CS2102
Database Systems

Slides adapted from Prof. Chan Chee Yong

LECTURE 11

MODERN DBMS

- Distributed DBMS
- OLAP
  - Multidimensional aggregation
- NoSQL
  - Graph database
- Future
  - NewSQL
  - Database-as-a-Service

Overview

- Distributed DBMS
- OLAP
  - Multidimensional aggregation
- NoSQL
  - Graph database
- Future
  - NewSQL
  - Database-as-a-Service

Distributed DBMS

Distributed DBMS

History

- Targeted to support the organizational structure of distributed enterprises
  - 1976: SDD-1 (*Computer Corporation of America*)
  - 1977: Distributed INGRES (*U.C. Berkeley*)
  - 1981: R* (*IBM Research*)

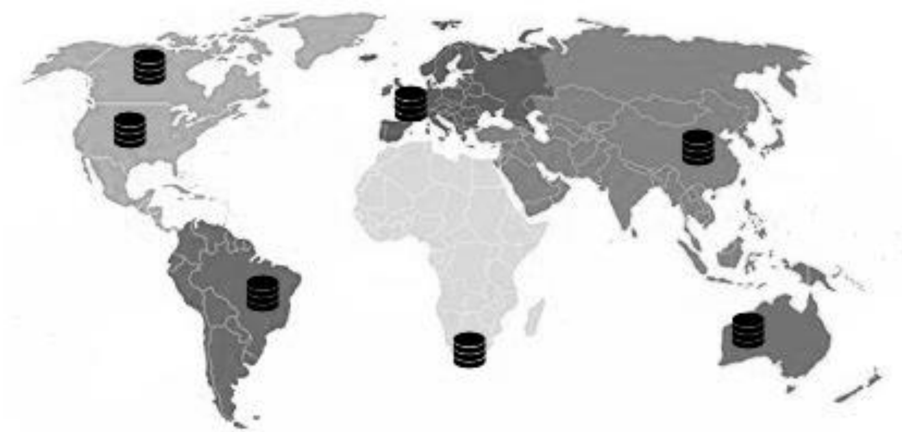
Distributed DBMS

Features

- Data is physically stored across multiple sites
  - But form a single logical database
  - Query and update feels like it is one database

Why

- Enterprise is often distributed
- Database recovery

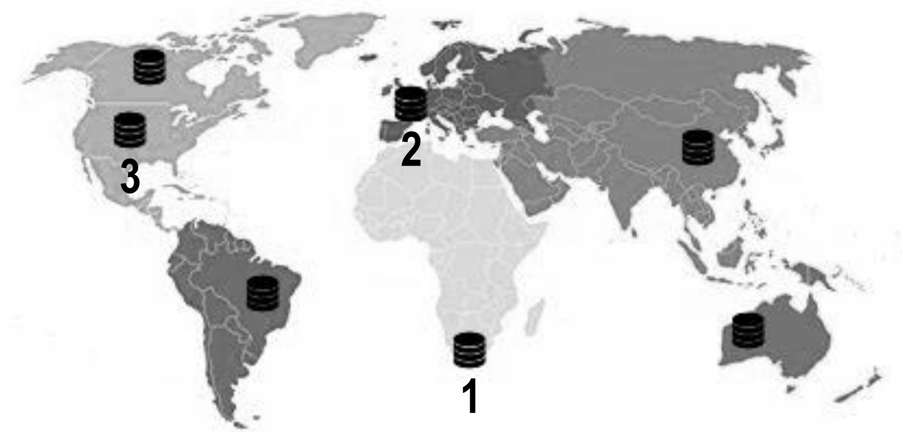


Distributed DBMS

Design

- Replication
  - Is table T available in more than one location?
    - Non-replicated
    - Partially replicated
    - Fully replicated

Loc	T_1	T_2	T_3	T_4	T_5
1	X			X	
2			X		
3		X			X



Distributed DBMS

Design

- Replication
  - Is table T available in more than one location?
    - Non-replicated
    - Partially replicated
    - Fully replicated

Loc	T_1	T_2	T_3	T_4	T_5
1	X			X	X
2	X		X		
3		X	X		X

Distributed DBMS

Design

- Replication
  - Is table T available in more than one location?
    - Non-replicated
    - Partially replicated
    - Fully replicated

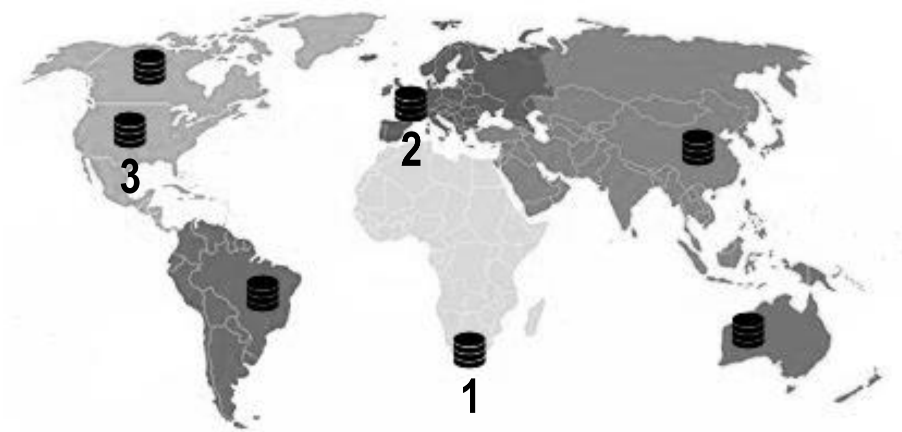
Loc	T_1	T_2	T_3	T_4	T_5
1	X	X	X	X	X
2	X	X	X	X	X
3	X	X	X	X	X

