The Structured Query Language and Spark SQL

- What is Apache Spark
  - Scalable, efficient analysis of big data
- The Big Data Problem
  - Data growing faster than CPU speeds
  - Data growing faster than per-machine storage
- The opportunity
  - Cloud computing is game changer
  - Provides access to low-cost computing and storage
  - Costs decreasing every year
  - The challenge is programming the resources

Cluster Computing Challenges and the Map reduce Programming Paradigm

- What makes apache spark scalable
- Hardware for Big Data
  - Consumer-grade hardware
  - Easy to add capacity cheaper
  - But, requires complexity in software
- Problems with cheap hardware
  - Failures
  - Network
    - Much more latency
    - Network slower than storage
  - Uneven performance
- What's hard about cluster computing
  - How do we split work across machine
    - Hash Table
      - Each time we see a word look up word in the table if it appears increment by one or add word to the dictionary
    - If document is big run partitions or shards and run on multiple machines
      - Then aggregate the machines to a final machine
      - The problem is that all the results have to fit on one machine
      - Alternative is using map reduce
  - Map Reduce for sorting
    - What word is used the most
    - Example – all words that are less than x go to the first machine
- How do we deal with failures
  - Start another task
- How to deal with slow tasks
  - Terminate and start another task

Apache Spark: Technology Trends, Opportunity, and Advantages

- Datacenter Organization
  - Node placed in rack interconnect with top of rack switches which are interconnected to other top of rack switches
- Map Reduce: Distributed Execution
  - Each stage passes through harddrive

- Map Reduce: Iterative Jobs
  - Iterative jobs involves a lot of disk I/O for each repetition
- Apache Spark Motivation
  - Use map reduce for complex jobs, interactive queries and online processing involves lots of disk I/O
    - Interactive mining
      - Reading from the disk for each query
    - Stream processing
      - Reading from the disk for each job
- Use Memory instead of disk
  - Each iteration read from disk and writing from disk
- Spark and Map Reduce Differences

	Apache Hadoop Map Reduce	Apache Spark
Storage	Disk only	In-memory or on disk
Operations	Map and Reduce	Many transformation and actions, including Map and Reduce
Execution model	Batch	Batch, interactive, streaming
Languages	Java	Scala, Java, R, and Python

- Other Spark and Map Reduce Differences
  - Generalized Patterns for computation
  - Lazy evaluation of lineage graph
    - Can optimize, reduce wait states, pipeline
- In memory operation
- Spark performance optimizations
  - Catalyst optimization engine
    - Reduction in execution time
  - Project Tungsten off-heap memory management
    - 75% reduction in memory usage
- Catalyst: Shared Optimization and Execution
- Java Virtual Machine Object Overhead
  - "abcd"
    - native 4 bytes
    - java 48 bytes
- Project Tungsten's Compact Encoding

Structured Data and the Structured Query Language

- Review: Key Data Management Concepts

    - A data model is a collection of concepts for describing data
    - A schema is a description of a particular collection of data, using a given data model
    - A relational data model is the most used data model
        - Relation, a table with rows and columns
        - Every relation has a schema defining fields in columns
- Relational database
    - Relational database: a set of relations
    - Two parts to a Relation
        - Schema: specifies name of relation, plus each column's name and type
        - Instance: the actual data at a given time
            - #rows = cardinality
            - #fields = degree
- What is a database
    - Large organized collection of data
        - Transactions used to modify data
    - Models real word
        - Entities
        - Relationships
- SQL – A language for Relational DBs
    - SQL = Structured Query Language
    - Supported by Spark DataFrames (SparkSQL)
    - Some of the functionality SQL provides
        - Create, modify, delete relations
        - Add, modify, remove tuples
        - Specify queries to find tuples matching criteria
- Queries in SQL
    - Single-table queries are straightforward
    - To find all 18 year old students, we can write:
    - To find just names and logins
        - SELECT S.name, S.login
        - FROM Stduents S
        - WHERE S.age = 18
- Querying Multiple Relations
- Cross Join
    - Cartesian product of two tables (ExS)
- Where Clause
    - Choose matching rows using where clause
        - S.sid = E.sid
- Select Clause
    - Use the filtering
        - S.name, E.cid
- Equivalent SQL Join Notations
    - Explicit Join notation (preferred)
        - SELECT S.name, E.classid

        - FROM Students S INNER JOIN Enrolled E on S.sid=E.sid
        - FROM Students JOIN Enrolled E on S.sid=E.sid
    - Implicit join notation (deprecated)
        - SELECT S.name, E.cid
        - FROM Students S, Enrolled E
        - WHERE S.sid = E.sid
- SQL Types of Joins
    - Unmatched keys with inner join just drop the rows
- SQL Joins: Left Outer Join
    - Unmatched row appears in the results with a NULL value
- SQL Joins: Right Outer Join
    - Unmatched row appears in the results with a NULL value for the matching id

Spark SQL and Spark DataFrames

- Spark Joins
    - SparkSQL and Spark DataFrames support joins
    - Join(other, on, how)
        - Other – right side of the join
        - On –join column name, list of column (names), or join expression
        - how – inner, outer, left, right_outer, left_semi
- Spark Join Examples
    - Inner Join – X.join(Y,cols)
        - Return DF of rows with matching cols in both X and Y