KNU CSE

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

WHY USE A DATABASE?

Data & Knowledge Engineering (**DKE**) Lab.

Prof. Suh, Young-Kyoon

Email: yksuh@knu.ac.kr

Acknowledgements

- Some slides were written by Richard T. Snodgrass (University of Arizona) and Christian S. Jensen (Aalborg University).
- Michael Soo (amazon.com) provided some of the query processing slides.
- Kristian Torp (Aalborg University) converted the slides from Island Presents to Powerpoint.
- Curtis Dyreson (Utah State University) merged the slides with those from Peter Stewart (Bond University).
- The motivational example was given by Gary Lindstrom and Wei Tao (University of Utah).
- Young Suh (Kyungpook National University) fixed some typos and merged his slides with those from Rick Snodgrass (University of Arizona).

Outline

- Why Use a Database?
- The Field
- Overview of the Course

Prevalence of Databases (DBs)

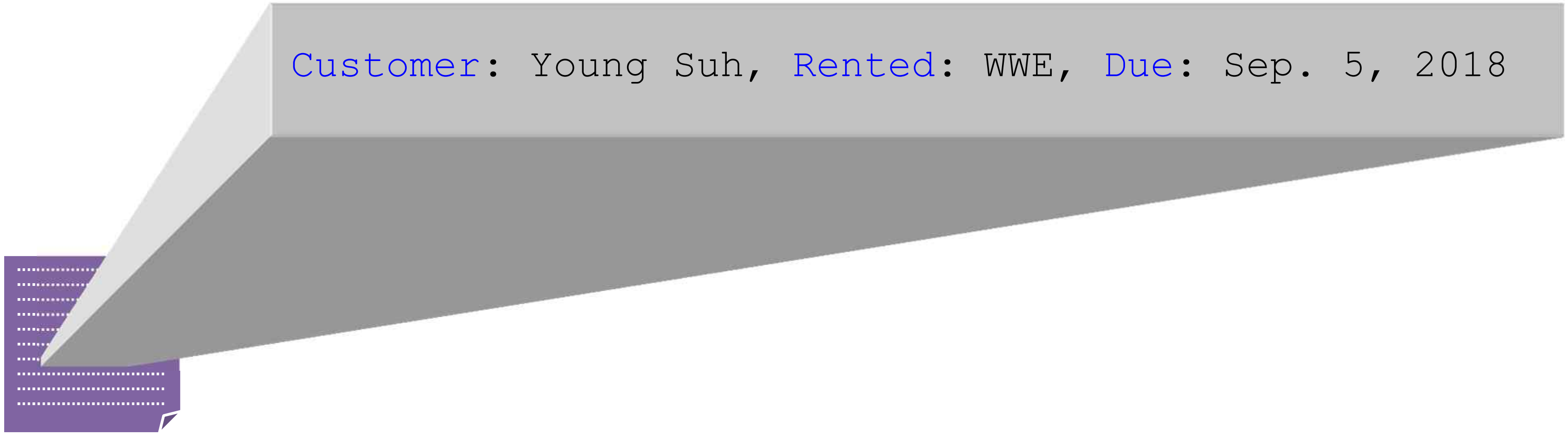
- Behind every successful website, there is a powerful database.
- Examples:
  - UPS / FedEx tracking
  - Amazon's/eBay's websites
  - Wal-Mart's inventory system
  - Dell's ordering system
  - Google's search engine
  - …

Data Management Example

- Scenario
  - You are a movie rental startup.
  - Your customers rent DVD copies of movies.
  - Several copies of each movie.
- Needs:
  - Which blu-ray disks have a customer rented?
  - Are any disks overdue?
  - When will a disk become available?
  - …

Solution: A “File-based” System

- (Create an) Edit `rented.txt` file



```
Customer: Young Suh, Rented: WWE, Due: Sep. 5, 2018
```

- Advantages?
  - Text editors are easy to use
  - Simple to insert a record
  - Simple to delete a record

Panacea???

Complication: Queries (질의)?

- Does not address needs

  Query 1: What movies has Gildong Hong rented?

  Execute (not quite right): Search for ‘`Gildong Hong`’.

  Execute: Search for

  ```
  ^\s+Customer:\s*Gildong\s+Hong\s*,\s+Rented:
  ```

  Query 2: Are any disks overdue?

  Execute: How to make this query???