Distributed DBMS

Design

- Fragmentation
  - For a single table T, is all its data in a one location?
    - Non-fragmented
    - Fragmented

Loc	T_1	T_2	T_3	T_4	T_5
1	T_1			T_4	
2		T_2	T_3	T_4	
3		T_2			T_5

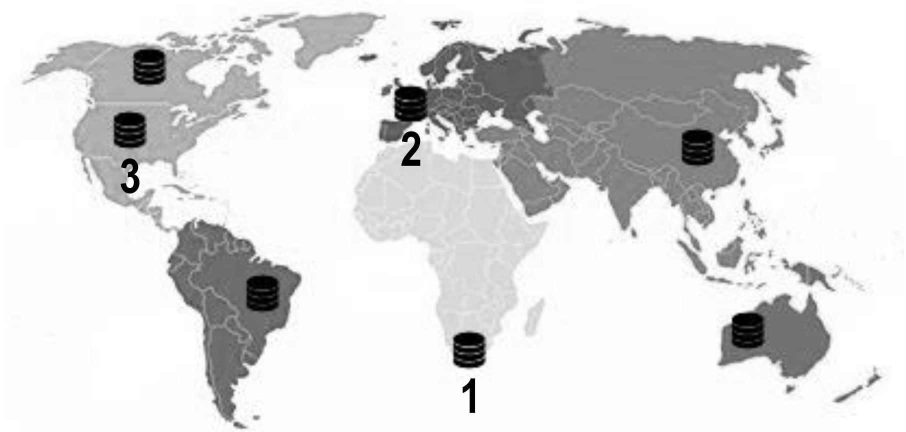


Distributed DBMS

Design

- Fragmentation
  - For a single table T, is all its data in a one location?
    - Non-fragmented
    - Fragmented

Loc	T_1	T_2	T_3	T_4	T_5
1	T_1^1			T_4	T_5^2
2	T_1^2	T_2^1	T_3		T_5^3
3		T_2^2			T_5^1



Distributed DBMS

Design

- Replication & fragmentation can be mixed

Loc	T_1	T_2	T_3	T_4	T_5
1	T_1^1 T_1^2			T_4	T_5^1 T_5^2
2	T_1^2	T_2^1	T_3		T_5^1 T_5^3
3	T_1^1	T_2^2			T_5^1

Distributed DBMS

Consistency? Availability? Partition tolerance?

- **Consistency:** *every read receives the most recent write or an error*
- **Availability:** *every request receives a (non-error) response, without guarantee that it contains the most recent write*
- **Partition tolerance:** *the system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes*

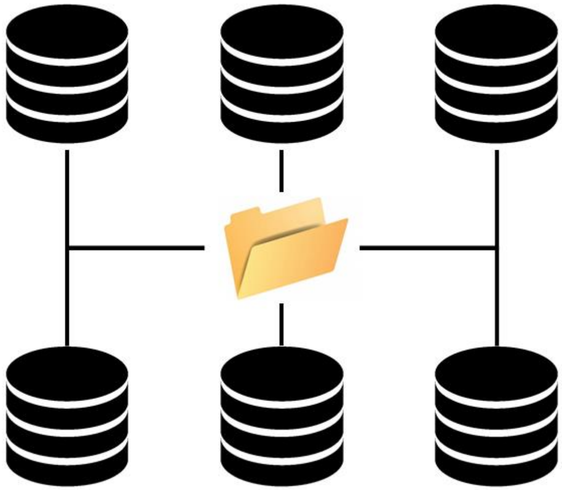
Parallel DBMS

Motivation

- Advantages of distributed DBMS without the disadvantages of distributed DBMS
- Use of shared file system across multiple machine

