CSCI1270
Introduction to Database Systems

with thanks to Prof. George Kollios, Boston University
Prof. Mitch Cherniack, Brandeis University
Prof. Avi Silberschatz, Yale University

What is a Database System?

- Database:

  A very large collection of *related* data

- Models a real world enterprise:
  - Entities
    - Sports: Teams, players, games
    - University: Students, professors, books, courses
  - Relationships
    - The Patriots are *playing* in the Superbowl

- DBMS (**D**ata**b**ase **M**anagement **S**ystem): A software system that can be used to store, manage, retrieve and transform data from a database.
- Database System: DBMS+data (+ applications)

Why Study Databases?

- Shift from computation to information
  - Always true for corporate computing
  - Scientific datasets:
    - Astronomy
    - Biology
    - Particle Physics
  - The Web
- DBMS encompasses much of CS in a practical discipline

Operating systems	Languages
Distributed systems	Performance
Theory	AI

- JOBS!

  High-paying jobs!!!

Why Databases?

- Why not store everything in flat files?

  i.e., use the file system of the OS: cheap, simple…

[***Name, Course, Grade***] ← File system does not know even this.

John Smith, CS22, B;

Mike Stonebraker, CS123, A;

Jim Gray, CS127, A;

John Smith, CS227, B+;

……

This is how things were in the "Bad Old Days"

- Simple? Yes, but not scalable…

  Have to scan and reparse the file on every access.

Scalability

- While you might build a database to store your favorite recipes, this is not the interesting case.

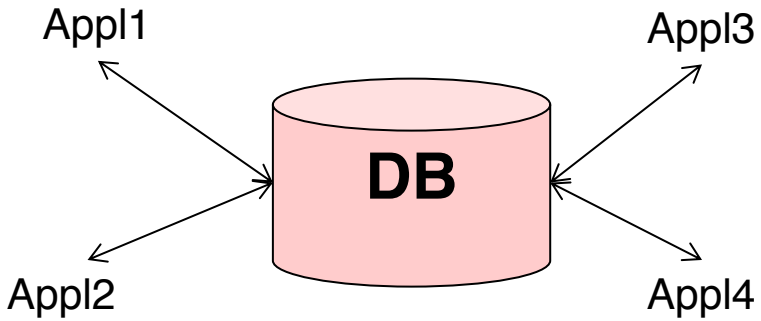
- Modern DBMS's are designed to scale

  (i.e., >> a terabyte)

  Must be well optimized on a single node

  Must be able to execute on 100's or 1000's of nodes.

DBMS Workloads



Data is:

mutable => changes often

(1) data and (2) structure

shared between different users and applications

persistent and long-lived => must withstand failures

evolvable

Storage Problems
Apps Own Files

- Data redundancy and inconsistency
  - Multiple file formats,
  - duplication of information in different files

*MyApp (**Name, Course, Email, Grade**)*

John Smith, CS112, js@cs.bu.edu, B
Jim Gray, CS560, jg@cs.bu.edu, A
John Smith, CS560, js@cs.bu.edu, B+

YourApp (***Name, Email, Course, Grade***)

Mike Stonebraker, ms@cs.bu.edu, CS234, A
J. Smith, js@cs.bu.edu, CS560, B+

Why is this a problem?

- Wasted space (?)
- Potential inconsistencies

  (e.g., multiple conventions, John Smith vs Smith J.)

Retrieval Problems

- Data retrieval (Query):
  - Find the students who took CS560
  - Find the students with GPA > 3.5

  For every query we would need to write a program!

  Each program would read the whole file.

- We need the retrieval to be:
  - Easy to write
  - Execute efficiently

Data Integrity

- Data Integrity in flat file model
  - Poor support for sharing:
    - Prevent simultaneous modifications
  - Poor coping mechanisms for system crashes
  - No means of Preventing Data Entry Errors (checks must be hard-coded in the programs)
  - Security problems
- Database systems offer solutions to all of the above problems

Evolution of data

- Long-lived data ➔ Evolution
- What happens if I need to change how the data is stored?
  - Access patterns change
  - Tuning
- Should not have to re-write all my applications.