- Requirements
  - Robust, sophisticated query language
  - Clear separation between data organization (schema) and data

<u>DataBase Management System (DBMS) Concepts</u>

Schema

DML

SQL

Complication: Integrity (무결성)

- Lacks data *integrity*, *consistency* (일관성)
  - Clerk misspells value/field

    ```
    Customer: Young Suk, Rented: Eraserhead, Deu: Sep. 5, 2018
    ```

  - Inputs improper value, same value differently

    ```
    Customer: Young Suh, Rented: The Eraserhead, Due: Oct. 5, 2018
    ```

  - Forgets/adds/reorders field

    ```
    Terms: weekly special Due: Sep. 5, 2018, Rented: Eraserhead
    ```

- Requirements
  - Enforce *constraints* (제약조건) to permit only *valid* information to be input.

DBMS Concepts

Integrity constraints

Types

Complication: Update (갱신)

- Add/delete/update fields in every record
  - Record store location.

    ```
    Customer: Young Suh, Rented: WWE, Due: Sep. 5, 2018, Store: Bukgu
    ```

  - Modify customer to first and lastname.

    ```
    First: Young, Last: Suh, Rented: WWE, Due: Sep. 5, 2018, Store: Bukgu
    ```

- Add/delete/update new information collections
  - `customer.txt` file to record information

    ```
    Customer: Young Suh, Phone: 557-3344
    ```

- Requirements
  - Ability to manipulate the way data is organized.

<u>DBMS Concepts</u>
DDL

Complication: Multiple Users (다중 사용자)

- Two clerks edit `rented.txt` file at the same time.

  1) 철수 starts to edit `rented.txt`, reads it into memory.

  2) 영희 starts to edit `rented.txt`.

  3) 철수 adds a record.

  4) 철수 saves `rented.txt` to disk.

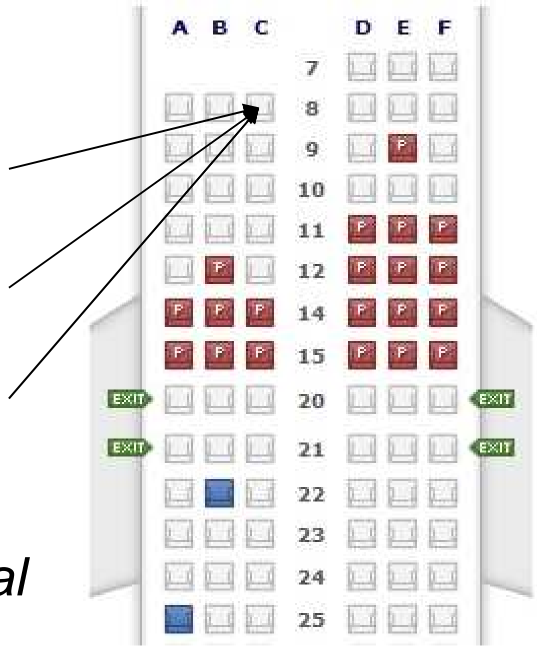
  5) 영희 saves `rented.txt` to disk.

  What's wrong with this scenario?

User 1

User 2

User 3



- Requirements
  - Must support multiple readers and writers.
  - Updates to data must (appear to) occur in *serial* order.

DBMS Concepts
Serializability
Concurrency control

Complication: Crashes (크래시)

- Crash during update may lead to *inconsistent* (불일치) state.
  - 철수 makes 250 of 500 edits to change ‘`Jane Doe`’ to her preferred name ‘`Jan Doe`’.
  - Before 철수 saves it, Windows crashes!
- Requirements
  - Must update on all-or-none basis (a.k.a, *atomicity*).
  - Implemented by *commit* (커밋) or *rollback* (롤백) if necessary.

DBMS Concepts

Transactions

Commit

Rollback

Recovery

Complication: Data Physically Separate (물리적으로 분리된 파일로 인한 난제)

- Need: want to inform Austin Power's (오스틴 파워) fans about new Austin Power's movie.
- Method
  - `customer.txt` contains addresses of customers.
  - Must merge with `rented.txt` to create mailing list.
- Problems ?!
  - Text editors incapable of such a merge (have to write a program)
  - What if several 홍길동s?
  - No information on some customers!?
- Requirements
  - *Uniquely* identify each customer.
  - Make sure we have information on customers that rent the movie.

DBMS Concepts

Joins

Keys

Foreign keys

Referential integrity

Complication: Security (보안)

- Customers want to know how many times a movie has been rented.
  - Provide access to `rented.txt`, but not to customer field, how I do that in an editor?
- Clerks under 19 should not see history of R-rated rentals (미성년자 관람불가 영화 렌탈).
  - Keep two lists of rentals?
- Requirements
  - Ability to control who has access to what information.