Main advantage

- High performance
- High availability



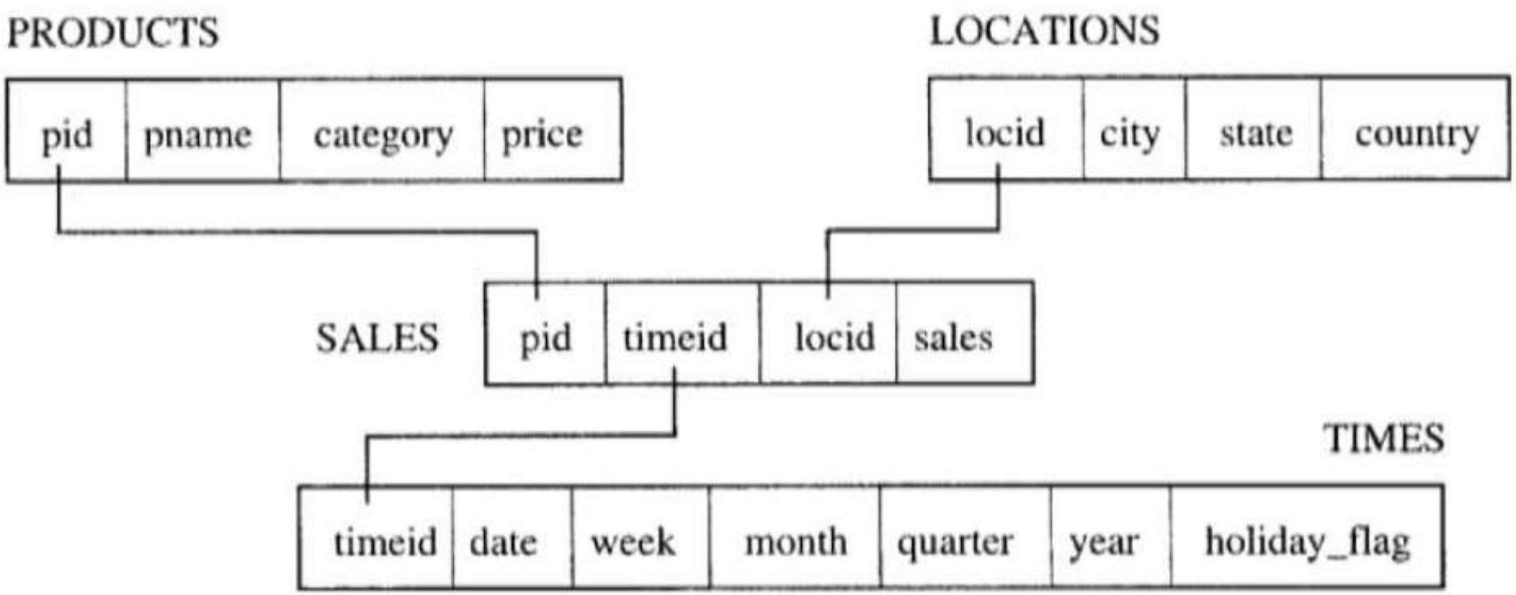
- Distributed DBMS
- OLAP
  - Multidimensional aggregation
- NoSQL
  - Graph database
- Future
  - NewSQL
  - Database-as-a-Service

OLAP

OLAP

Star schema

- Data is modeled using a fact table and dimensions tables



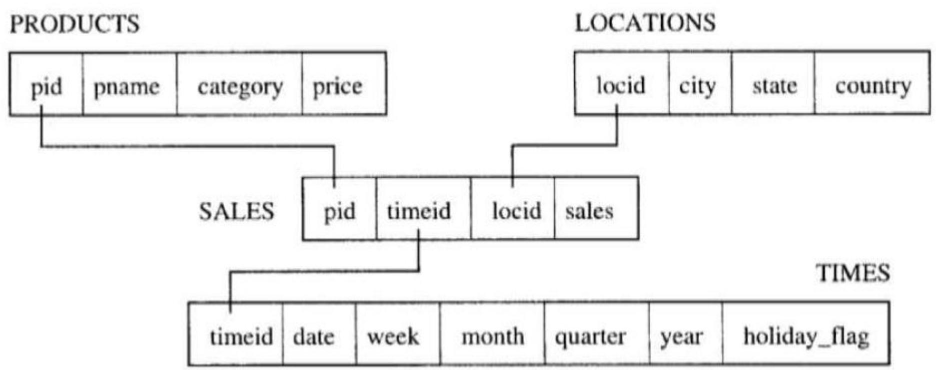
(Ramakrishnan & Gehrke, 2003)

OLAP

Multidimensional aggregation

- Total sales
  - for each product?
    - and each time?
      - and each location?

```
SELECT P.category,
       T.year,
       L.city,
       SUM(S.sales)
FROM Sales S
  JOIN Products P on ...
  JOIN Times T on ...
  JOIN Locations L on ...
GROUP BY P.category,
         T.year,
         L.city
```

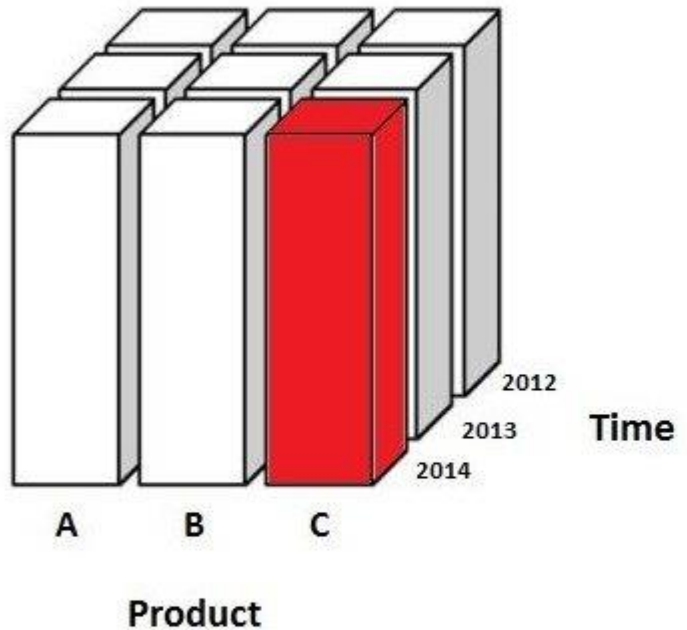


(Ramakrishnan & Gehrke, 2003)

