CS3200: Database Design

Dr. John Rachlin
Email: j.rachlin@northeastern.edu

Introduction and Key Concepts

Why learn about databases?

Answer: Because databases are *ubiquitous*!

u·biq·ui·tous

/yo͞o'bikwədəs/

adjective

adjective: **ubiquitous**

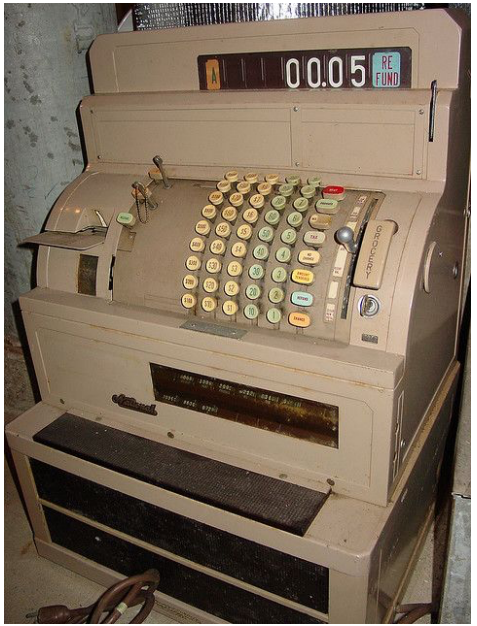
present, appearing, or found everywhere.

"his ubiquitous influence was felt by all the family"

synonyms: omnipresent, ever-present, everywhere, all over the place, pervasive, universal, worldwide, global; More

antonyms: rare

Grocery shopping before databases

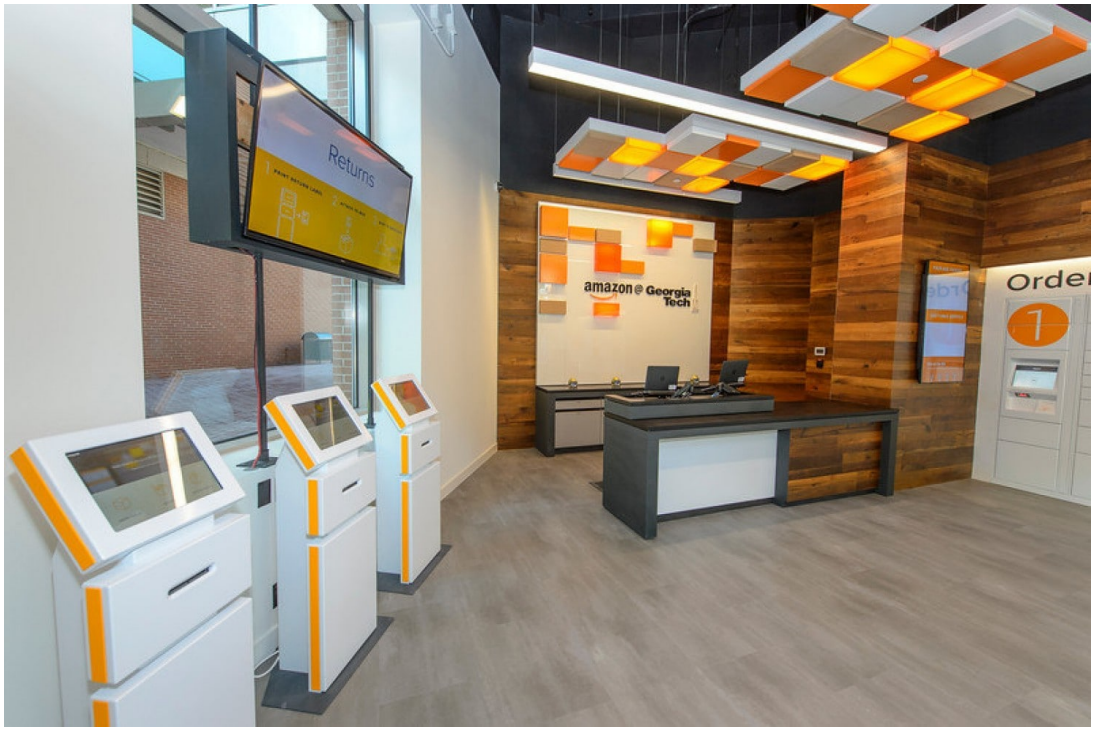


Amazon to Buy Whole Foods for $13.4 Billion

Amazon agreed to buy the upscale grocery chain Whole Foods for $13.4 billion, in a deal that will instantly transform the company that pioneered online shopping into a merchant with physical outposts in hundreds of neighborhoods across the country.