<u>DBMS Concepts</u>

Security

Views

Complication: Efficiency (효율성)

- Your customer list grows enormously.
  - `rented.txt` file gets huge (gigabytes, terabytes, or more of data).
  - *Slow* to edit.
  - *Slow* to query for customer information.
- Requirements
  - New data structures to improve query performance
  - System automatically modifies queries to improve speed.
  - Ability of system to scale (확장성) to handle huge datasets

DBMS Concepts

Indexes

Query optimization

Database tuning

Complication: New Needs (새로운 필요들)

- What pairs of movies are often rented together?
  - Calculate probability of movie combinations.
- Do we need more copies of the Austin Powers movie anywhere?
  - Visualize rental history of Austin Powers by store area.
- Requirements
  - Collect and analyze summary data (요약 데이터).
  - Use computer to *mine* for interesting trends and *predict* future trends (흥미로운 트렌드를 캐거나 예측)
  - Support access to data by sophisticated programs.

DBMS Concepts

Data mining

Big Data Analytics

Data warehouses

Database API

Limitations of File-based Systems

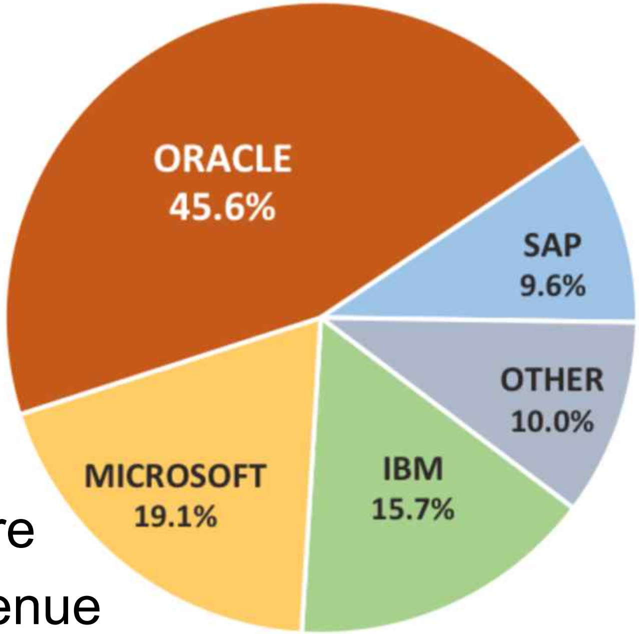
- Program must implement
  - Security (보안)
  - Concurrency control (동시성 제어)
  - Support for schema reorganization (스키마 재구성)
  - Data structures for performance improvement (성능 향상을 위한 데이터 구조): e.g. indexes (색인)
- Observation
  - Many applications need these services with high performance.
- Solution
  - Build and sell a software system to provide services, which is what?!

Outline

- Why Use a Database?
- The Field
- Overview of the Course

Major Players in the Field (industry)

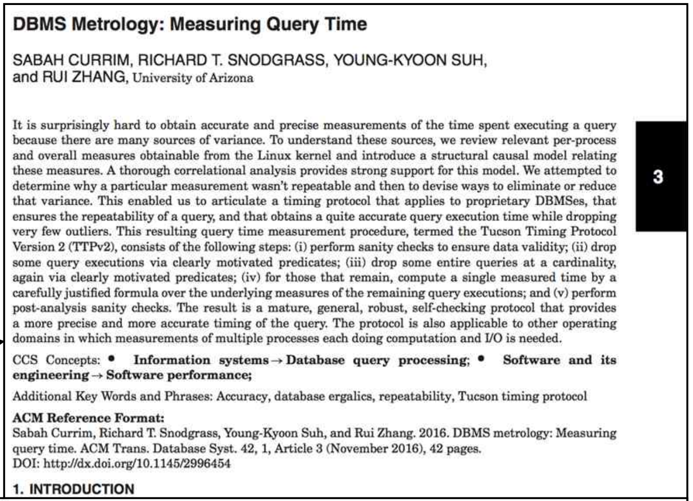
- Oracle
  - About 46% market share
  - $40 B (USD) (40조원) annual revenue
  - 138K employees (2018)
- Microsoft
  - Produces **SQL Server**; 19% market share
  - $110.4 B (US dollars) (2018) annual revenue
  - 131K employees worldwide (2018)
- IBM
  - Produces **DB2**; 16% market share
  - $79.14 B (USD) (2018) annual revenue
  - 380K employees worldwide (2018)



[Commercial Database Market Share]
(by Garner, Inc. 2016)

The Field (academic)

- (Referred) Professional Publications
  - *ACM Transactions on Database Systems (TODS)*
    - Quarterly, 700 pages per year
  - *IEEE Transactions on Knowledge and Data Engineering (TKDE)*
    - Bimonthly, 1K pages a year
  - *The VLDB Journal (VLDBJ)*
    - Bimonthly, 450 pages a year
  - *Information Systems (Info Sys)*
    - Bimonthly, 600 pages a year
  - Numerous other journals
- Unrefereed Technical Pubs.
  - *ACM SIGMOD record, IEEE Data Engineering Bulletin*