- Solution: Data independence!

Data Organization

- **Data Models**: a framework for describing
  - data objects
  - data relationships
  - data semantics
  - data constraints
- Presents primitives for
  - Representing Data → Data Definition Language (DDL)
  - Manipulating Data → Data Manipulation Language (DML)
- We will concentrate on the **relational model**
- Other models:
  - ***Entity-Relationship model*** **(we will discuss)**
  - object-oriented model
  - semi-structured data models, XML
  - Array data model

Database Schema

- Similar to types and variables in programming languages
- **Schema** – the structure of the database
  - e.g., the database consists of information about a set of customers and accounts and the relationship between them
  - Expressed in some *data model* (DDL)
  - Occurs at multiple levels
    - **Logical schema**: database design at the logical level
    - **Physical schema**: database design at the physical level

Levels of Data Modeling

- **Logical level**: describes data stored in database, and the relationships among the data.

```
type customer = record
                    name : string;
                    street : string;
                    city : integer;
                end;
```

- **Physical level:** describes how a record is stored.
  - Sorting, page-alignment, indexes
- Also, **View level**: application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes.

Relational Model

- Example of tabular data in the relational model

Attributes

Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
019-28-3746	Smith	North	Rye	A-201

Key

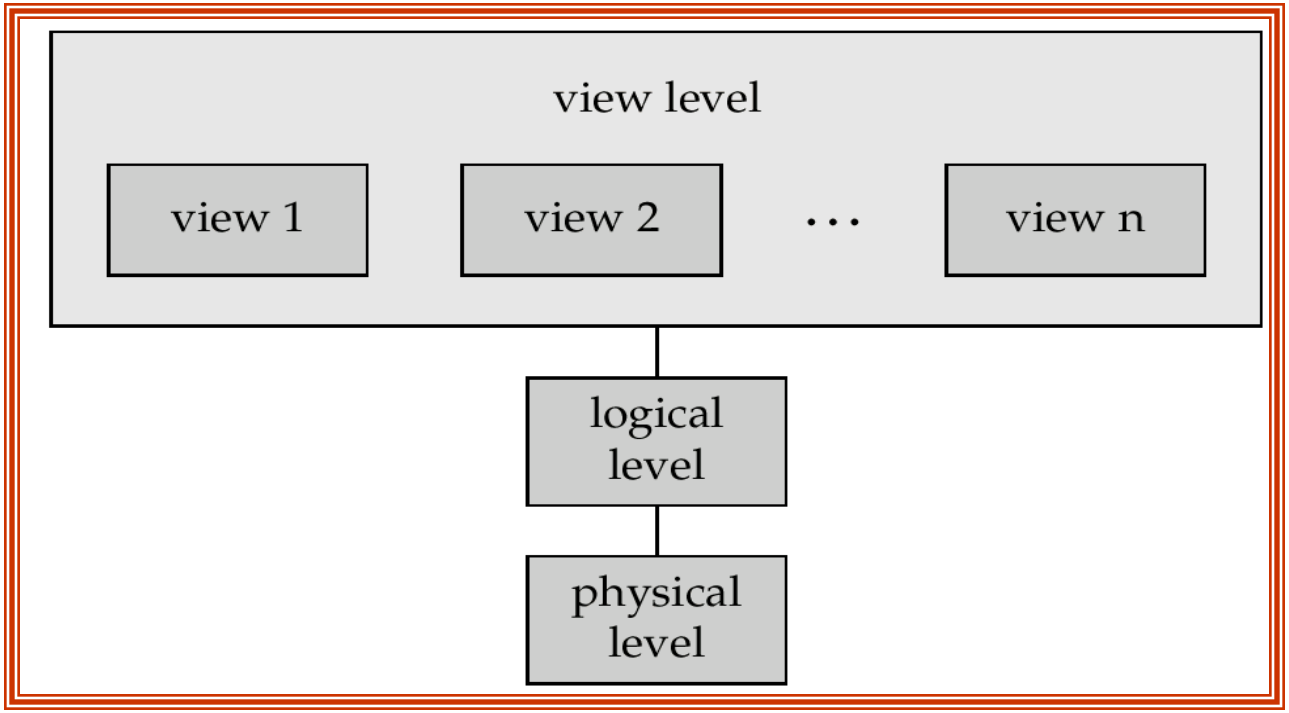
Key

Schema = **Customer** (Customer-id, customer-name, customer-street, customer-city, account-number)

This whole table is an instance of the Customer schema.

