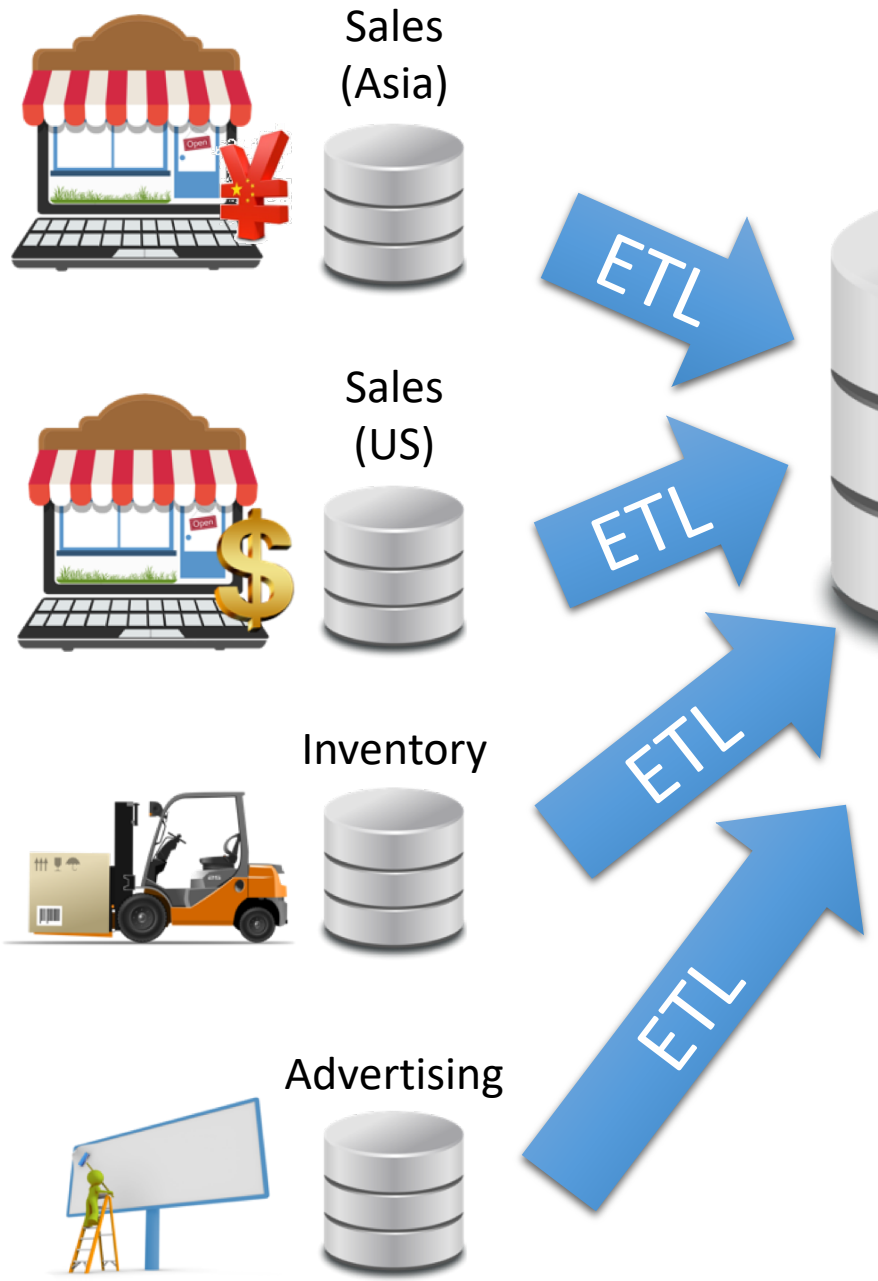
What is Data Warehouse like in the Big Data Era?