DBMS Metrology: Measuring Query Time

SABAH CURRIM, RICHARD T. SNODGRASS, YOUNG-KYOON SUH,
and RUI ZHANG, University of Arizona

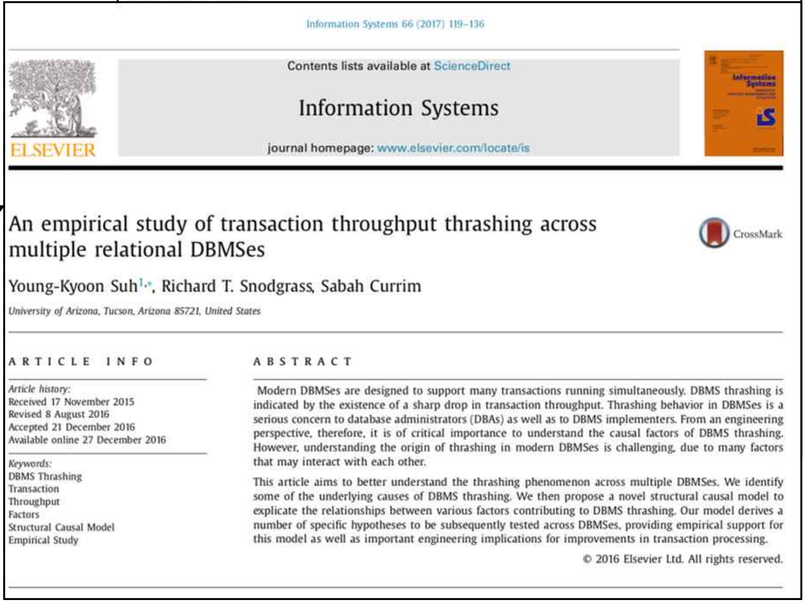
It is surprisingly hard to obtain accurate and precise measurements of the time spent executing a query because there are many sources of variance. To understand these sources, we review relevant per-process and overall measures obtainable from the Linux kernel and introduce a structural causal model relating these measures. A thorough correlational analysis provides strong support for this model. We attempted to determine why a particular measurement wasn't repeatable and then to devise ways to eliminate or reduce that variance. This enabled us to articulate a timing protocol that applies to proprietary DBMSes, that ensures the repeatability of a query, and that obtains a quite accurate query execution time while dropping very few outliers. This resulting query time measurement procedure, termed the Tucson Timing Protocol Version 2 (TTPv2), consists of the following steps: (i) perform sanity checks to ensure data validity; (ii) drop some query executions via clearly motivated predicates; (iii) drop some entire queries at a cardinality, again via clearly motivated predicates; (iv) for those that remain, compute a single measured time by a carefully justified formula over the underlying measures of the remaining query executions; and (v) perform post-analysis sanity checks. The result is a mature, general, robust, self-checking protocol that provides a more precise and more accurate timing of the query. The protocol is also applicable to other operating domains in which measurements of multiple processes each doing computation and I/O is needed.

CCS Concepts: • **Information systems → Database query processing;** • **Software and its engineering → Software performance;**

Additional Key Words and Phrases: Accuracy, database ergalics, repeatability, Tucson timing protocol

ACM Reference Format:
Sabah Currim, Richard T. Snodgrass, Young-Kyoon Suh, and Rui Zhang. 2016. DBMS metrology: Measuring query time. ACM Trans. Database Syst. 42, 1, Article 3 (November 2016), 42 pages.
DOI: http://dx.doi.org/10.1145/2996454

1. INTRODUCTION



Information Systems 66 (2017) 119–136

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

An empirical study of transaction throughput thrashing across multiple relational DBMSes

Young-Kyoon Suh[1,*], Richard T. Snodgrass, Sabah Currim

University of Arizona, Tucson, Arizona 85721, United States

ARTICLE INFO

Article history:
Received 17 November 2015
Revised 8 August 2016
Accepted 21 December 2016
Available online 27 December 2016

Keywords:
DBMS Thrashing
Transaction
Throughput
Factors
Structural Causal Model
Empirical Study

ABSTRACT

Modern DBMSes are designed to support many transactions running simultaneously. DBMS thrashing is indicated by the existence of a sharp drop in transaction throughput. Thrashing behavior in DBMSes is a serious concern to database administrators (DBAs) as well as to DBMS implementers. From an engineering perspective, therefore, it is of critical importance to understand the causal factors of DBMS thrashing. However, understanding the origin of thrashing in modern DBMSes is challenging, due to many factors that may interact with each other.