View of Data

A logical architecture for a database system



Data independence!

Data retrieval

- Queries

  Query = Declarative data retrieval

  *describes **what** data to retrieve, not **how** to retrieve it*

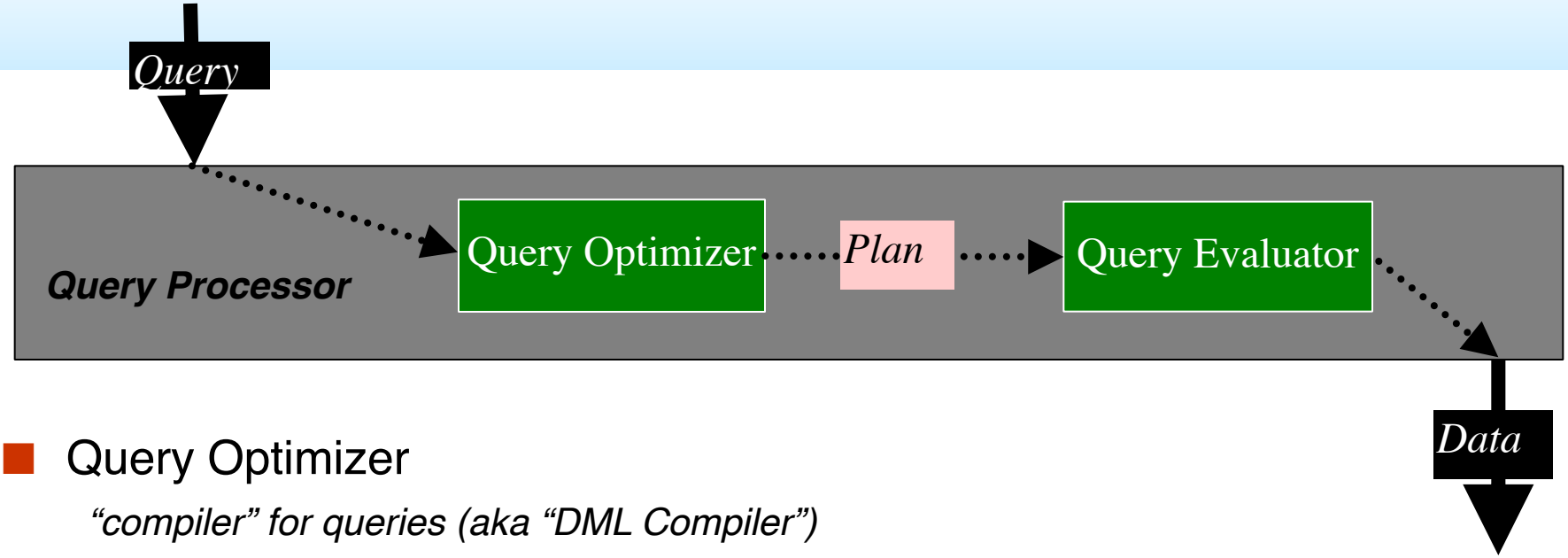
  Ex. Give me the students with GPA > 3.5 vs

  Scan the student file and retrieve the records with GPA > 3.5

- Why?