OLAP

Multidimensional aggregation

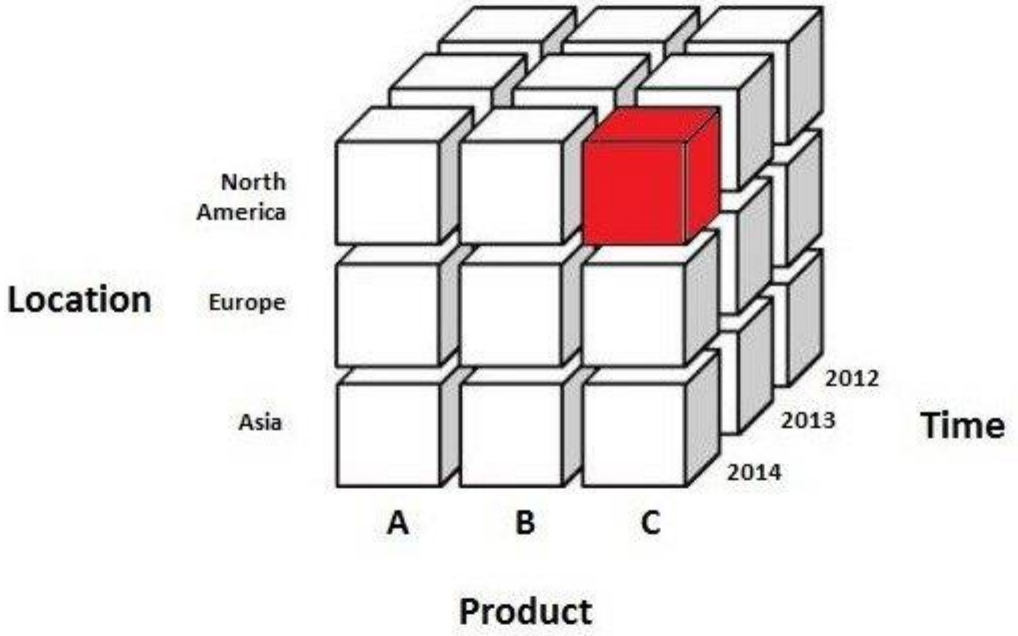
- Total sales
  - for each product?
    - and each time?
      - and each location?



OLAP

Multidimensional aggregation

- Total sales
  - for each product?
    - and each time?
      - and each location?

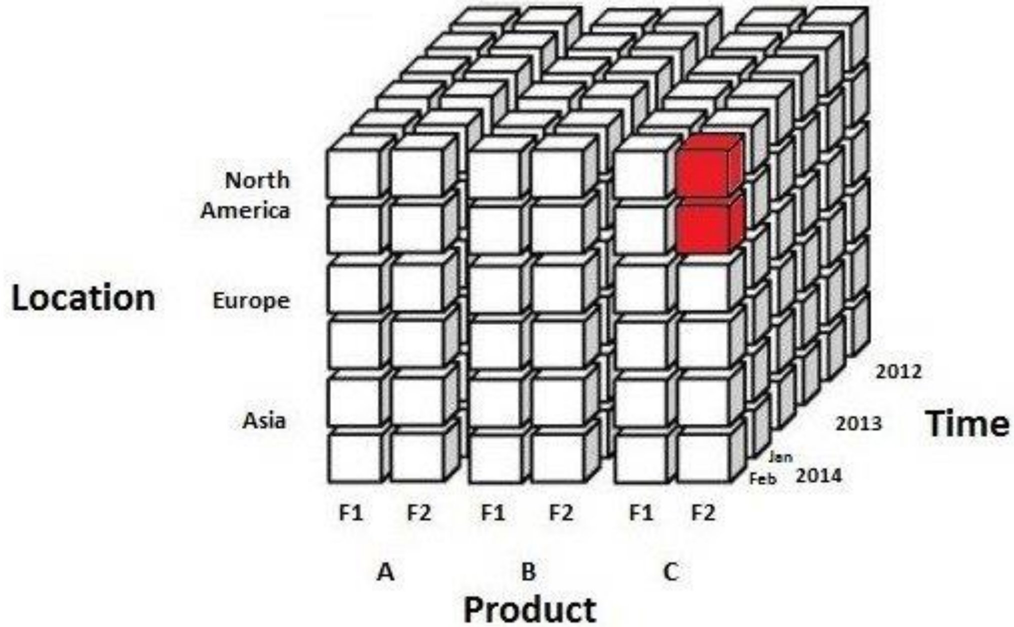


https://openit.com/faster-analysis-with-olap/

OLAP

Multidimensional aggregation

- Total sales
  - for each product?
    - and each time?
      - and each location?
        - and?