This article aims to better understand the thrashing phenomenon across multiple DBMSes. We identify some of the underlying causes of DBMS thrashing. We then propose a novel structural causal model to explicate the relationships between various factors contributing to DBMS thrashing. Our model derives a number of specific hypotheses to be subsequently tested across DBMSes, providing empirical support for this model as well as important engineering implications for improvements in transaction processing.

The Field (Cont’d)

- Conferences
  - ACM SIGMOD International Conference on Management of Data (*SIGMOD*)
    - Late May or early June, 500 pages a year (acceptance ratio: ~ 20%)
  - ACM Principles of Database Systems (*PODS*): theory-oriented
    - In conjunction with SIGMOD, 300 pages a year (a/r: ~ 20%)
  - Proceedings of the Very Large Data Bases Endowment (*PVLDB*)
    - Mid-September but monthly reviewed, 500 pages a year (a/r: ~ 20%)
  - IEEE International Conference on Data Engineering (*ICDE*)
    - April, 600 pages a year (a/r: ~ 25%)
  - EDBT/ICDT: alternate years (March/January), 400 pages a year
  - 8-10 specialized conferences a year: 300 x 8 = 2400 pages a year
- 8K pages per year of research papers. Amazing?!

Outline

- Why Use a Database?
- The Field
- Overview of the Course

The Course

- Motivation (**M**)

- Introduction
  - Core Concepts
  - Functions of a DBMS
  - When Not to Use a DBMS

- Conceptual Database Design
  - Basic Entity-Relationship Model
  - Constraints
  - Specialization and generalization
  - View integration

The Course (Cont'd)

- The Relational Model
  - Definition
  - Integrity constraints
  - Mapping an ER schema to tables (relations)
- Relational Query Languages
  - Relational algebra
  - Relational calculi: domain and tuple
- SQL DDL and DML: using Oracle with labs
  - Schema definition
  - Querying
  - Modifications
  - Views (read-only)
  - Embedded SQL / (Oracle) PLSQL
  - Transaction management

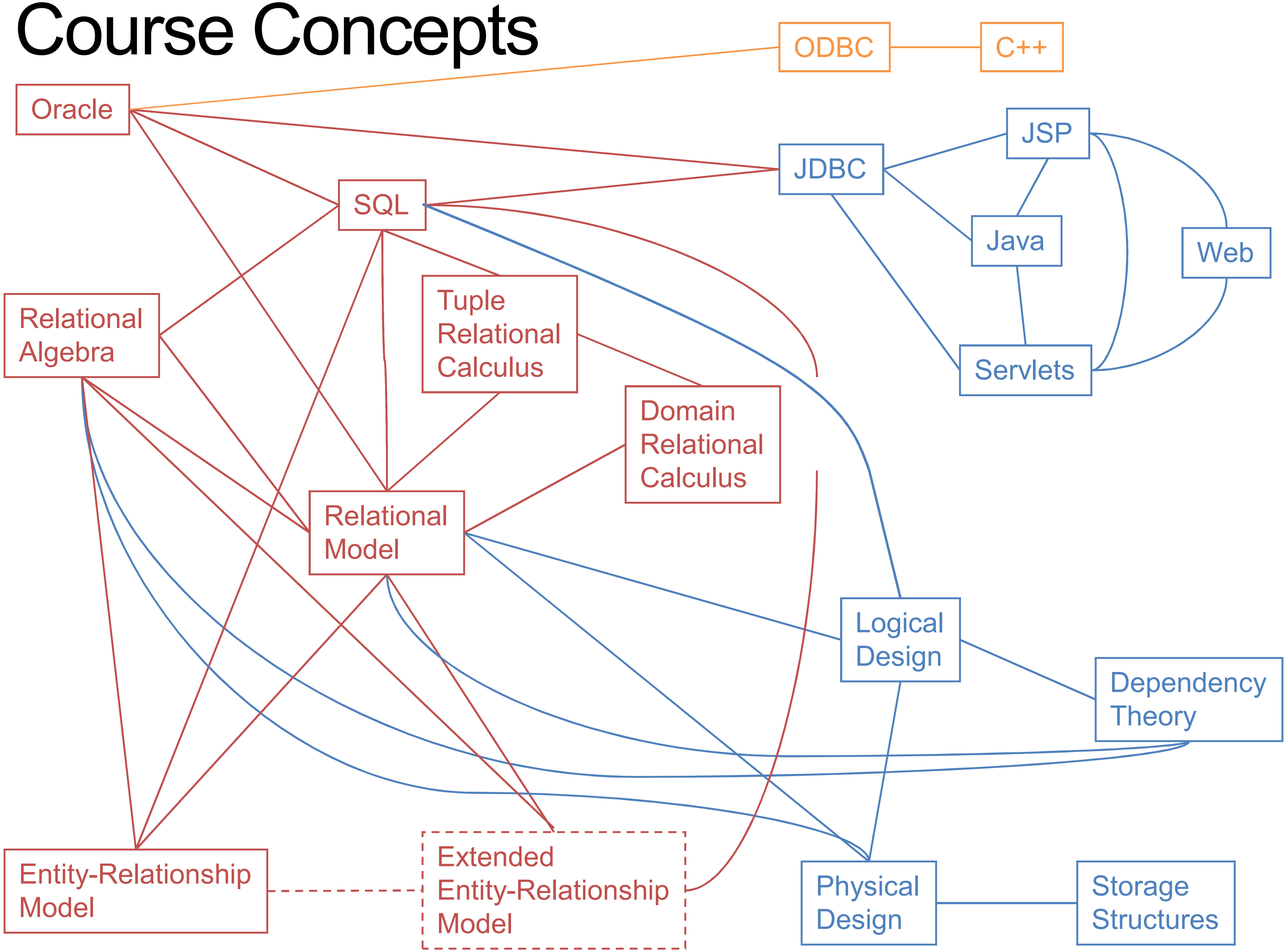
The Course (Cont’d)