Source: The New York Times, June 16, 2017

Will Amazon transform the supermarket?



BERKELEY, Calif. — Amazon has been aggressively courting students as part of its experiment to bring its enormous online shopping operation into the brick-and-mortar world. Now, the company is launching Amazon Instant Pickup, a service that allows customers to order certain items from their smartphones for pickup within minutes of purchase.

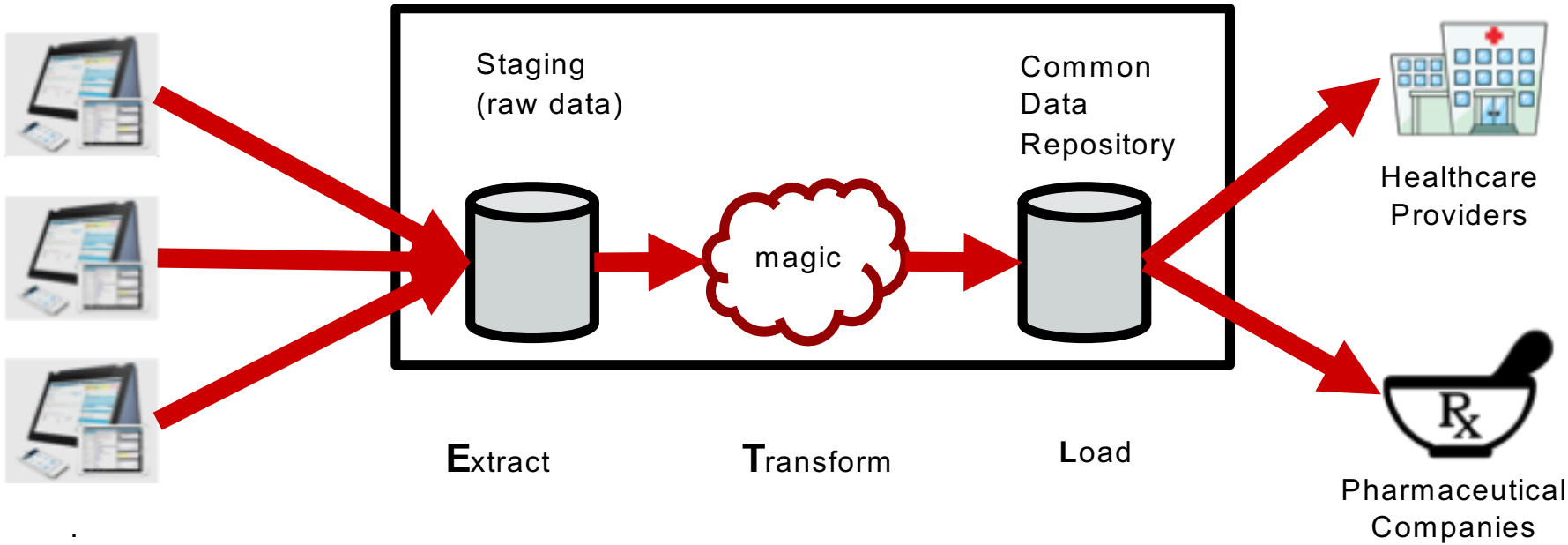
Source: The Washington Post (Aug 16, 2017)

EMR: Electronic Medical Records

- Patient Demographics
- Doctors
- Insurance
- Medical history
- Allergies
- Procedures / Measurements (BP, Temp, O_2)
- Order Lab Tests Lab Results
- Dr. Notes

Next time you get a physical, notice how much time your doctor spends at a computer terminal!

Optum Analytics: Aggregator of 90+ million patient records



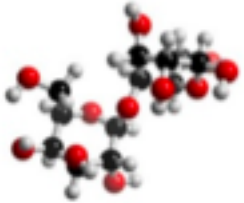
.
.
.

What questions would you ask if you had the medical records for 90+ million patients?