Data Warehouse

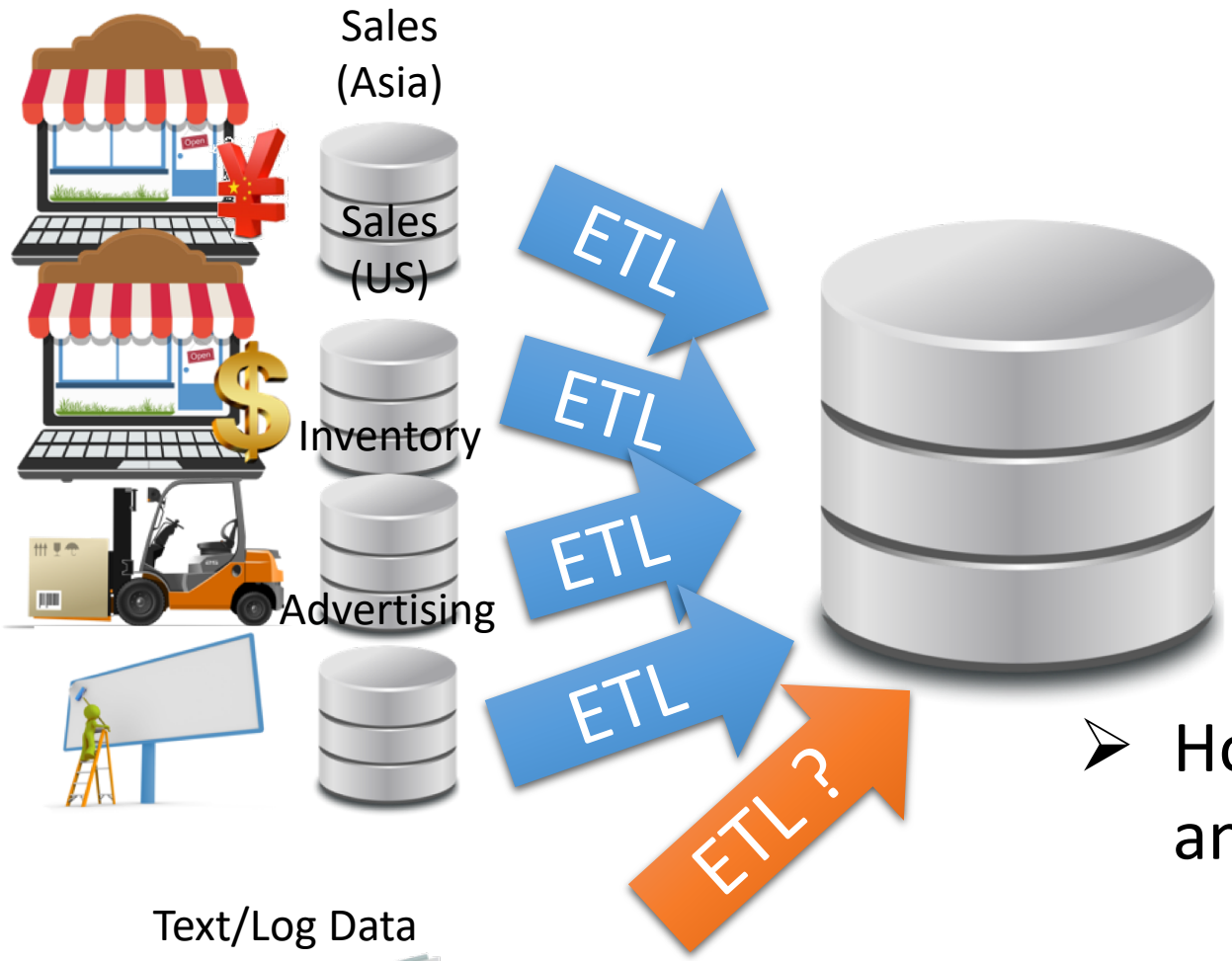


Collects and organizes historical data from multiple sources

So far …

- Star Schemas
- Data cubes
- OLAP Queries

Data Warehouse



Collects and organizes historical data from multiple sources

- How do we deal with semi-structured and unstructured data?
- Do we really want to force a schema on load?

Data Warehouse

Collects and organizes historical data from multiple sources

How do we **clean** and **organize** this data?