- Web/Database Connections: perhaps through project
  - HTTP (HyperText Transfer Protocol)
  - Static Database Access
  - Dynamic Database Access
- Database Application Design and Implementation
  - Direct database interfaces (via SQL*PLUS or else)
  - Indirect interfaces: JDBC/ODBC
  - Web interfaces: JSP, Node.js, …

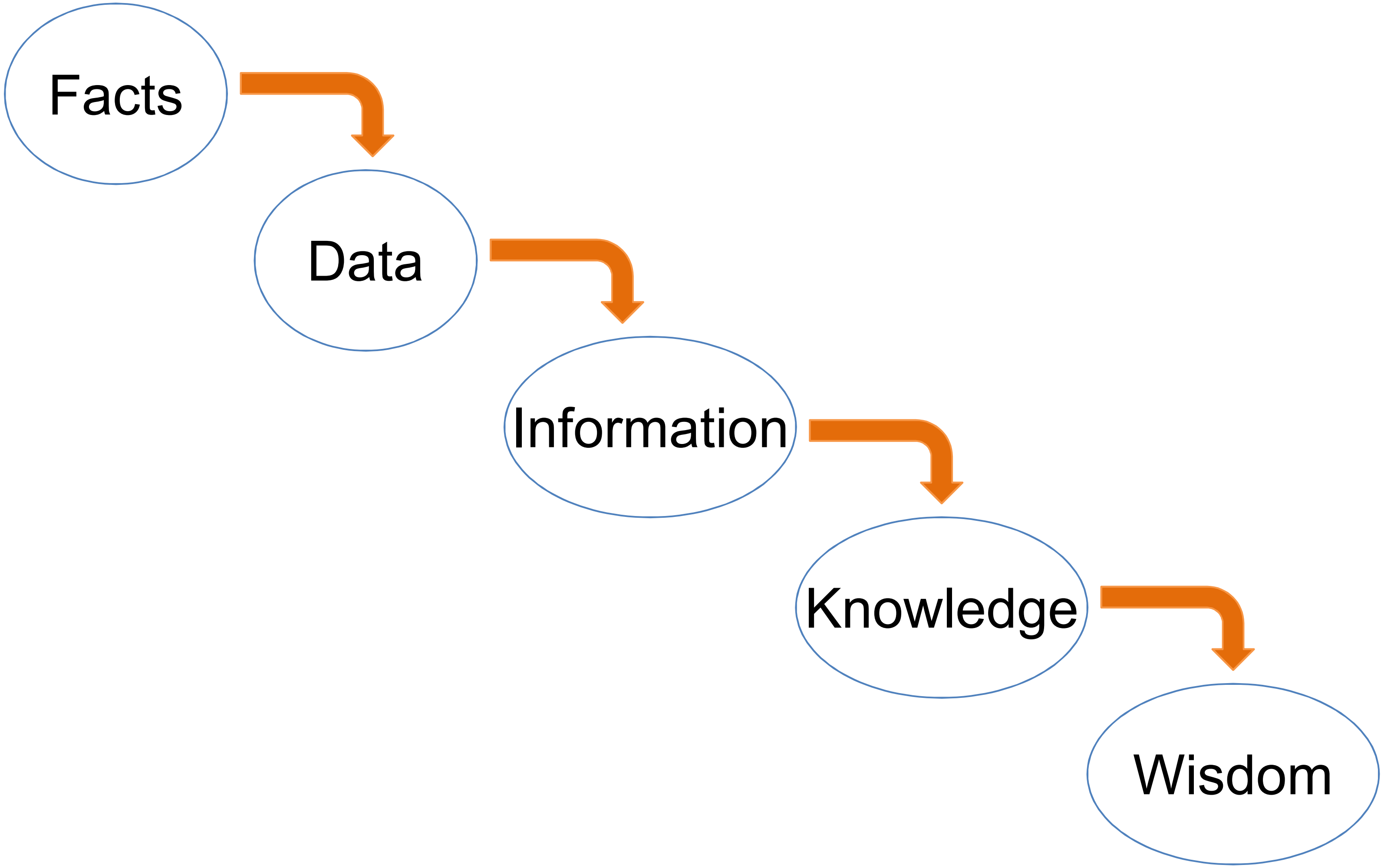
The Course (Cont'd)

