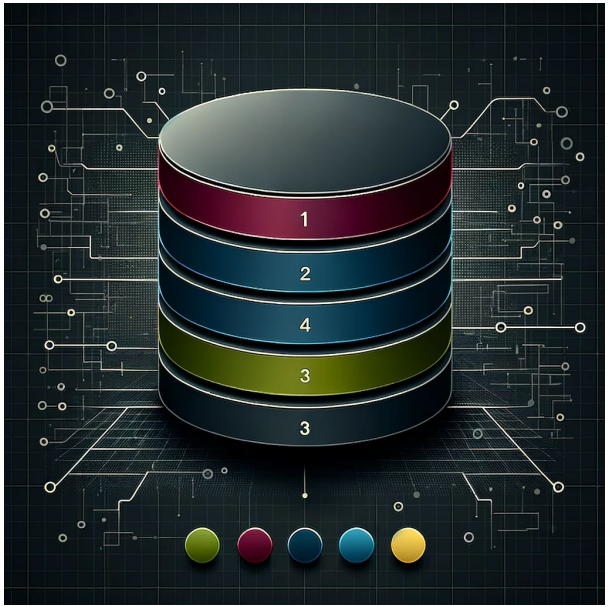
Partitioning (“sharding”) is the process of dividing a database into smaller, manageable pieces

This allows:

- Handling large datasets more efficiently
- Handling large query loads efficiently

Why?

- Network capacity
- Storage capacity
- CPU capacity



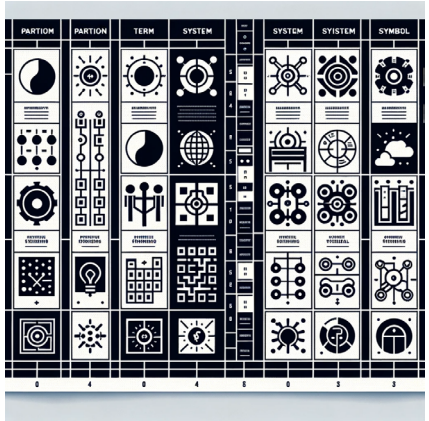
Partitioning: A Lexicon

Partitions have many names

- Shard
- Region
- Table
- Vnode
- Bucket

Common concept: divide data in a database into smaller chunks

- KV store is popular because it makes this *easy*
- Storage is often *aggregated* in similar fashion (e.g., RAID sets.)



Horizontal Scaling with Partitions

Divide data into distinct subsets

- Each partition (“shard”) is independently manageable
- Subset generation is *flexible* (pick the one for the job)

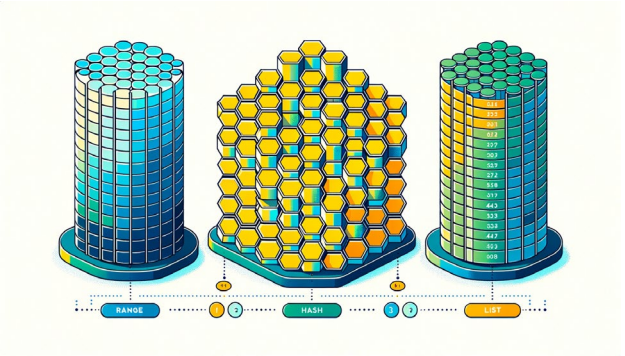
Permits parallel activity

- Assumes independence
- *Routing* overhead must be much lower than processing

Enables dynamic resourcing

- Add more servers = increased capacity
- Requires rebalancing

Data Partitioning Approaches



Range: use keys to “bucket” data (e.g., dates)

Hash: use keys + hash (e.g., customer ID)

List: explicit placement

Composite: combination of two or more other strategies

Round-robin: cyclic assignment

Directory: lookup table

Vertical: partition against sets of columns

Horizontal: partition against sets of rows

Functional: partition based upon expected consumer of the data

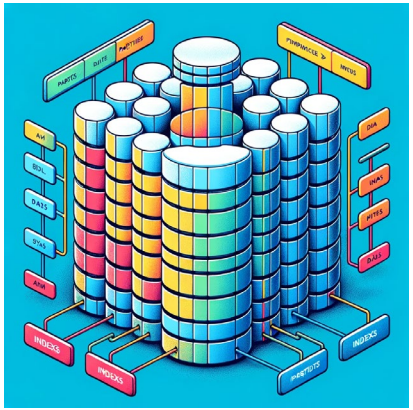
Note: there are many others, usually tuned to specific data sets and/or usage patterns

Indexing versus Partitioning

Indexing empowers query performance

- Local indices are specific to the partition
- Global indices *span* partitions
  - Can also be partitioned

Rebuilding indices can be *expensive*



Index: efficient lookup but extra I/O (update)

Partition: good scaling – so long as you don't involve multiple partitions in a single operation

Cost for both: increased complexity

Goal: balance *complexity* against *performance*

Rebalancing

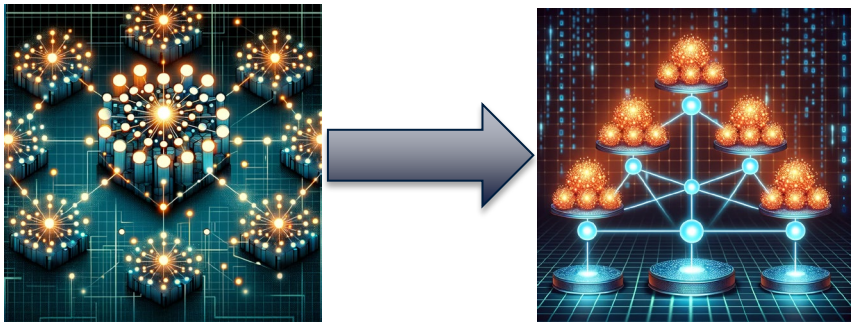
Goal: distribute data evenly over partitions