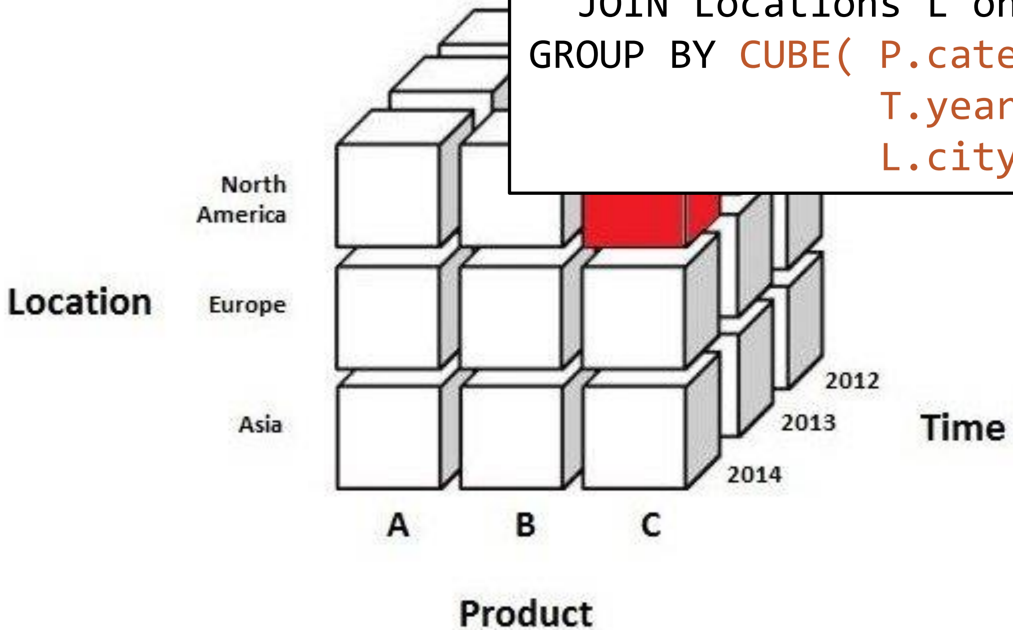


OLAP

Multidimensional aggregation

- Total sales
  - for each product?
    - and each time?
      - and each location?

```
SELECT P.category,
       T.year,
       L.city,
       SUM(S.sales)
FROM Sales S
  JOIN Products P on ...
  JOIN Times T on ...
  JOIN Locations L on ...
GROUP BY CUBE( P.category,
               T.year,
               L.city )
```



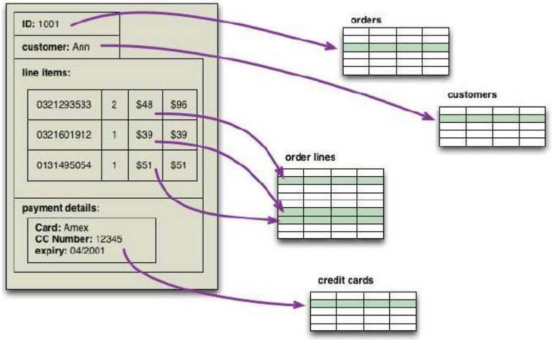
- Distributed DBMS
- OLAP
  - Multidimensional aggregation
- NoSQL
  - Graph database
- Future
  - NewSQL
  - Database-as-a-Service

NoSQL

NoSQL

Not only SQL

- Early NoSQL systems
  - Google's Bigtable; Amazon's Dynamo; Yahoo!'s PNUTS
- Data models
  - Key-value
  - Document
  - Graphs
  - etc...



(Martin Fowler, 2012)

- Features?
  - Schema-less data
  - Simple access API (put, get, and delete)
    - Instead of query language
  - Limited/no ACID transactional support
  - Weak consistency for replicated data