Depends on use …

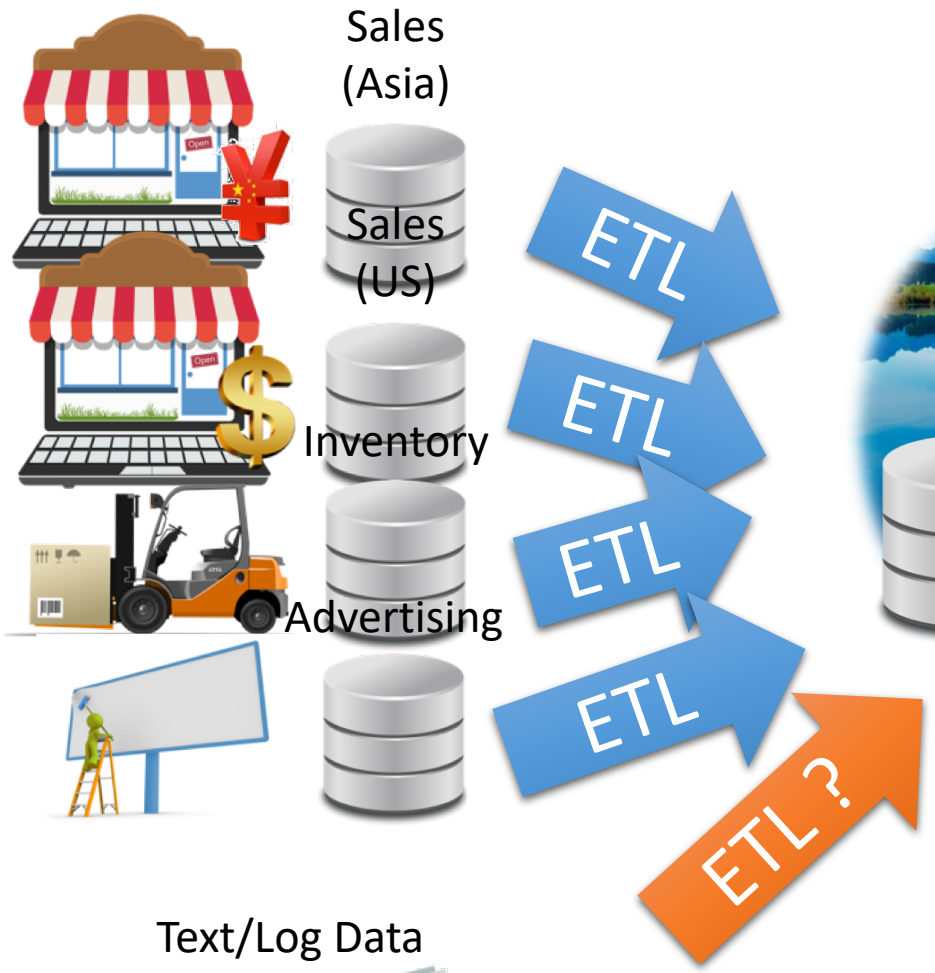
ETL ?

How do we **load** and **process** this data in a relational system?

Depends on use …
Can be difficult …
Requires thought …

Text/Log Data

Photos & Videos



Data Lake *

Store a copy of all the data

- in one place
- in its original "natural" form

Enable data consumers to choose how to transform and use data.

- *Schema on Read*

Enabled by new Tools:
Map-Reduce & Distributed Filesystems

What could go wrong?

*Still being defined…[Buzzword Disclaimer]

The Dark Side of Data Lakes

- Cultural shift: *Curate* → *Save Everything!*
  - Noise begins to dominate signal
- Limited data governance and planning

  Example: *hdfs://important/joseph_big_file3.csv_with_json*
  - **What** does it contain?
  - **When** and **who** created it?
- No cleaning and verification → lots of dirty data
- New tools are more complex and old tools no longer work

Enter the data scientist

A Brighter Future for Data Lakes

Enter the data scientist

- Data scientists bring new skills
  - Distributed data processing and cleaning
  - Machine learning, computer vision, and statistical sampling
- Technologies are improving
  - SQL over large files
  - Self describing file formats & catalog managers
- Organizations are evolving
  - Tracking data usage and file permissions
  - New job title: data engineers

What you've learnt?

- Why Data Warehouse?
- Why Data Lake?
- Why Data Scientists?
- Why Data Engineers?

How to improve query performance?