- Enables dynamic partitioning (add/remove resources)
- Spread load evenly for better utilization and higher performance

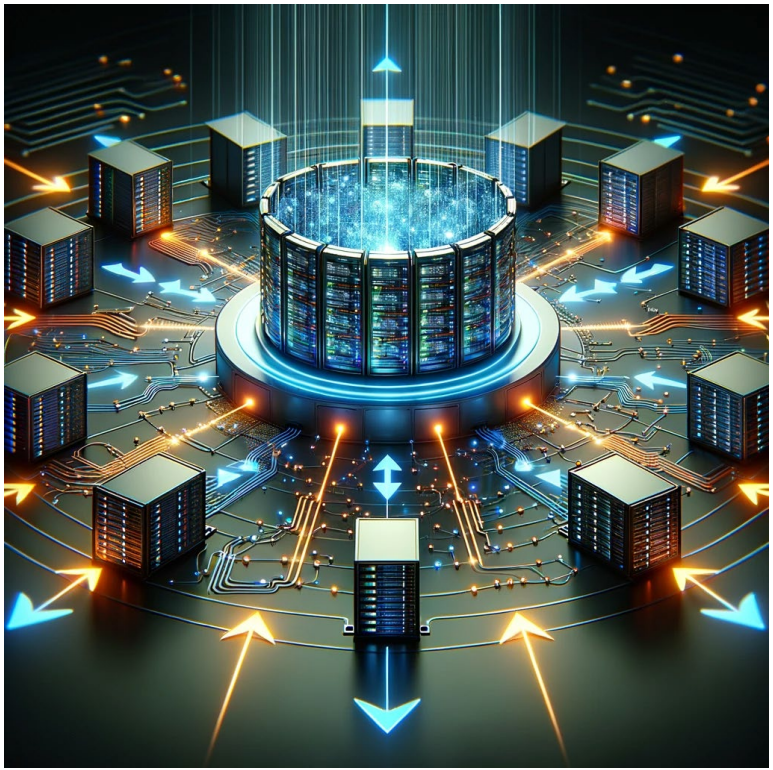
Challenges:

- Rebalancing can be expensive
- Ensuring data integrity
- Rebalancing *at runtime*
- Resource consumption

Yet another layer of complexity!



Replication & Fault Tolerance



Partitioning provides scalability

Replication provides data redundancy

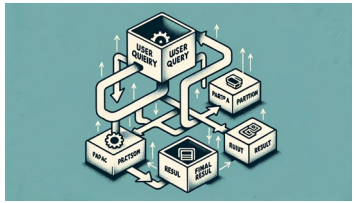
- Multiple copies
- Transparent failover

Combining them provides robust scalability

Cost? Complexity of course

- Replication strategy
- Consistency Model
- Routing/request management

Routing & Query Execution



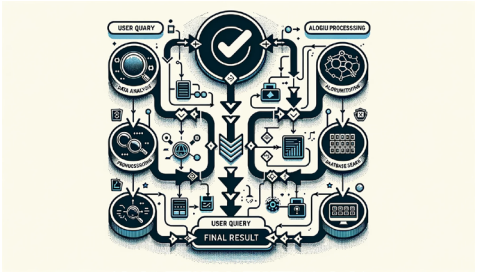
Operations must be routed to correct partition

- Partitioning key – map record to partition key
- Map – maps specific values to partitions (e.g., “customer ID”)
- Routing Layer – implements more complex routing decision(s)
- Fixed (hash) routing, consistent hashing
- Lookup table (“Directory”)
- Client-side
- Query language rewriting

Multi-partition

- Multicast + aggregation

Replication + Failover



Historical Context of Partitioning

1960s – File System Storage

1970s – Relational Database Management System (RDBMS)

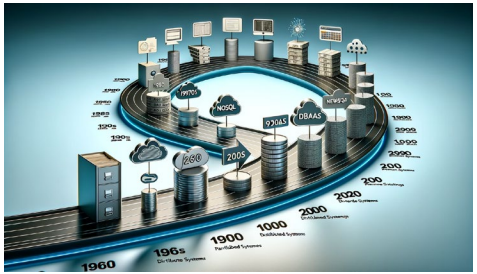
1980s – RAID (Redundant Array of Independent Disks)

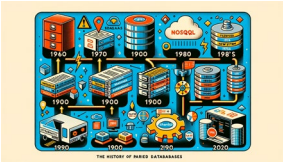
1990s – Distributed Databases

2000s – NoSQL Databases (Document, Graph, Vector)

2010s – Cloud-based Databases, Sharding/Replication

2020s – Real-time Processing Databases, Vector Databases





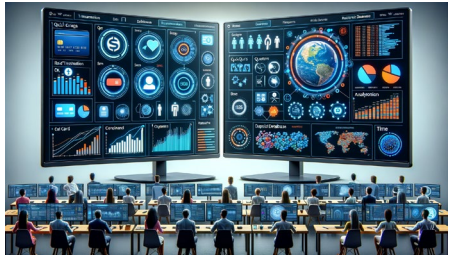
Workloads: Transactional versus Analytical

Transactional:

- Online Transaction Processing (OLTP)
- High concurrency
- Fast Query Processing
- ACID properties
- Fine grained locking
- Index Optimization
- Small Transactions

Analytical:

- Complex Analysis
- Large Data volumes
- Business Intelligence
- Complex queries
- Read-intensive operations
- Delayed consistency
- Columnar storage
- Batch Processing



Questions?


THE UNIVERSITY OF BRITISH COLUMBIA
THE UNIVERSITY OF BRITISH