Graph database system

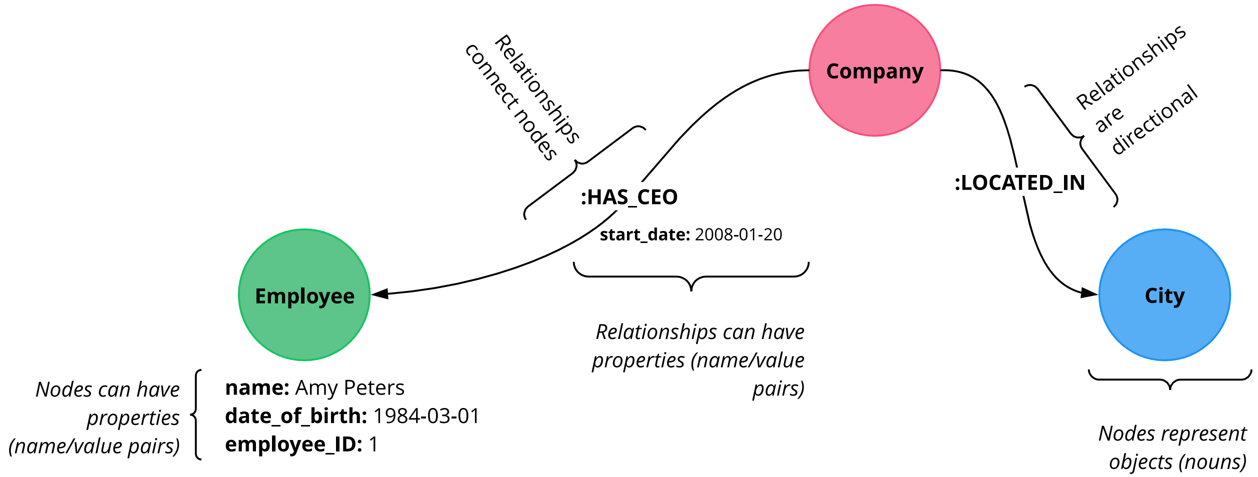
Introduction

- Based on different graph data models:
  - **Property graphs**
    - *Systems:* JanusGraph, Neo4J, etc
  - **RDF graphs**
    - RDF = resource description framework
    - Data stores known as triplestores/semantic graph databases
      - Store data as (subject, predicate, object) triples
    - Query language: SPARQL
    - Supports RDF schema (RDFS) & web ontology language (OWL) inference
    - *Systems:* AllegroGraph, GraphDB, etc
  - **Hypergraphs**
    - *Systems:* HyperGraphDB, Microsoft Graph Engine, etc

Property graph data model

Model

- Nodes represent entities
  - Each node has at least one label and possibly properties
- Directed edges represent relationships between entities
  - Each relationship has a type and possibly properties
- Each property is a key-value pair