- Logical Database Design
  - Properties of a good design
  - Functional dependencies and keys
  - Normal forms (정규화): 3NF, BCNF
  - Decomposition algorithms
- Physical Database Design
  - Relational structures: heap, sorted, compressed
  - Indexes: primary and secondary, B-trees
- Latest Topics (if time)
  - Data models: Unstructured, key-value paired, …
  - Query languages: SPARQL, …
  - Next-generation DB technologies: Hadoop, NoSQL, Column-oriented…

Course Concepts



Your Intellectual Journey



Facts

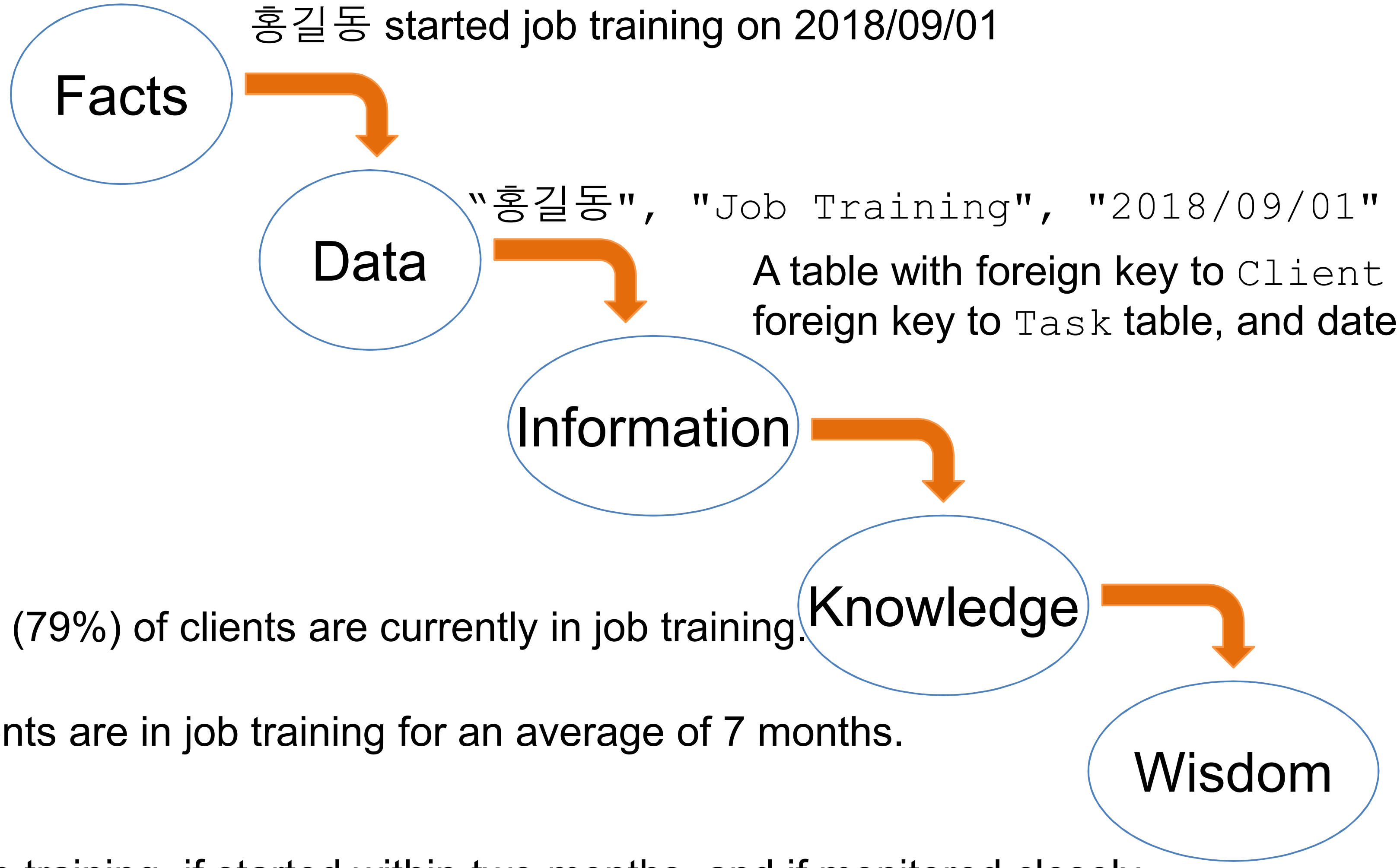
Data

Information

Knowledge

Wisdom

An Example



Facts

홍길동 started job training on 2018/09/01

Data

```
“홍길동", "Job Training", "2018/09/01"
```

Information

A table with foreign key to `Client` table, foreign key to `Task` table, and date.

