Cloud Computing: Database as a Service Part I

Vijay Dialani, PhD
Boise State University
vijaydialani@boisestate.edu

What is a Database?

A database is an organized collection of data. The data are typically organized to model aspects of reality in a way that supports processes requiring information.

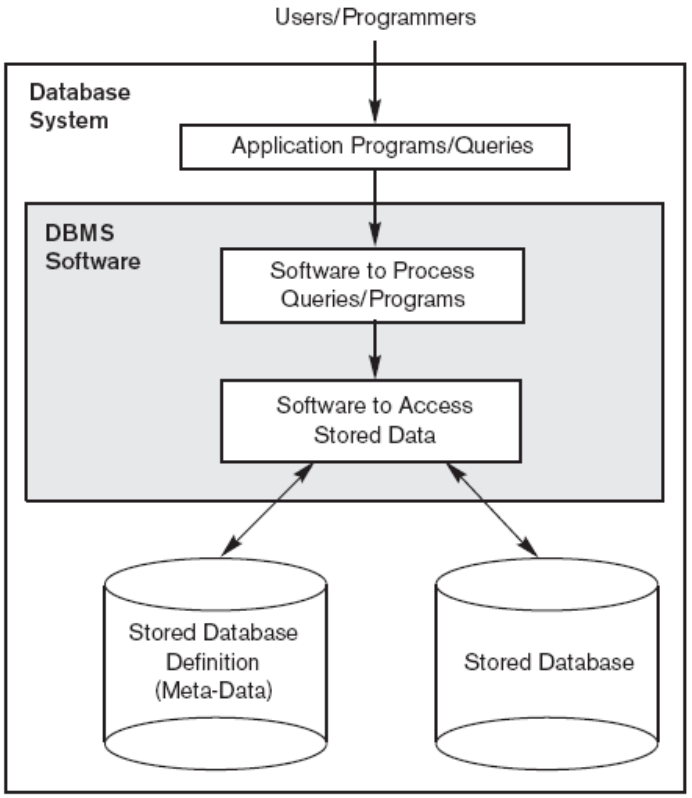


Figure 1.1
A simplified database system environment.

Types of Databases

A

- Active database
- Animation database

B

- Back-end database

C

- Centralized database
- Cloud database
- *Collection database*
- Collective Optimization Database
- Column-oriented DBMS
- Cooperative database
- Correlation database
- Current database

D

- Distributed database

D cont.

- Document-oriented database

E

- EDA database
- Endgame tablebase

G

- Gellish database
- Government database

M

- Mobile database

N

- Navigational database
- Non-native speech database

O

- Object database
- Online database
- Operational database
- Oxford English Corpus

P

- Parallel database
- Probabilistic database
- Project-Level Aid Database

R

- RecordSetter
- Relational database

S

- Simple Sloppy Semantic Database
- Suppliers and Parts database

T

- Triplestore

V

- Very large database
- Virtual private database
- Vulnerability database

X

- XLDB

SQL and NOSQL Databases

Data Model	Performance	Scalability	Flexibility	Complexity	Functionality
Key–Value Store	high	high	high	none	variable (none)
Column-Oriented Store	high	high	moderate	low	minimal
Document-Oriented Store	high	variable (high)	high	low	variable (low)
Graph Database	variable	variable	high	high	graph theory
Relational Database	variable	variable	low	moderate	relational algebra

NOTE

Rest of the presentation is about creating Database as a service using
Relational Data Model

