Big Data

Tutorial #6

Isabelle Kuhlmann
2020-06-19

Institute for Web Science and Technologies · University of Koblenz-Landau, Germany

Outline

1) Lecture Recap
2) Explain Plans
3) Homework Exercise

What previously happened...

Lecture Recap

Sharding

- Distribute parts of data to different nodes
- Aspects that need to be considered:
  - Location
  - Load balance
  - Sequential reads
- Either *manual sharding* or *auto sharding*

Master-Slave Replication

- Only the master receives updates and propagates them to the slave nodes
- Disadvantages:
  - Master is bottleneck for updates
  - Potential inconsistencies if slaves have different values
- Advantages:
  - Read-resilient
  - Slave can be made master if original master fails

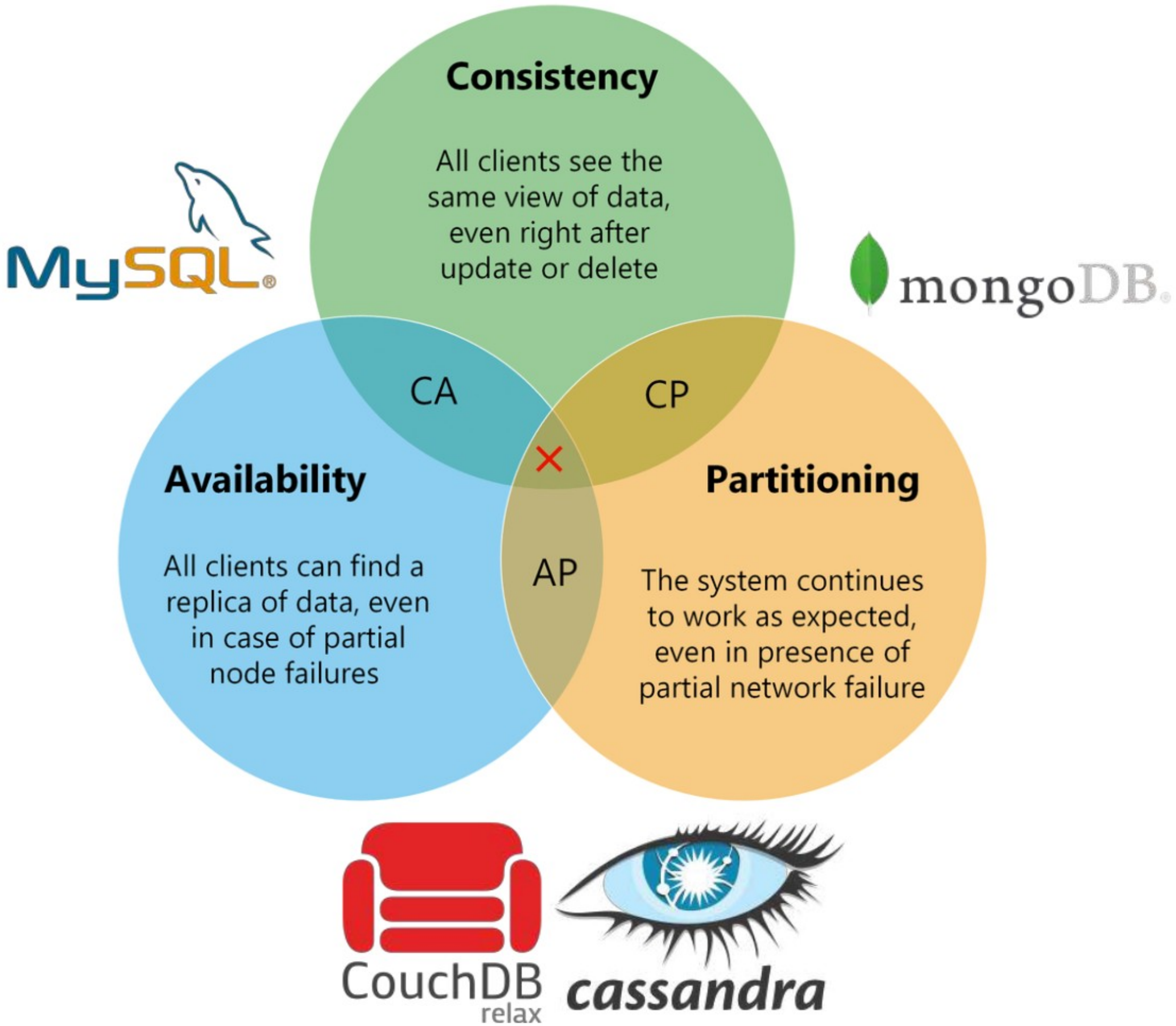
Peer-to-Peer Replication

- Each node can receive updates and can propagate them to the other nodes
- Disadvantage:
  - Potential for inconsistencies: write-write conflict/inconsistencies on read
- Advantage:
  - Resilient against loss of peer