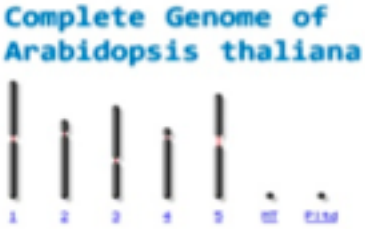
Bioinformatics Databases

Sequences

Structure

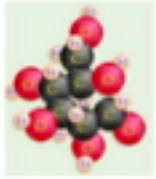


Genome



Pathways

Lipids, Carbohydrates

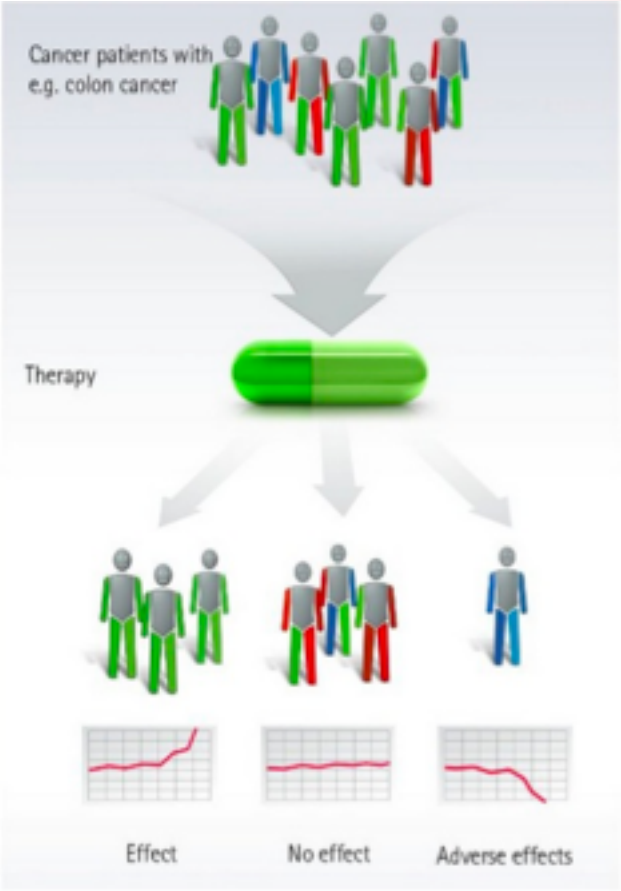


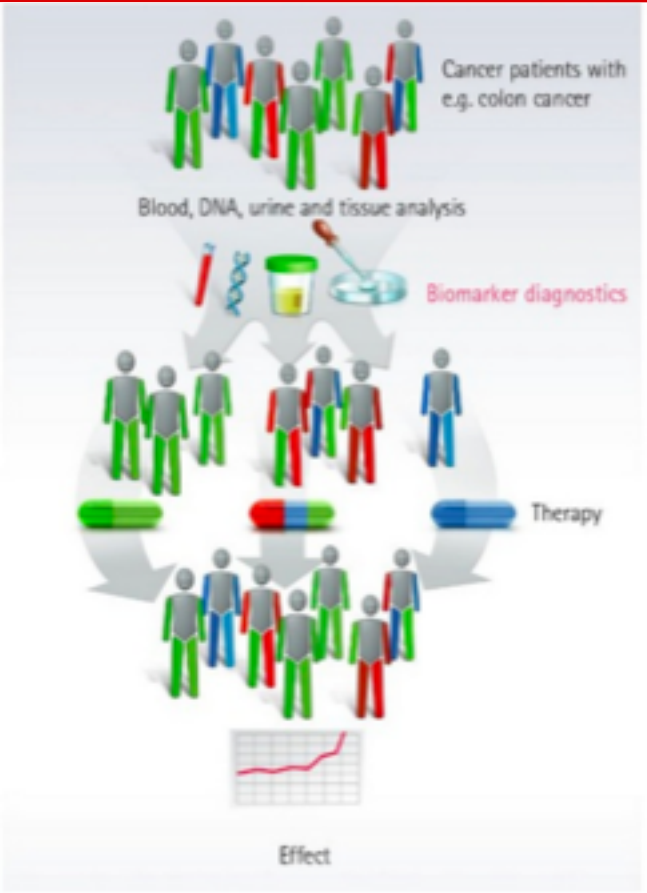
Literature

Primary Database: Direct experimental results
Secondary Database: Derived databases from transformation & analysis

Source: https://www.slideshare.net/shwetakagliwal/biological-databases-11267007

The coming age of personalized medicine





The promise of personalized medicine:

- Improved efficacy
- Reduced adverse side-effects
- Reduced “trial-and-error” delays in the treatment of time-critical diseases such as cancer.