Part II will discuss NOSQL implementations

Data Model

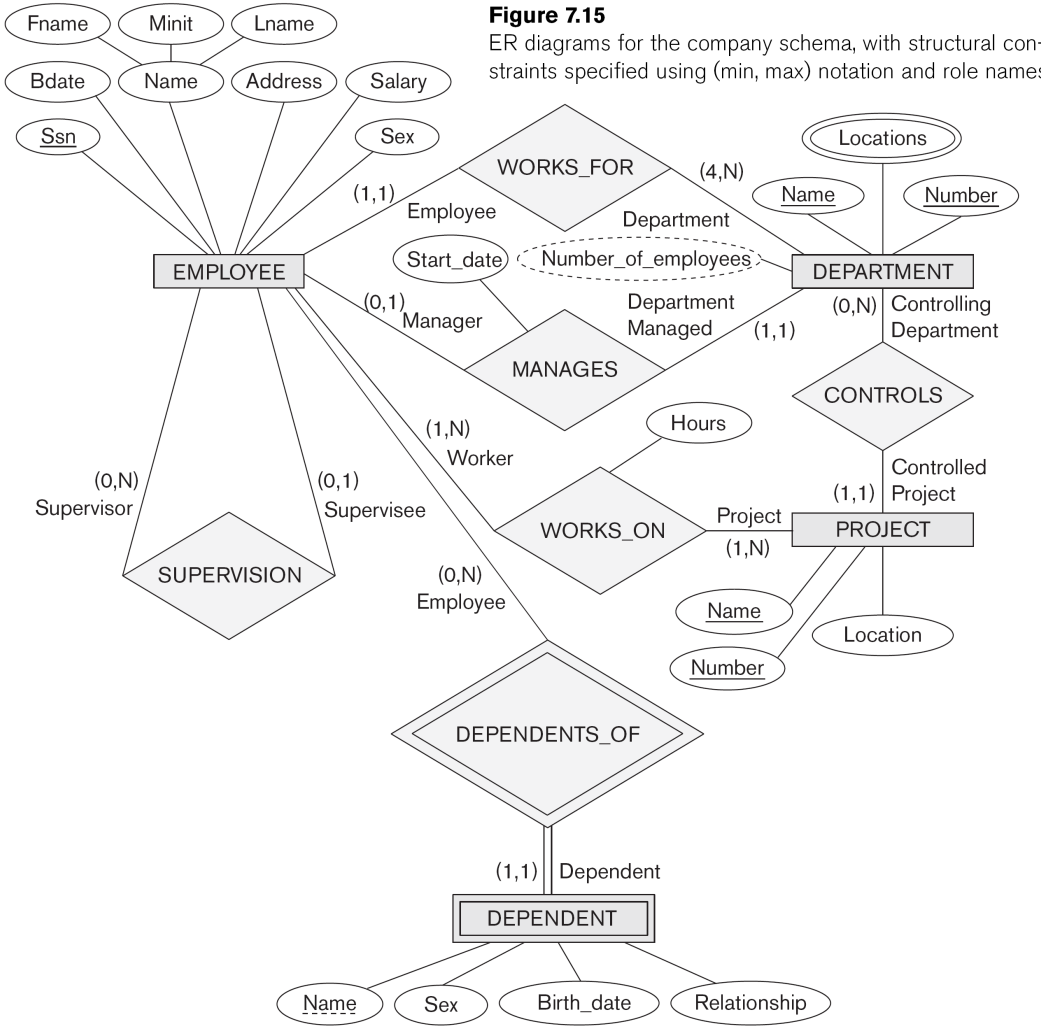
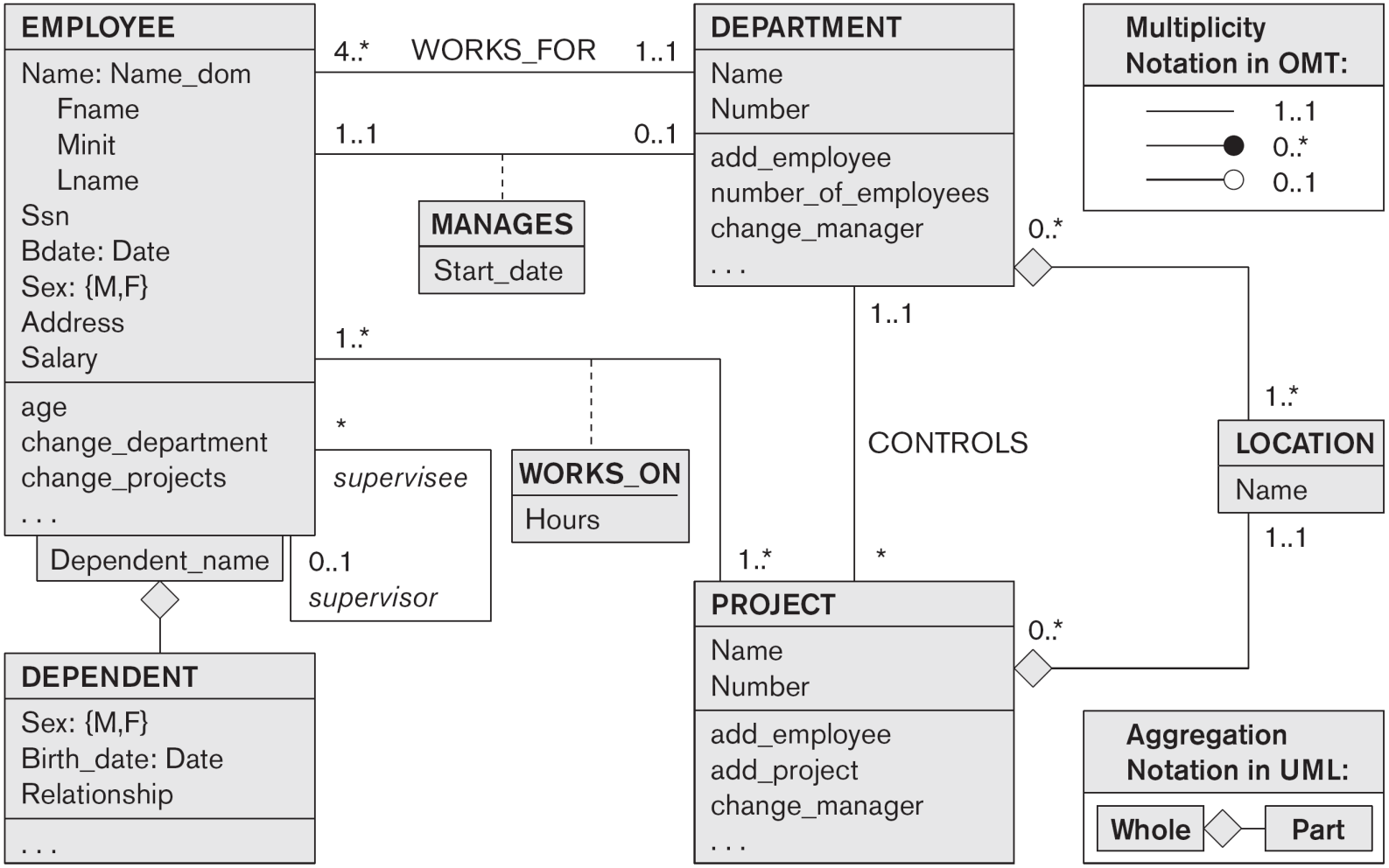