Knowledge

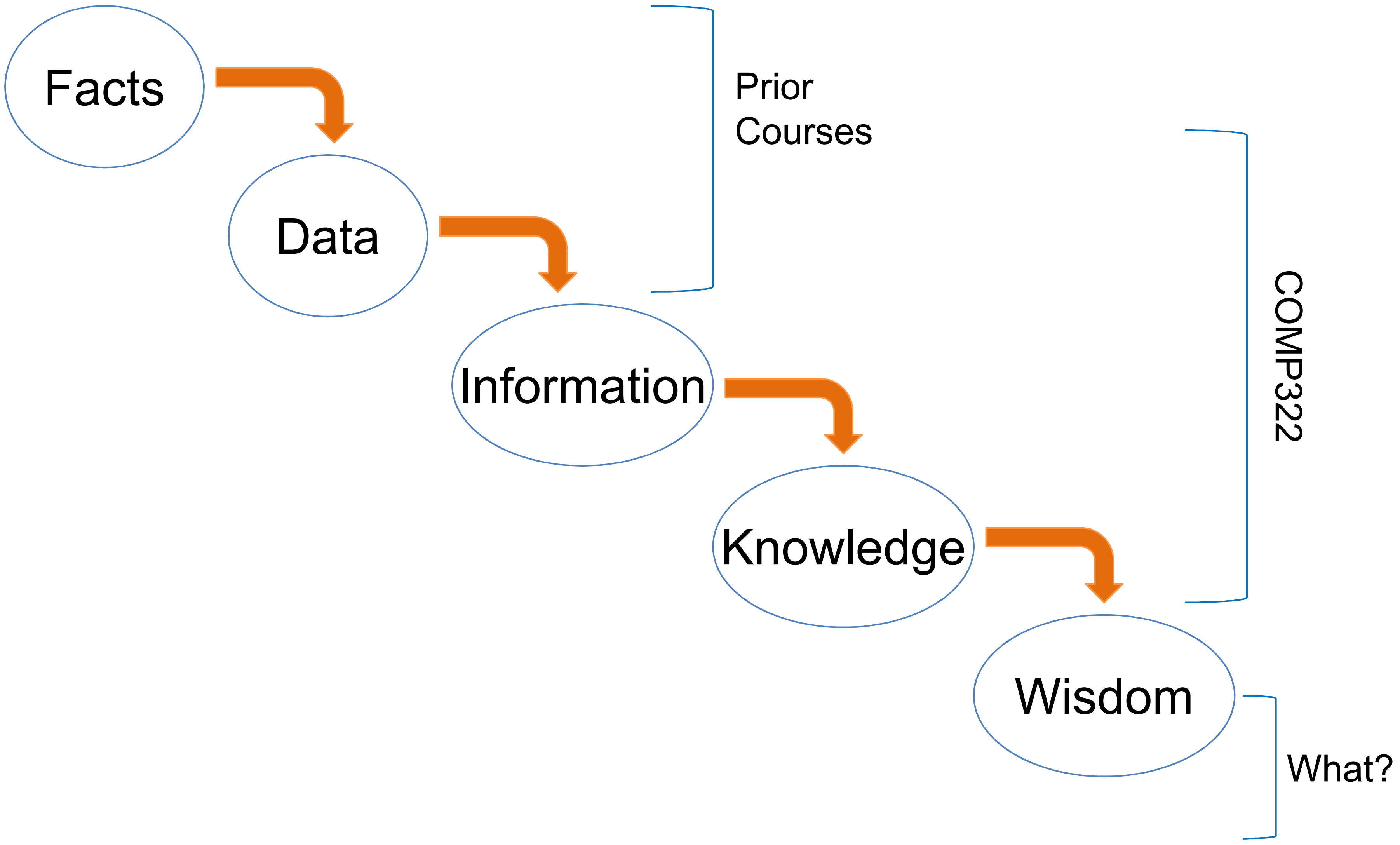
198 (79%) of clients are currently in job training.

Clients are in job training for an average of 7 months.

Wisdom

Job training, if started within two months, and if monitored closely, can be effective for a majority of people on welfare.

Relationship to Other Courses



QUESTIONS?