10 MB/s

SELECT SUM(Sales)
FROM Table
WHERE Country = ’Canada’

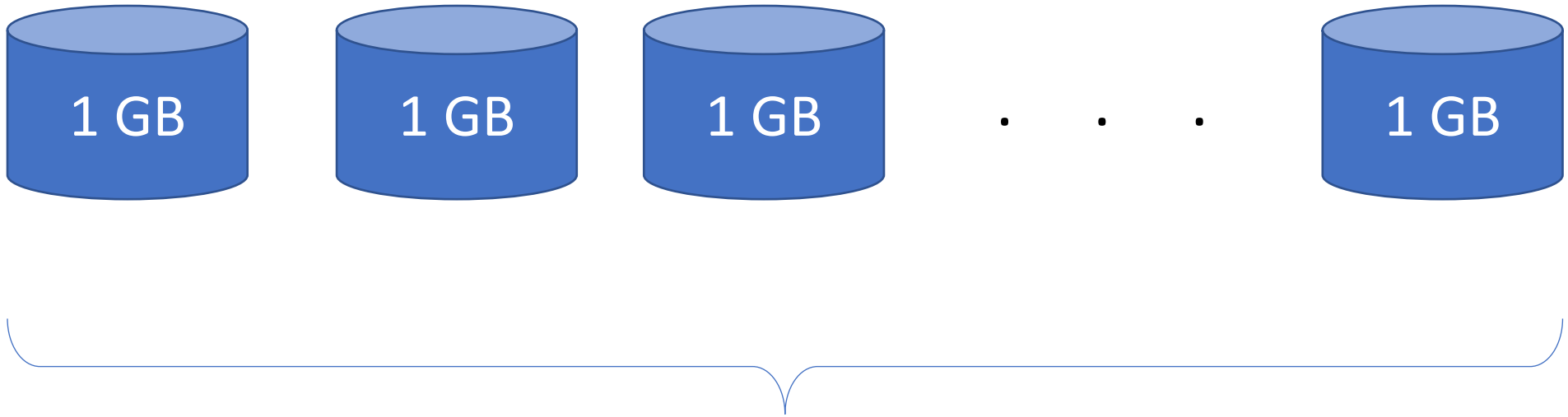
1.2 Day

Idea 1: Pre-computation

- **SELECT** SUM(Sales) **FROM** Table **WHERE** Country = “Canada” **1.2 Million**
- **SELECT** SUM(Sales) **FROM** Table **WHERE** Country = “USA” **2.4 Million**
- **SELECT** SUM(Sales) **FROM** Table **WHERE** Country = “China” **1.8 Million**
- …

1. Which query should be precomputed?
2. What if data is updated?
3. How to use precomputed query results?

Idea 2: Parallel Database



1000 machines

SELECT SUM(Sales)
FROM Table
WHERE Country = 'Canada'

1.7 mins

David DeWitt

David DeWitt

The **Gamma database** machine project
DJ DeWitt, S Ghandeharizadeh… - … on Knowledge and …, 1990 - ieeexplore.ieee.org
… sign and implementation of highly **parallel database** ma- chines. In a number of ways, the design of **Gamma** is based on what we learned from our earlier **database** ma- chine DIRECT [IO] … **Gamma** is similar to a number of other active **parallel database** machine efforts …
☆ 99 Cited by 883 Related articles All 41 versions

The "DeWitt Clause" [edit]

Several commercial database vendors include an end-user license agreement provision, known as the *DeWitt Clause*, that prohibits researchers and scientists from explicitly using the names of their systems in academic papers.[4][5]

MapReduce: A major step backwards

By David DeWitt on January 17, 2008 4:20 PM | Permalink | Comments (44) | TrackBacks (1)
[Note: Although the system attributes this post to a single author, it was written by David J. DeWitt and Michael Stonebraker]

Parallel DBMSs

- How to evaluate a parallel DBMS?
- How to architect a parallel DBMS?
- How to partition data in a parallel DBMS?