Figure 7.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.

[14]In some notations, particularly those used in object modeling methodologies such as UML, the (min, max) is placed on the *opposite sides* to the ones we have shown. For example, for the WORKS_FOR relationship in Figure 7.15, the (1,1) would be on the DEPARTMENT side, and the (4,N) would be on the EMPLOYEE side. Here we used the original notation from Abrial (1974).

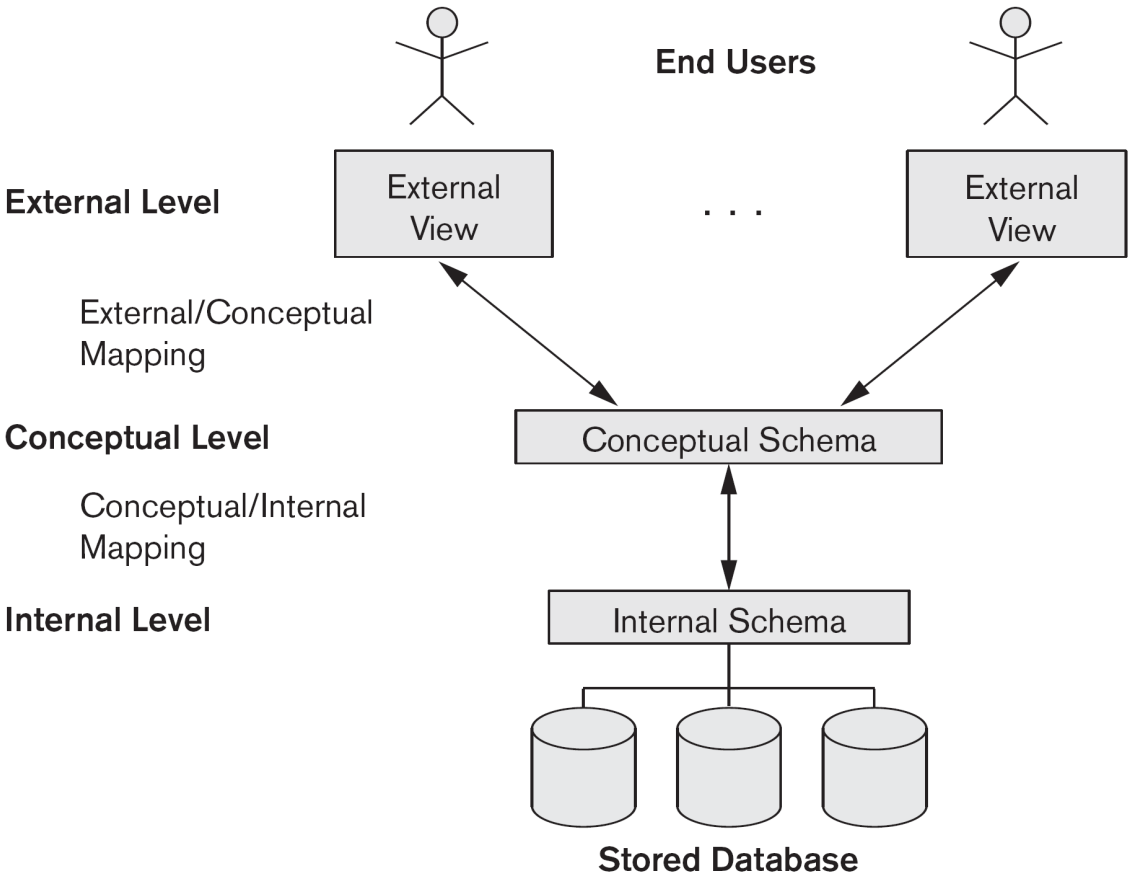
Data Model -> To Physical Database

Figure 7.16
The COMPANY conceptual schema in UML class diagram notation.



SQL databases conceptual view

Figure 2.2
The three-schema architecture.



Relational Cloud

Relational Cloud: A Database-as-a-Service for the Cloud
Carlo Curino et.al.
http://people.csail.mit.edu/nickolai/papers/curino-relcloud-cidr)