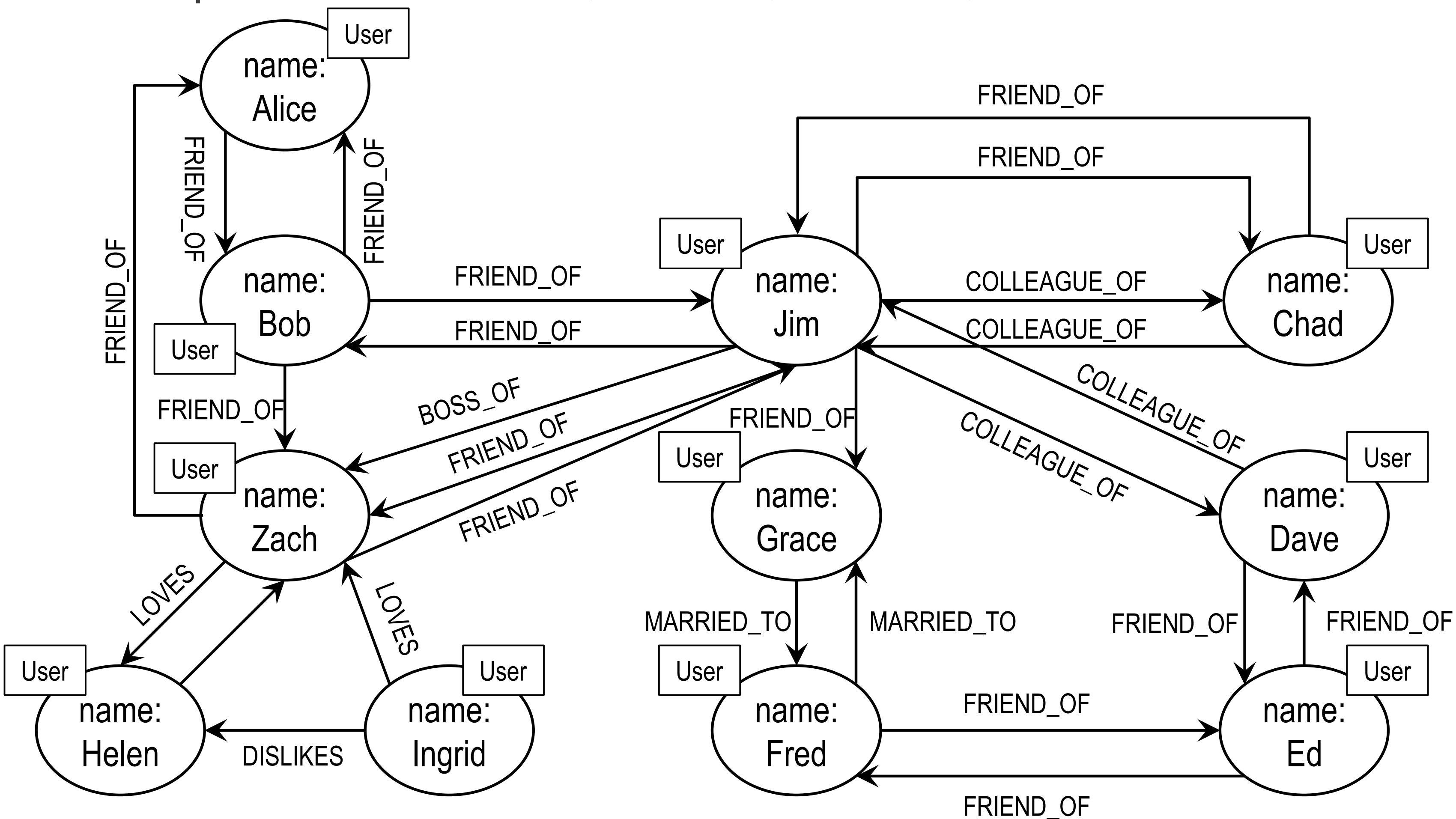


https://neo4j-contrib.github.io/developer-resources/get-started/graph-database/

Property graph data model

Example

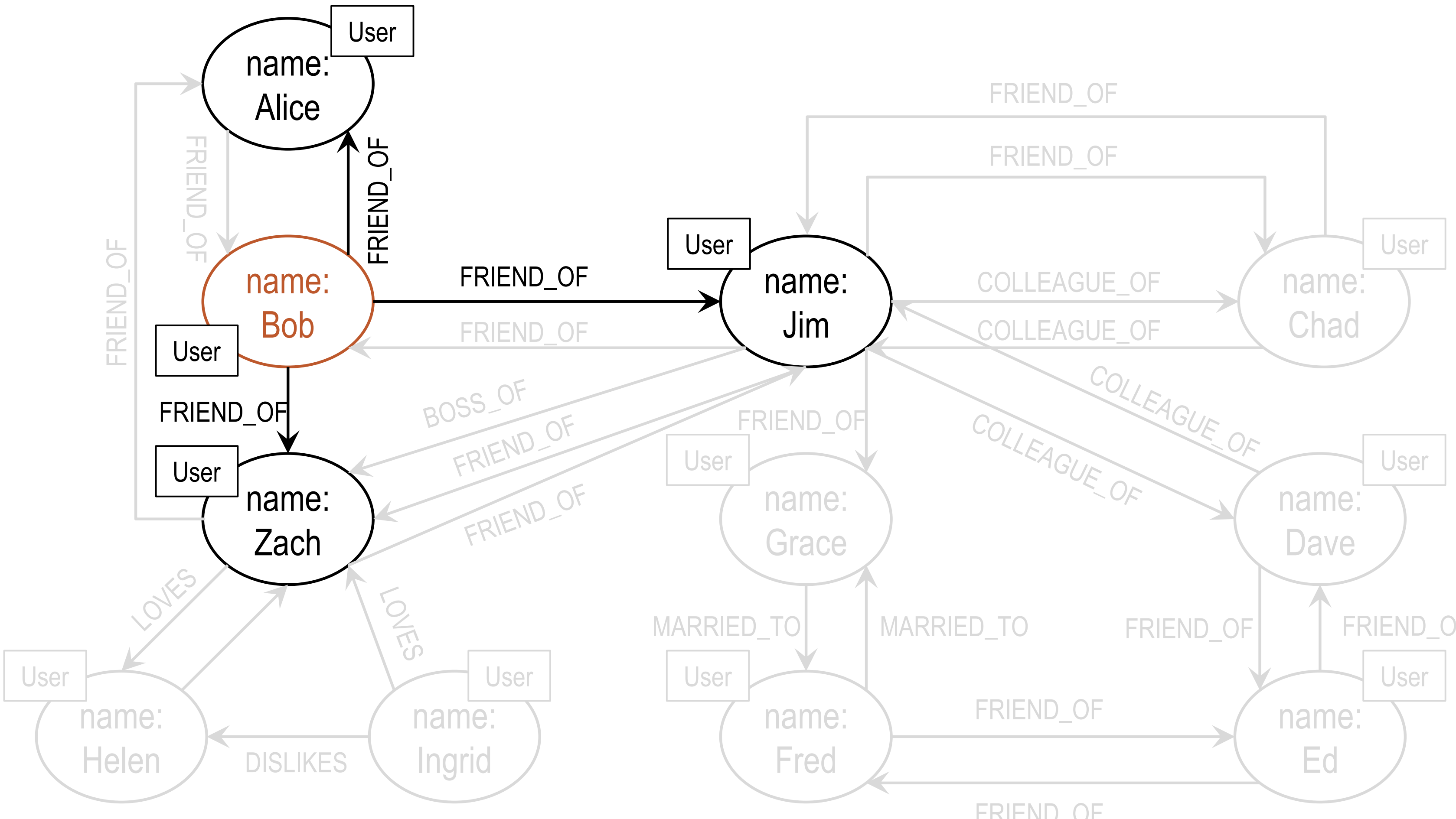
- Adapted from Robinson, Webber, & Eifrem, 2015