Course Objectives

Broadly speaking, CS5200 has five main objectives:

1. Foundational concepts: Terminology related to databases and database software

2. Designing and building a *relational* database: How to model the "real world" as a collection of tables.

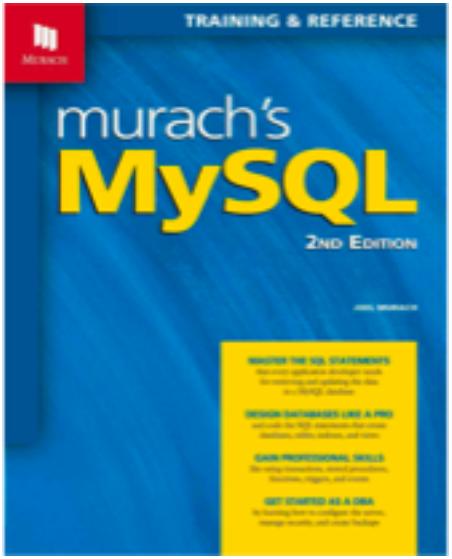
3. Interacting with databases via SQL: building tables, manipulating data, querying and analysis

4. Developing database applications culminating in a student project

5. NoSQL: Beyond the relational model

Textbooks

Database Systems A practical approach to design, implementation and management (6th edition) by Thomas Connolly and Carolyn Begg



Murach's My SQL by Joel Murach (2nd edition)

Grading

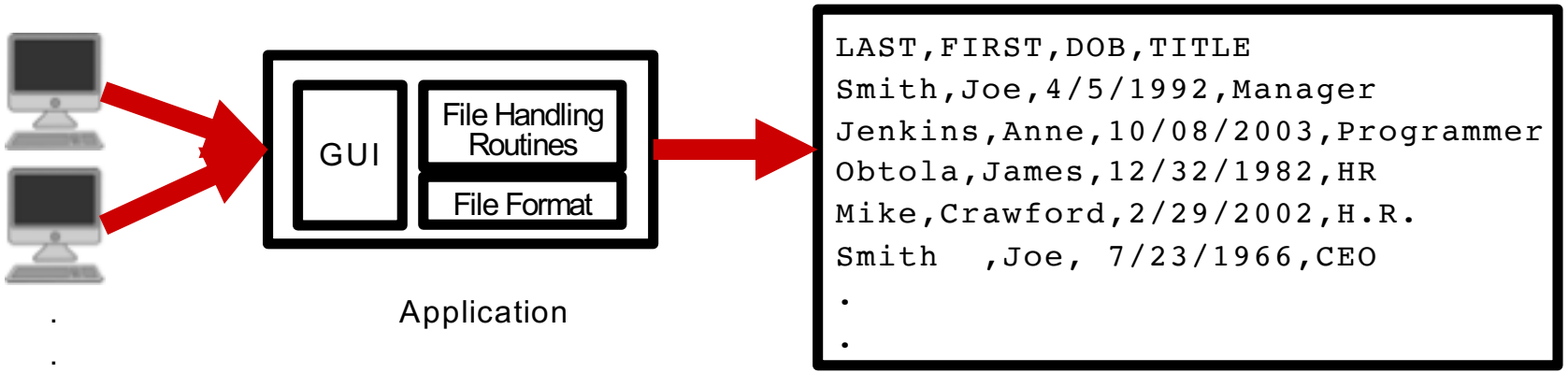
- 5 Homework assignments (30%)
- 5 Quizzes (30%)
- Class Project (30%)
- Participation (10%)

Before Databases: File-Based Systems

Problem: You need to build a human-resources (HR) application to manage information about your employees.

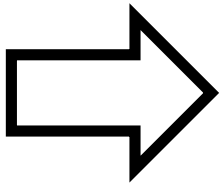
User requirements:

- Search ("Show me information about Joe Smith")
- Add / Delete ("Enter data about Anne Jenkins, our new hire. Delete Saniya Lee, she retired")
- Update ("James Obatola was promoted, update his title.")