Parallel DBMSs

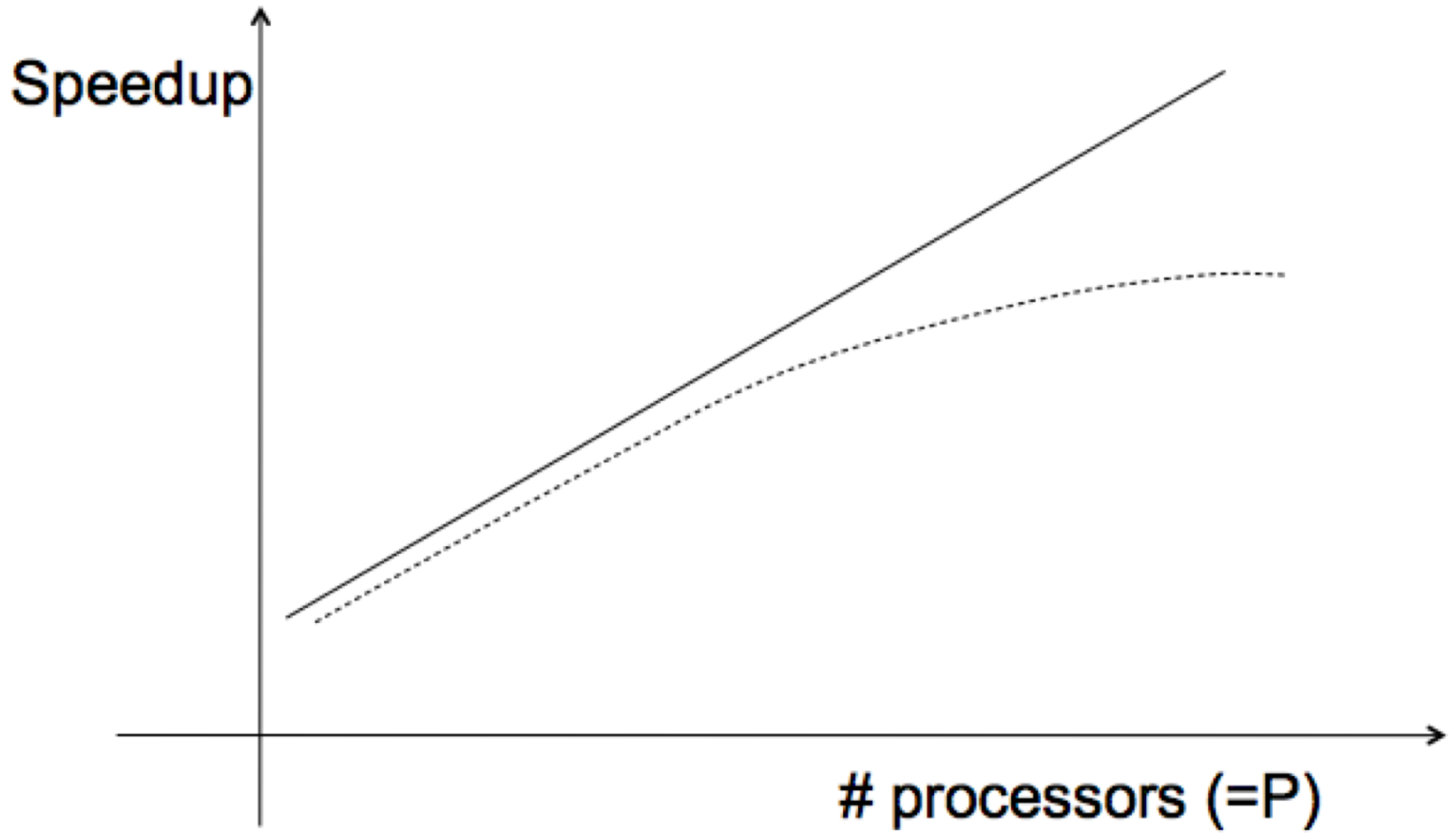
- **How to evaluate a parallel DBMS?**
- How to architect a parallel DBMS?
- How to partition data in a parallel DBMS?