  1. Easier to write
  2. More efficient to execute (why?)
  3. More amenable to change (create/drop an index)

Data retrieval



- Query Optimizer

  "compiler" for queries (aka "DML Compiler")

  Query Plan similar to Assembly Language Program

Optimizer can do better With declarative queries:

1. *Algorithmic Query (e.g., in C) ⇒ 1 Plan to choose from*
2. *Declarative Query (e.g., in SQL) ⇒ n Plans to choose from*

SQL

- SQL: widely used (declarative) non-procedural language
  - E.g. find the name of the customer with customer-id 192-83-7465

    select *customer.customer-name*
    from *customer*
    where *customer.customer-id* = ‘192-83-7465’
  - E.g. find the balances of all accounts held by the customer with customer-id 192-83-7465

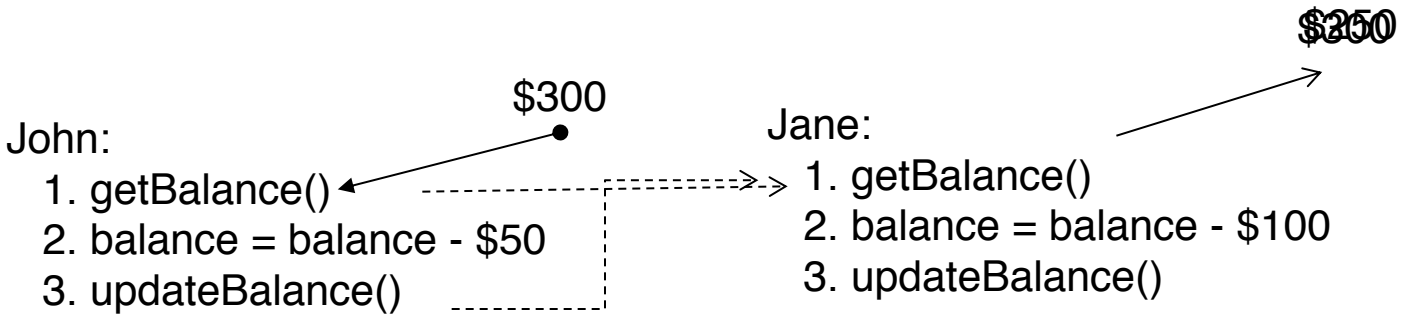
    select *account.balance*
    from *depositor*, *account*
    where *depositor.customer-id* = ‘192-83-7465’ **and**
    depositor.account-number = account.account-number

Data Integrity

Transaction processing

- Why must we manage concurrent access to data?

John and Jane withdraw $50 and $100 from a common account…



Initial balance is $300. Final balance…

It depends…

Data Integrity
Recovery

Transfer $50 from account A ($100) to account B ($200)

1. A.getBalance()
2. If($balance_A$ > $50)
3. $balance_A$ = $balance_A$ – 50
4. A.updateBalance($balance_A$)
5. B.getBalance()
6. $balance_B$ = $balance_B$ + 50
7. B.updateBalance($balance_B$)

Recovery management

Database Administrator (DBA)

- Database design
  - Balance needs of multiple applications
- Database tuning
  - Make applications run faster
  - Pick physical structures
- Manage reliability, correctness, security, …

Alternative Approaches

- NoSQL (e.g., Mongo DB, Cassandra, CouchBase)
- Hadoop
- Etc.

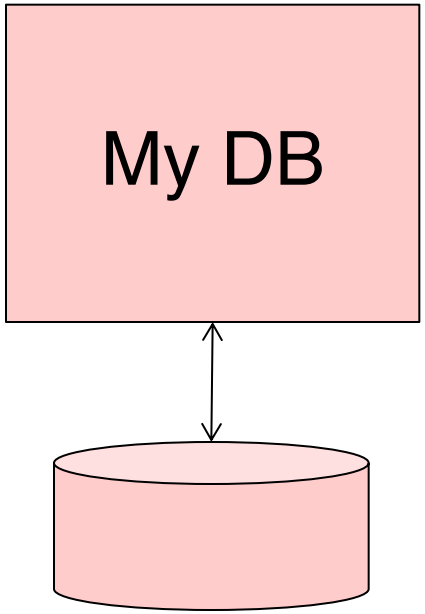
- They give up some of the things that we just discussed in order to
  - scale better
  - be easier to deploy

- OPINION: Misguided attitude that it's only code ->
  thus I'll do it myself.