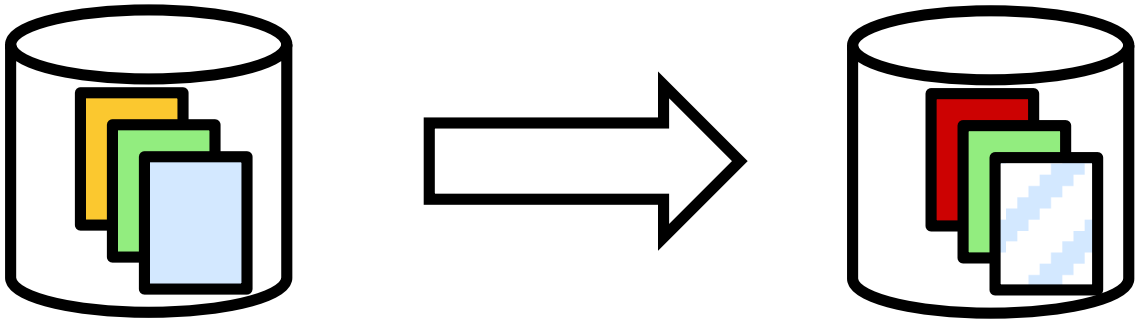
Problems with File-Based Systems

Isolated data is more difficult to access and integrate.
More burden on application developer.



Problems with File-Based Systems

Decentralization leads to duplication.
Why is this bad?



SALES DEPT.

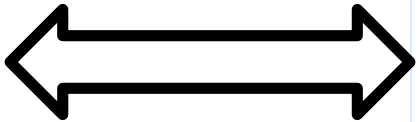
LEGAL DEPT.

Problems with File-Based Systems

Data dependence / Incompatible File Formats:
Applications are written with a particular file format in mind.

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<response uri="http://fake-url.com">
   <result>
      <Accounts>
         <row no="1">
            <FL val="id">1242160000000072037</FL>
            <FL val="phone"><![CDATA[null]]></FL>
            <FL
val="website"><![CDATA[www.joshuawyse.com]]></FL>
            <FL val="employees"><![CDATA[0]]></FL>
            <FL val="billingStreet"><![CDATA[null]]></FL>
            <FL val="shippingStreet"><![CDATA[null]]></FL>
            <FL val="billingCity"><![CDATA[null]]></FL>
            <FL val="shippingCity"><![CDATA[null]]></FL>
            <FL val="billingState"><![CDATA[null]]></FL>
            <FL val="shippingState"><![CDATA[null]]></FL>
            <FL val="billingCode"><![CDATA[null]]></FL>
            <FL val="shippingCode"><![CDATA[null]]></FL>
            <FL val="billingCountry"><![CDATA[null]]></FL>
            <FL val="shippingCountry"><![CDATA[null]]></FL>
            <FL val="description"><![CDATA[null]]></FL>
         </row>
      </Accounts>
   </result>
</response>
```

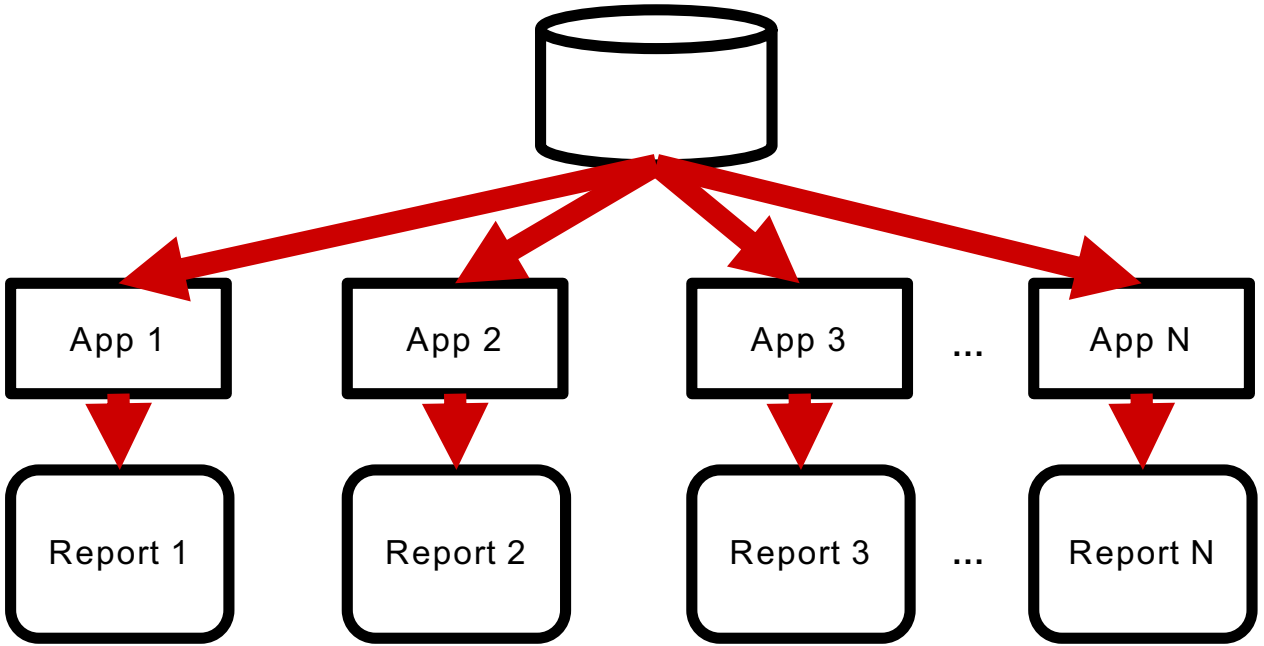


JSON:

```
{
    "id": "1242160000000072038",
    "description": "3",
    "website": "3",
    "numberOfEmployees": "3",
    "phone": "3",
    "name": "account3",
    "shippingAddress": {
        "country": "3",
        "stateOrProvidence": "3",
        "city": "3",
        "postalCode": "3",
        "street1": "3"
    },
    "billingAddress": {
        "country": "3",
        "stateOrProvidence": "3",
        "city": "3",
        "postalCode": "3",
        "street1": "3"
    }
}
```

Problems with File-Based Systems

Fixed Queries (no *ad hoc* exploration) → App Proliferation



The rise of the database

Database technologies were created because:

- **Definition of data was embedded in application programs, rather than being stored separately and independently.**
- **No control over access and manipulation of data beyond that imposed by application programs.**

The Solution:

The database and Database Management System (DBMS).

Codd, 1970.

Edgar Frank Codd
1923-2003

Information Retrieval

A Relational Model of Data for Large Shared Data Banks

E. F. Codd
IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

History of Databases

1970	E.F. Codd (IBM) and the relational model
70's	Early relational database implementations
1976	The Entity-Relationship Model: A tool for designing databases
1979	Oracle (#1) DB2 by IBM (#2)
1980	SQL Standards emerge (there are still many dialects)
1990's	The growing importance of data-mining, machine learning, and the Internet
1995	MySQL
2000's	Beyond the relational database, Big Data, NoSQL
2008	Sun Microsystems acquires MySQL
2010	Oracle acquires Sun Microsystems

What is a Database?

- Shared collection of logically related data (and a description of this data), designed to meet the information needs of an organization.
- System catalog (metadata) provides description of data to enable program–data independence.
- Logically related data comprises entities, attributes, and relationships of an organization's information.
- In a relational database, the data is represented as a collection of tables.

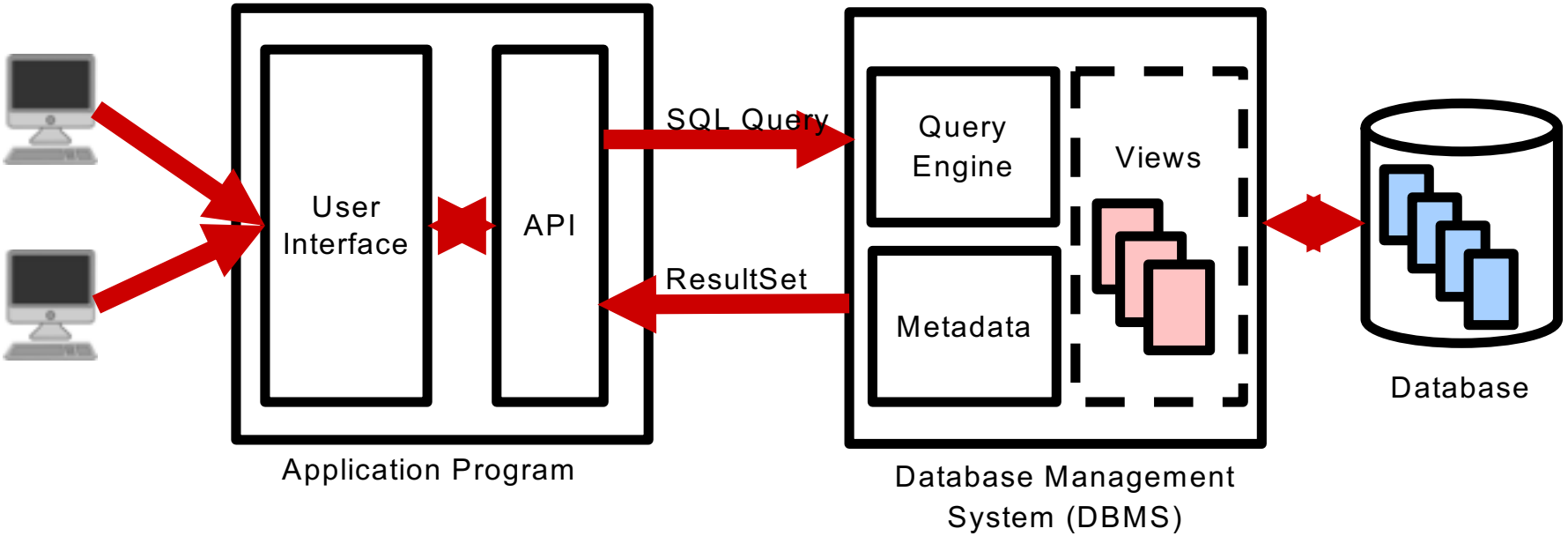
Database Management System (DBMS)