Consistency

- Two strategies of maintaining consistency:
  - **Pessimistic:** prevent conflicts, e.g. by locking resources
  - **Optimistic:** discover and resolve conflicts, e.g. by conditional updates
- Types of inconsistency:
  - Logical inconsistency
  - Replication inconsistency
  - Read-your-writes inconsistency

CAP Theorem


Consistency
All clients see the same view of data, even right after update or delete
MySQL
mongoDB
CA
CP
Availability
Partitioning
All clients can find a replica of data, even in case of partial node failures
AP
The system continues to work as expected, even in presence of partial network failure
CouchDB relax
cassandra

Quorums

- *Intuition:* The more nodes are involved in a request, the lower the chance for an inconsistency
- For example, Cassandra allows for using quorums

Write Quorums

- Number of nodes that acknowledge a write operation
- Replication factor (N): number of nodes involved in a replication
  - Often, a factor of 3 is enough
- Number of confirmed writes (W)
- You reach a majority when W > N/2
  - Strong consistency: W = N

Read Quorums

- Number of nodes that need to be contacted before getting the most current data
- Number of nodes contacted when reading (R)
- Examples:
  - If N = 3 and W = 2: contact at least two nodes
  - If N = 3 and W = 1: contact all three nodes
- You get strongly consistent reads if R + W > N

Learn about a DataFrame’s Lineage

Explain Plans

Explain Plans

- Spark offers the `explain()` command
  - Can be used on any DataFrame object
  - Shows the DataFrame's lineage/how Spark will execute the query
- Example: `flightdata.sort("count").explain()`
  → Resulting explain plan:

```
== Physical Plan ==
*(2) Sort [count#12 ASC NULLS FIRST], true, 0
     +- Exchange rangepartitioning(count#12 ASC NULLS FIRST, 200)
        +- *(1) FileScan csv [DEST_COUNTRY_NAME#10,…]…
```

- How to read an explain plan:
  - Top: end result
  - Bottom: source(s) of the data

Explain Plan – Example

```
flightdata.groupBy("DEST_COUNTRY_NAME").sum("count")\
.withColumnRenamed("sum(count)","destination_total")\
.sort(desc("destination_total")).limit(5).explain()
```

```
== Physical Plan ==
TakeOrderedAndProject(limit=5, orderBy=[destination_total#36L DESC NULLS LAST]...
    +- *(2) HashAggregate(keys=[DEST_COUNTRY_NAME#10], functions=[sum(...
        +- Exchange hashpartitioning(DEST_COUNTRY_NAME#10, 200)
            +- *(1) HashAggregate(keys=[DEST_COUNTRY_NAME#10], functions=[partial_sum(...
                +- *(1) FileScan csv [DEST_COUNTRY_NAME#10,count#12]
```

Now it’s your turn!

Homework Exercise

Exercise #1: CAP Theorem

- Why is MongoDB on the Consistency/Partitioning side of the CAP theorem?

Exercise #2: Explain Plans

- Write a piece of code using (a) SQL and (b) Spark that produces the following explain plan:

```
== Physical Plan ==
*(3) Sort [ORIGIN_COUNTRY_NAME#11 ASC NULLS FIRST], true, 0
+- Exchange rangepartitioning(ORIGIN_COUNTRY_NAME#11 ASC NULLS FIRST, 200)
   +- *(2) HashAggregate(keys=[ORIGIN_COUNTRY_NAME#11], functions=[sum(...
      +- Exchange hashpartitioning(ORIGIN_COUNTRY_NAME#11, 200)
         +- *(1) HashAggregate(keys=[ORIGIN_COUNTRY_NAME#11], functions=[partial_sum(...
            +- *(1) Project [ORIGIN_COUNTRY_NAME#11, count#12]
               +- *(1) Filter (isnotnull(DEST_COUNTRY_NAME#10) && (DEST_COUNTRY_NAME#10 =
United States))
                  +- *(1) FileScan csv [DEST_COUNTRY_NAME#10,ORIGIN_COUNTRY_NAME#11,count#12]
```

Thank you for your Attention!

WeST
People and Knowledge Networks