Performance Metrics for Parallel DBMSs

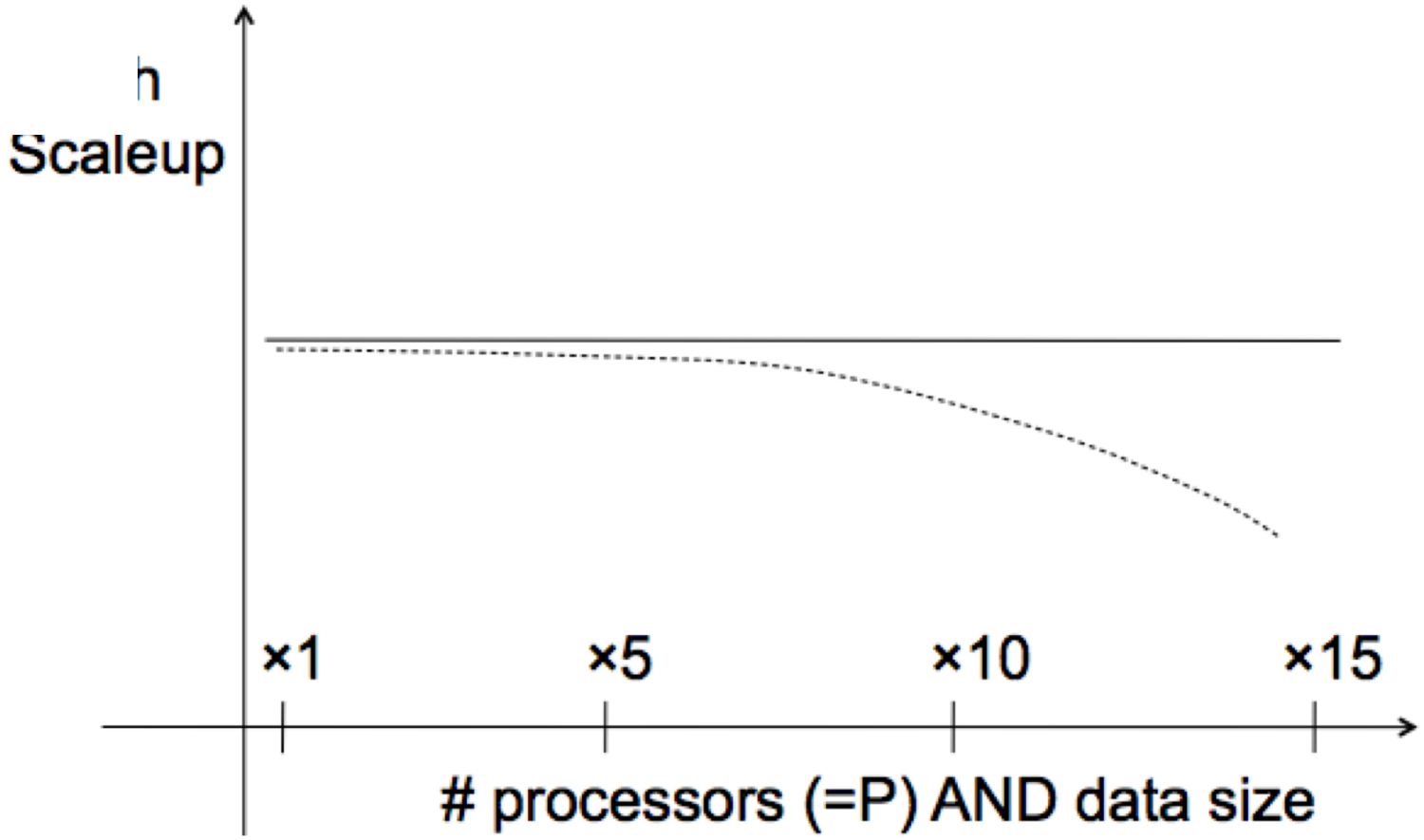
- **Speedup**
  - ✓ More processors → Higher speed