The Hardware Matters

- Legacy of disk-based processing
  - The disk is always the bottleneck
  - Minimize disk accesses
- These days main memory is getting cheap and plentiful.
  - Store all data in main memory.
  - Still need some way to persist data (e.g., disk, NVM)
- For distributed DBMS's, the network is the bottleneck
  - Minimize message passing and data copying.
- Very fast networks are starting to be affordable.

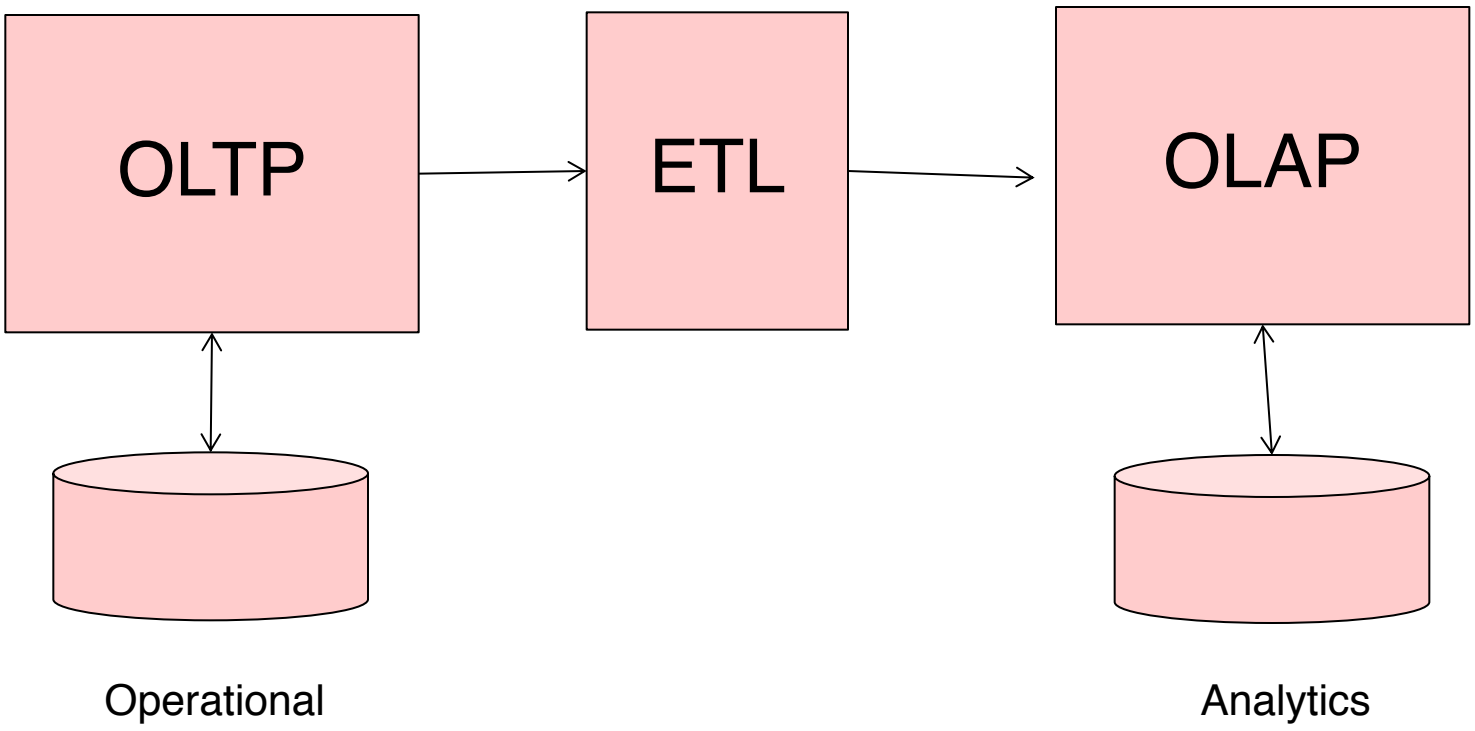
Assumption



