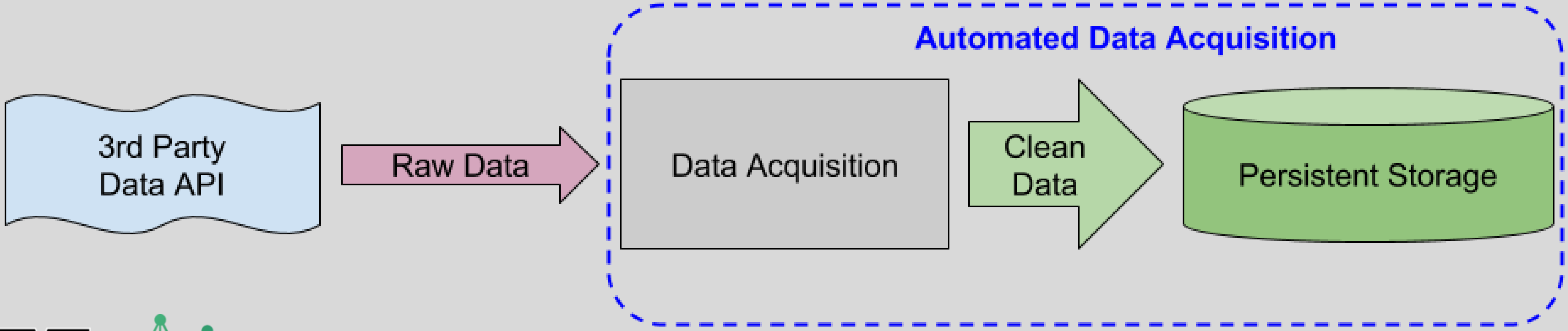
Introduction to Data Science and Analytics

SQL and

Big Data Ecosystems

Structuring data into Databases

- Databases provide persistent structured storage
- After acquisition and reshaping, data often persisted
- Key for data reusability for analytics
- Often automated



Learning Activities

Preliminary examples of storing data into a data base

Practice querying to inspect data

Much more work in the Database and Analytics course

- Module 2: Database Design and Loading
- Module 3: Data Engineering, ETL Pipelines

SQL Analytics

*The **combination** of*

- column functions,
- joins,
- aggregates,
- nested queries, and
- other advanced **SELECT** syntax

enables analytical data access

What is Big Data?

Four characteristics

- Volume (width and/or depth)
- Velocity
- Variety
- Veracity

These Vs lead to fifth V, **Value**

Big Data Require Big Systems

- Big Data technologies typically leverage horizontally scalable systems
- Always Trade-Offs on Cost versus Performance / Capacity
  - Hadoop vs Spark
  - MongoDB vs Redis