- **Scaleup**
  - ✓More processors → Can process more data

Linear v.s. Non-linear Speedup



Linear v.s. Non-linear Scaleup



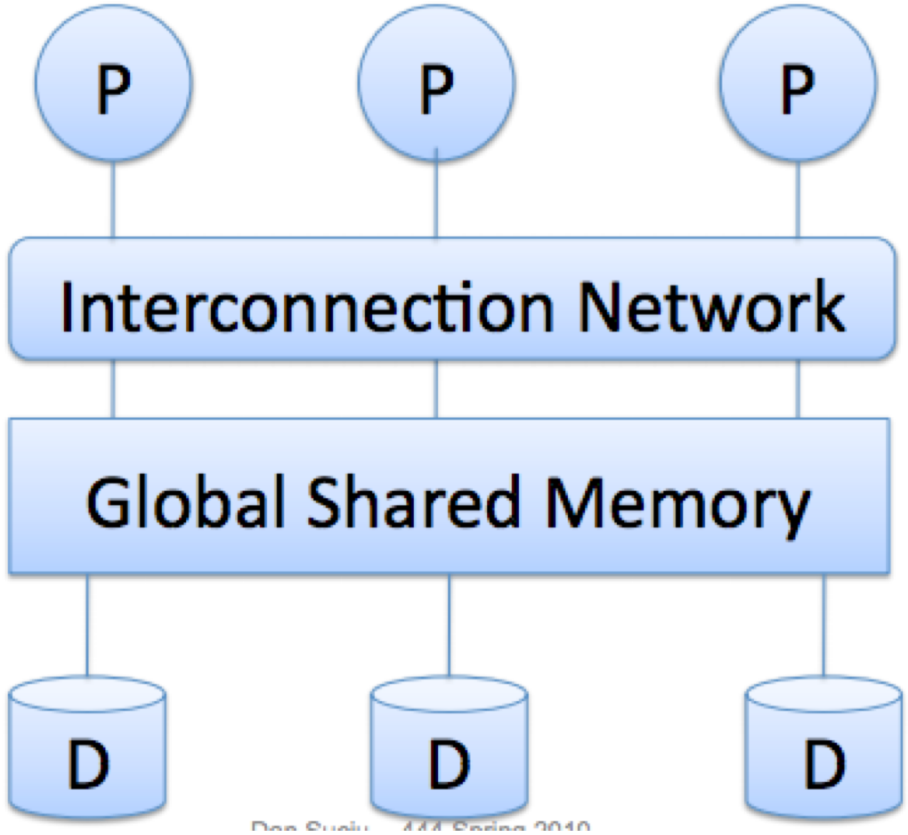
Parallel DBMSs

- How to evaluate a parallel DBMS?
- **How to architect a parallel DBMS?**
- How to partition data in a parallel DBMS?

Three Architectures

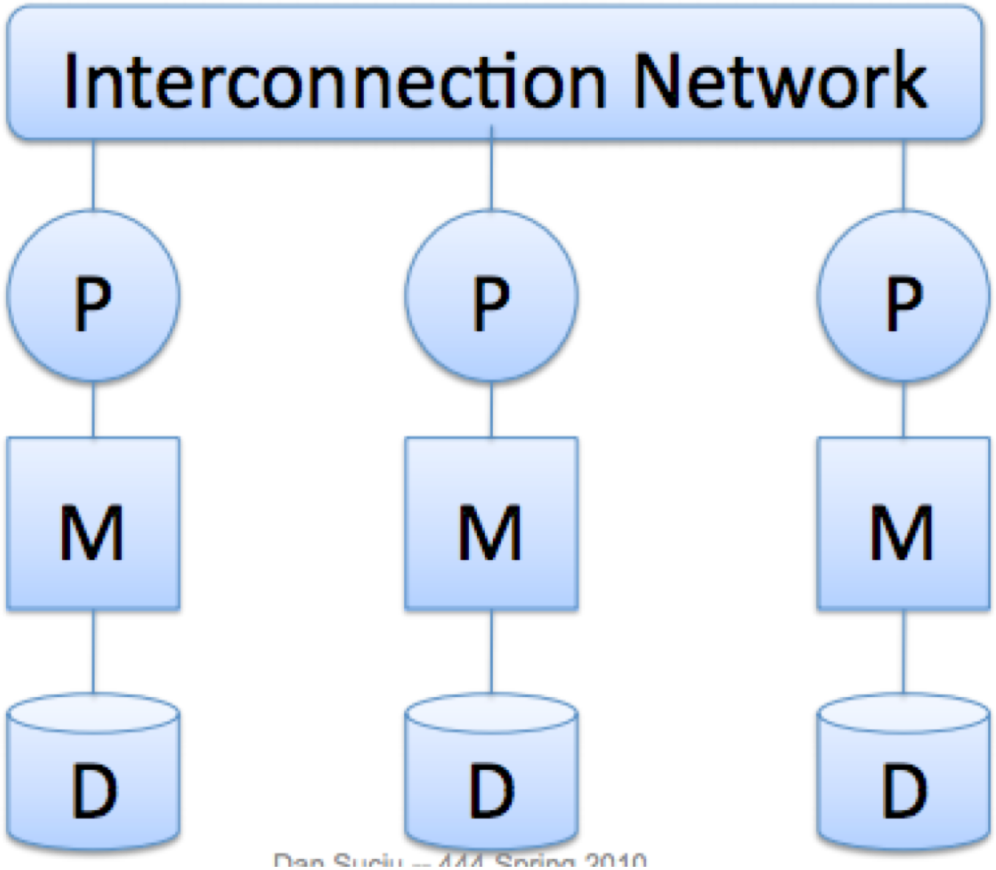
- **Shared Memory**
- **Shared Nothing**
- **Shared Disk**