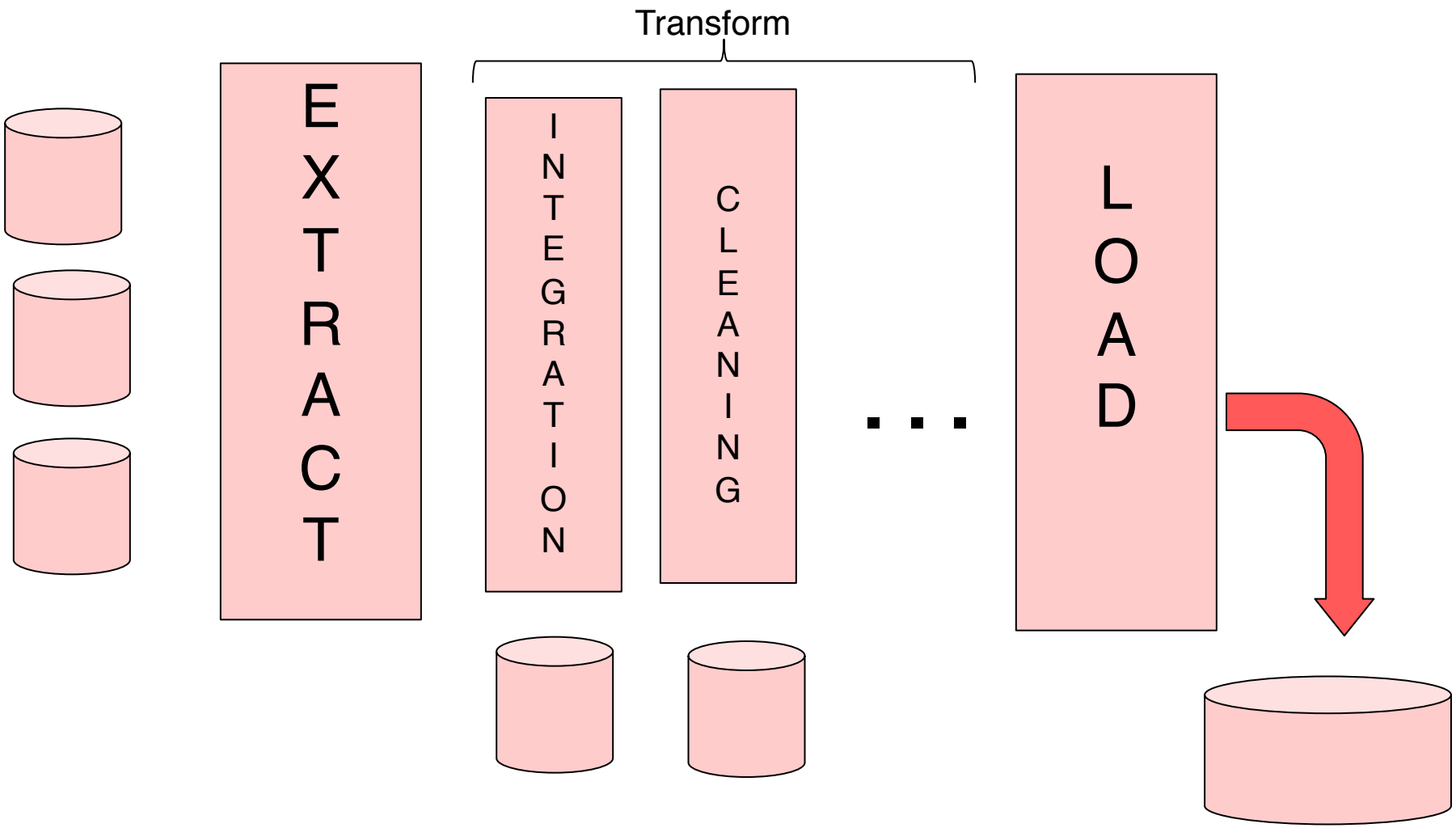
The Workload Matters

- Workload = the queries that are presented to the system.
- Applications have different characteristics that effect the workload.
- One-Size does not fit all.
  - OLTP
  - OLAP
  - Science

Reality



A Closer Look
ETL



Outline

- 1st half of the course: application-oriented
  - How to develop database applications: User + DBA
- 2nd part of the course: system-oriented
  - Learn the internals of a relational DBMS (developer for Oracle..)