Property graph data model

Query

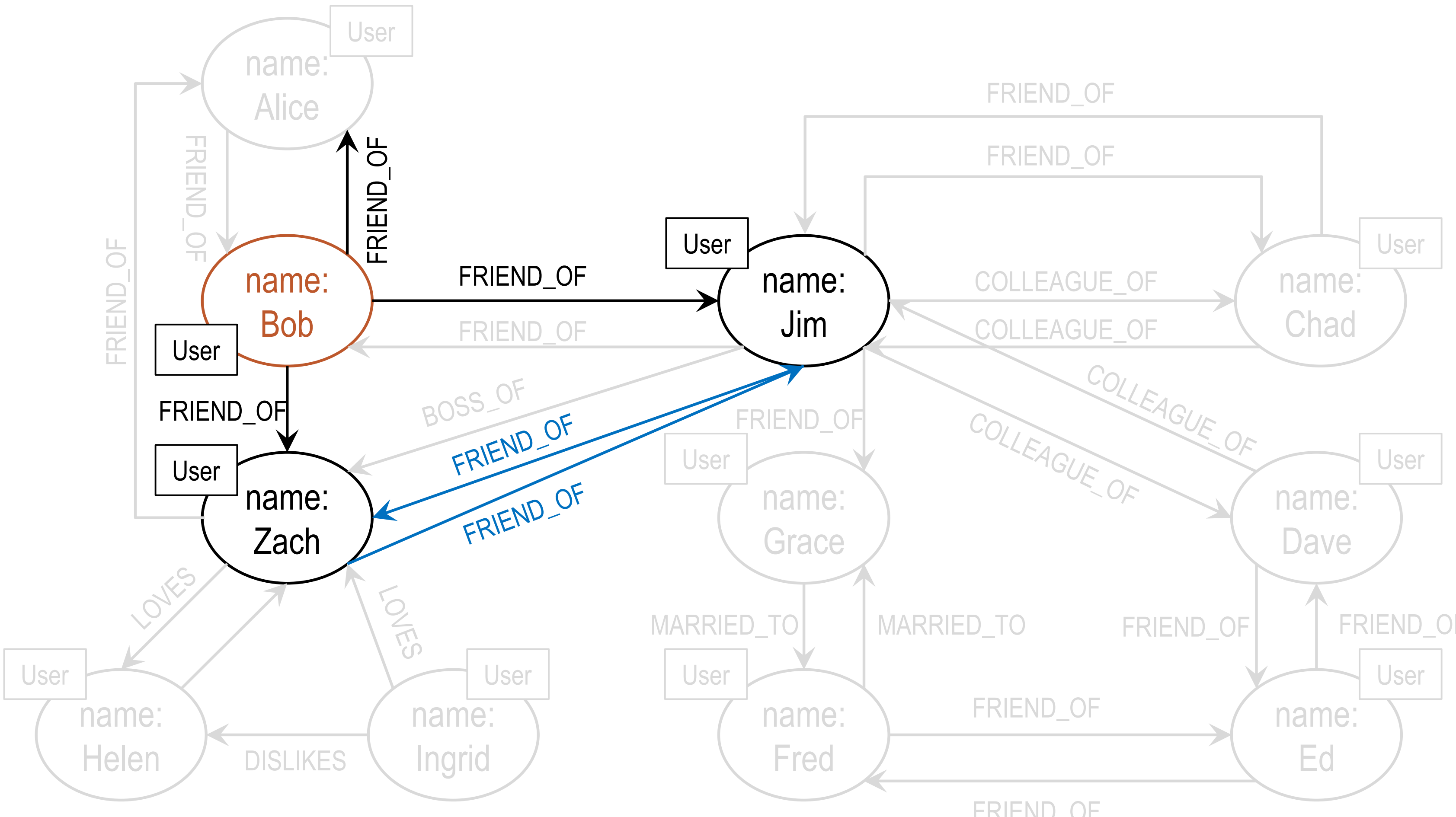
- Find all friends of Bob



Property graph data model

Query

- Find all users who are friends of Bob that shares similar friend as Bob



Property graph data model

Neo4j cypher query language

- Declarative language based on property graph model
- Example:
  - Find all users who are friends of Bob that shares similar friend as Bob
- Query:

```
MATCH  (e)<-[:FRIEND_OF]-(bob)-[:FRIEND_OF]->(f)-[:FRIEND_OF]->(e)
WHERE  bob.name = “Bob”
RETURN f.name AS name,
       count(e) AS score,
       collect(e.name) AS friends
ORDER BY score DESC
```

name	score	friends
“Zach”	2	[“Alice”, “Jim”]
“Jim”	1	[“Zach”]

Property graph data model

Neo4j cypher query language

- Declarative language based on property graph model
- Example:
  - For each user, find the number of his/her direct/indirect friends
- Query:

```
MATCH  (u:User)
OPTIONAL MATCH (u)-[:FRIEND_OF*]->(u2:User)
RETURN u.name AS name,
       count(DISTINCT u2) AS num
ORDER BY name
```

name	num
“Alice”	5
“Bob”	5
“Chad”	5
“Dave”	2
“Ed”	2
“Fred”	2
“Grace”	0
“Helen”	0
“Ingrid”	0
“Jim”	5
“Zach”	5

- Distributed DBMS
- OLAP
  - Multidimensional aggregation
- NoSQL
  - Graph database
- Future
  - NewSQL
  - Database-as-a-Service

Future

NewSQL

NewSQL database systems

- Targeted at OLTP workloads
- Features
  - Relational data model
  - SQL query language
  - ACID transactions
  - Runs on distributed cluster of shared-nothing nodes
- Some examples:
  - Clustrix
  - CockroachDB
  - Google Spanner
  - MemSQL
  - Microsoft Azure Cosmos DB
  - VoltDB

Database-as-a-Service (DBaaS)

RDBMS

- Amazon RDS (Amazon, Aurora, MySQL, MariaDB, SQL Server, Oracle, PostgreSQL)
  - https://aws.amazon.com/rds/
- Google Cloud SQL (MySQL, PostgreSQL)
  - https://cloud.google.com/sql/

NoSQL

- Amazon DynamoDB
  - https://aws.amazon.com/dynamodb/
- Microsoft Azure Cosmos DB
  - https://azure.microsoft.com/en-us/services/cosmos-db/

NewSQL

- Google Cloud Spanner
  - https://cloud.google.com/spanner/