Shared Memory



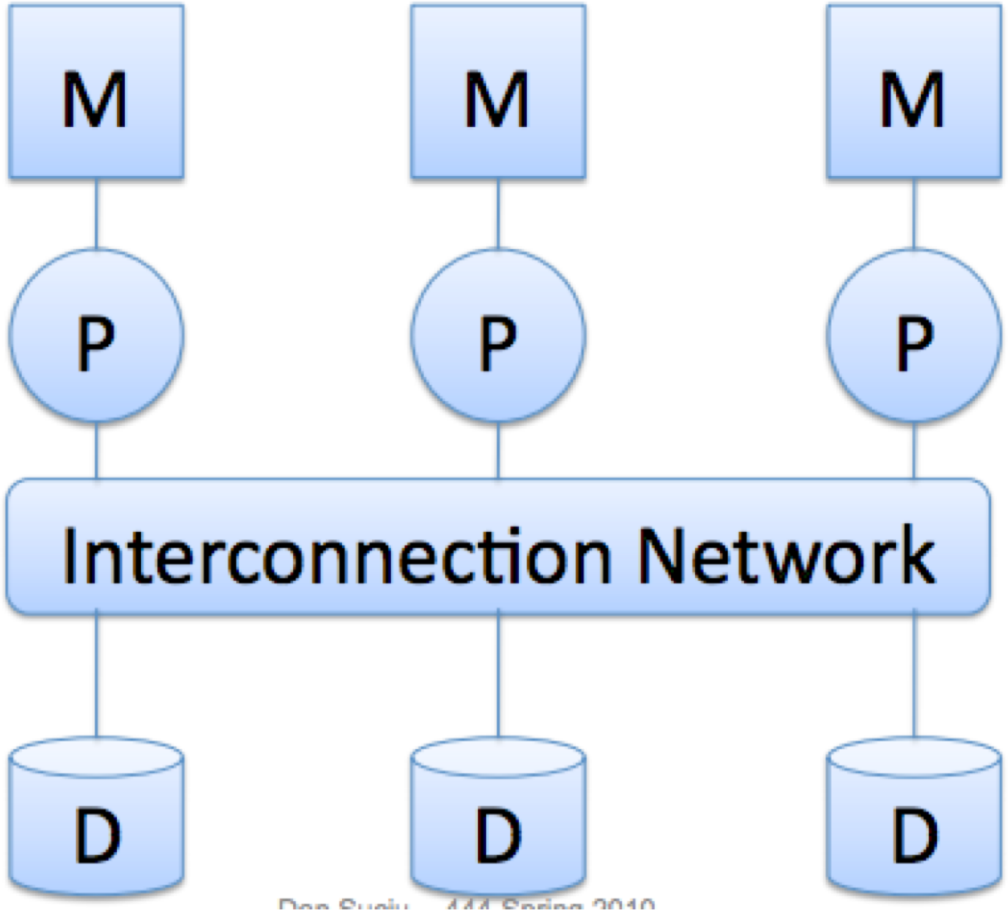
GPU

Shared Nothing



Parallel DBMSs, MapReduce, Spark

Shared Disk



Azure Data Warehouse

Three Architectures

- **Shared Memory**
- **Shared Nothing**

Computation vs. Communication Trade-offs

- **Shared Disk**

Economic Consideration

Parallel DBMSs

- How to evaluate a parallel DBMS?
- How to architect a parallel DBMS?
- **How to partition data in a parallel DBMS?**

Horizontal Data Partitioning

- **Round Robin**
  - ✓☺ Load Balancing
  - ✓☹ Bad Query Performance
- **Range Partitioning**
  - ✓☺ Good for range/point queries
  - ✓☹ Data Skew (i.e., Bad Load balancing)
- **Hash Partitioning**
  - ✓☺ Good for point queries
  - ✓☹ Hard to answer range queries

Summary

- **How to improve query performance?**
  - ✓ Precomputation
  - ✓ Parallelism

- **Parallel DBMSs**
  - ✓ Evaluation metrics: Speedup and Scaleup
  - ✓ Architecture: Shared-memory, shared-nothing, shared-disk
  - ✓ Data Partition: Round Robin, Range Partitioning, Hash Partitioning

Sources

- Dan Suciu's CSE 444 slides, Spring 2010.
- UC Berkeley DS100 Fall 2017