Big Data Ecosystems

Usually Distributed Systems

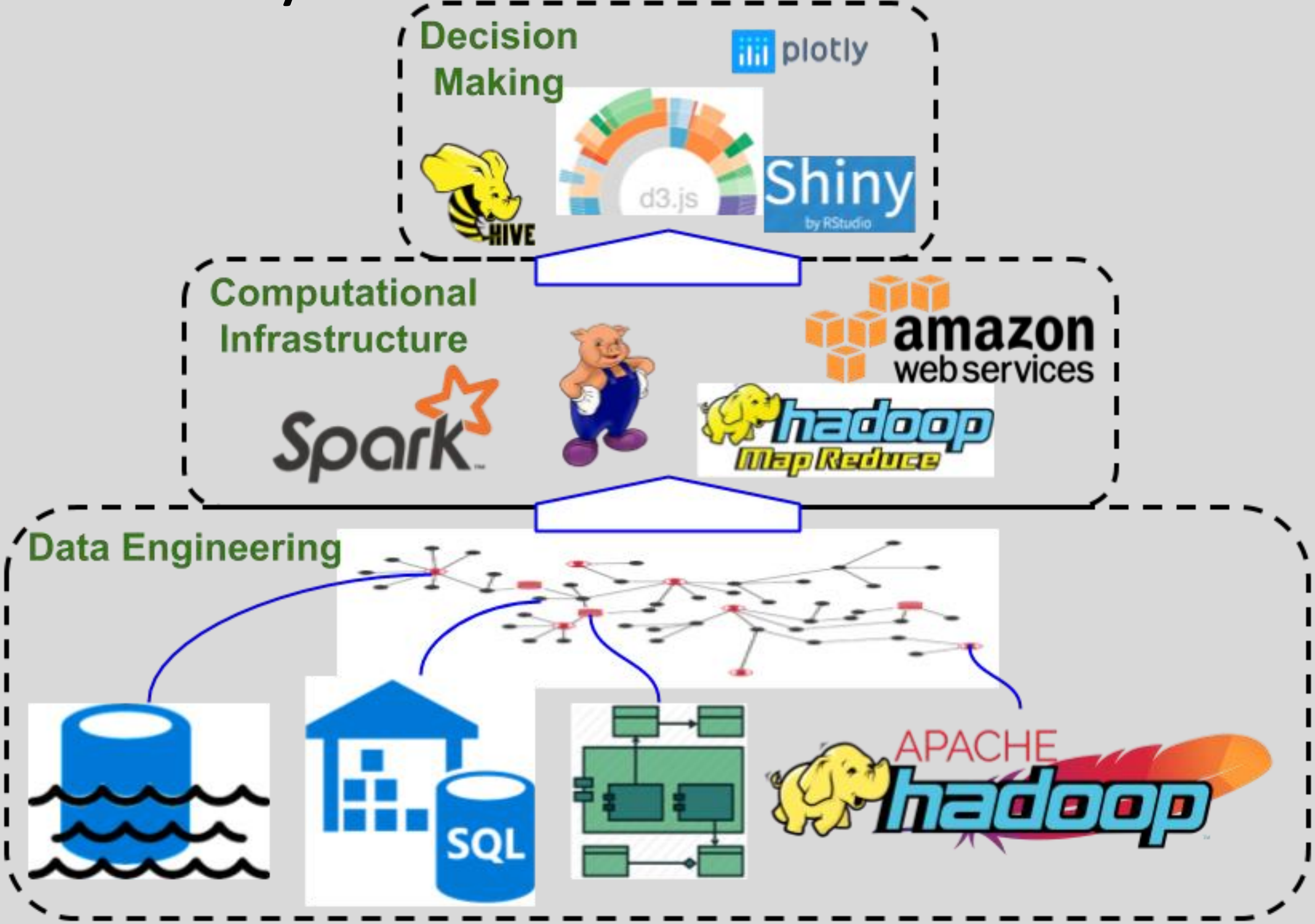
Composition can be a mix of:

- Commodity Components
- HPC components

Often significant levels of abstraction to support diverse workloads

- Analytics
- Transactions
- Machine Learning

Big Data Ecosystems



Big Data Ecosystems

Common Systems

- Data Processing
- Data Lakes
- Decision Support
- Operationalized Machine Learning
  - Example: Fraud Detection
  - Example: Automated Image/Video Analysis

Data Lakes vs. Data Warehouses

- **Data Lakes** are the unstructured, raw data from all sources
  - Sensors, social media, measurements, etc.
- **Data Warehouses** are the highly structured, aggregated data from all operational data stores
  - Relational database aggregators
    - ERP, Sales, CRM, etc.

DATA SCIENCE
& ANALYTICS

Big Data Ecosystems : Hadoop

Hadoop is an Apache Foundation, Open Source project for distributed computing

Core Components include:

- Hadoop Distributed File System (HDFS)
- Hadoop Map Reduce – Compute
- Hadoop Yarn – Job/Task Scheduling

HDFS – Design Goals

- Designed to store a very large amount of data (PB)
- Data is to be spread across a large number of commodity computers
- Data storage reliable: redundancy even when node failure
- Horizontal scaling to accommodate computational load

HDFS - Design Trade-offs / Assumptions

Applications assumed to be dominated by long sequential file reads

Non-transactional

- e.g., write once, read many

No local caching of file data, must re-read

- this is why we have *Spark* on top

HDFS - Design Trade-offs / Assumptions

Individual nodes in the system are expected to fail

- Permanent fail
- Intermittent unavailability

Data replication used to facilitate fault tolerance

DATA SCIENCE
& ANALYTICS

Map-Reduce

Map-Reduce is a programming paradigm that is best suited for “divide and conquer” processing

- Not natively ideal for general purpose computation / algorithms
- Computation should be a share-nothing
  - Node X should need no data/results from Node Y for optimal performance
  - Performance / Scalability degrades with cross-node computational dependencies

Map-Reduce → SQL ?

Map-Reduce not well suited for Analytical SQL queries

- Hive : SQL layer / engine on top of Hadoop
- Apache Tez and Spark SQL aimed at accelerating SQL style access to HDFS stored data
- Impala and Drill additional technologies

Introduction to Data Science and Analytics

SQL and

Big Data Ecosystems

DATA SCIENCE
& ANALYTICS