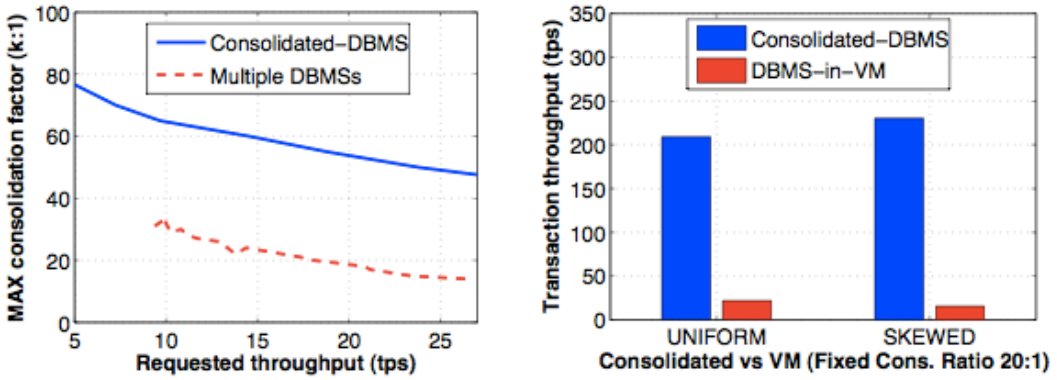


Figure 3: Multiplexing efficiency for TPC-C workloads

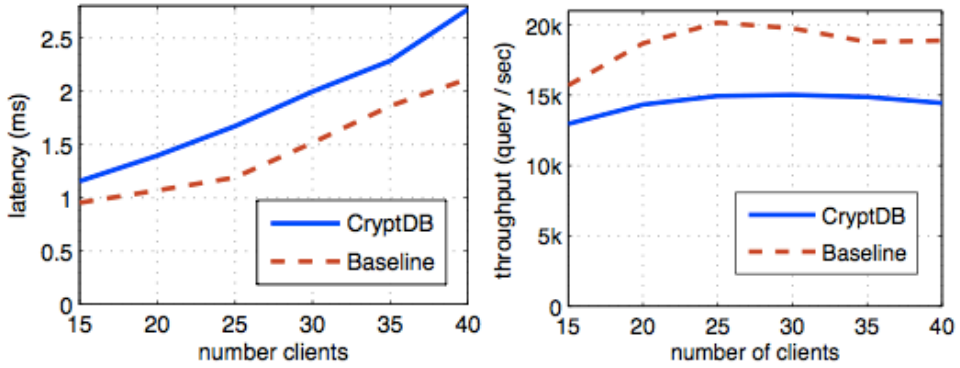


Figure 5: Impact of privacy on latency and throughput

SQL databases Partitioning Schemes

Problem I: Given a data model, we have too many number of rows, but each query operates on a range of rows.

Solution:
For a **share nothing architecture**, we can partition the data

Range partitioning: Selects a partition by determining if the partitioning key is inside a certain range. An example could be a partition for all rows where the column zipcode has a value between 70000 and 79999.

List partitioning: A partition is assigned a list of values. If the partitioning key has one of these values, the partition is chosen. For example all rows where the column Country is either Iceland, Norway, Sweden, Finland or Denmark could build a partition for the Nordic countries.

Hash partitioning The value of a hash function determines membership in a partition. Assuming there are four partitions, the hash function could return a value from 0 to 3. Composite partitioning allows for certain combinations of the above partitioning schemes, by for example first applying a range partitioning and then a hash partitioning.

Column Stores

Problem II: Given a data model and instance data, the query time varies because of the query optimizer introduced variance

Solution:
Use column stores – C-Store: A Column-oriented DBMS, Michael Stonebraker et.al.

Lets Review the Paper.

Column Stores

Problem III: Given a database, how do I ensure that the database is secured.

Solution:
Operate on Encrypted data – CryptDB: Protecting Confidentiality with Encrypted Query Processing, Popa et.al

Column Stores

Problem IV: Not all the items in the database are queried equally, how do I create the dynamic partitioning and load balancing

Solution:
Schism: a Workload-Driven Approach to Database Replication and Partitioning by Carlo Curino et.al.

Column Stores

Problem V: A lot of my data is shared between applications

Solution:
Use Multi-tenant databases:
Ruminations on Multi-Tenant Databases
http://www.db.in.tum.de/research/publications/conferences/BTW2007-mtd.pdf

Questions and Answers