- A software system that enables users to define, create, maintain, and control access to the database.

- (Database) application program: a computer program that interacts with database by issuing an appropriate request (SQL statement) to the DBMS.

- A *Relational* Database Management System (RDBMS or RDB) uses the relational model: i.e., it uses tables to represent entities. This remains the dominant model in industry today, but other approaches have emerged.

Database Application

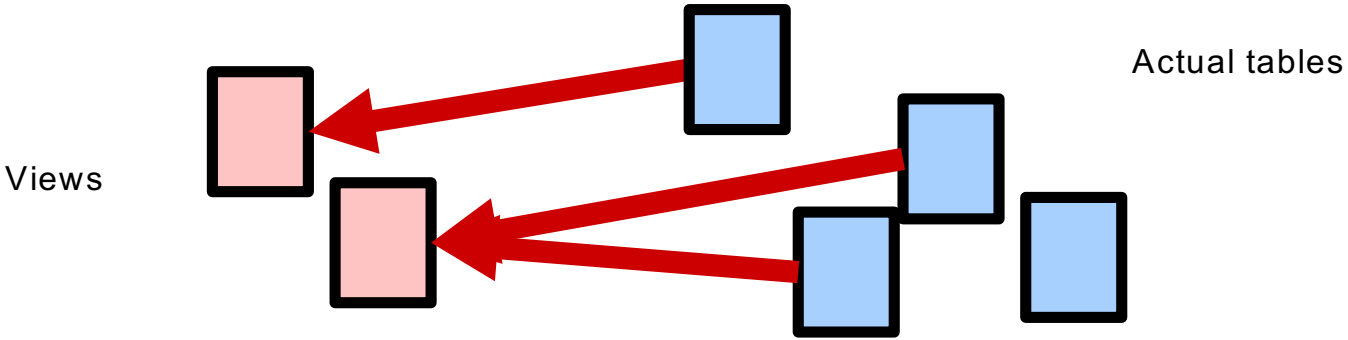


(Usually running on a separate server)

Views

A view provides access to essentially some subset of the database. It allows each user to have his or her own view of the database. Benefits include:

- Reduce complexity
- Provide an additional level of security
- Provide a mechanism to customize the appearance of the database
- Present a consistent, unchanging picture of the structure of the database, even if the underlying database is changed



Advantages of DBMSs

- Control of data redundancy
- Data consistency
- More information from the same amount of data
- Sharing of data
- Improved data integrity
- Improved security
- Enforcement of standards
- Economy of scale

Advantages of DBMSs

- Balance conflicting requirements
- Improved data accessibility and responsiveness
- Increased productivity – ad hoc queries
- Improved maintenance through data independence
- Increased concurrency
- Improved backup and recovery services

Disadvantages of DBMSs

- Complexity
- Size
- Cost of DBMS
- Additional hardware costs
- Cost of conversion
- Performance bottlenecks
- Higher impact of a failure

Many of these problems become magnified at “big data” scales.

Roles in the Database Environment

- Data Administrator (DA)
- Database Administrator (DBA)
- Database Designers (Logical and Physical)
- Application Programmers
- End Users (naive and sophisticated)

Modeling the real world

- A database is a model of real world *entities, attributes*, and *relationships*:
- Each particular database carves out a domain of connected or related entities
- Entities are the things we want to model and can include things like:

Employees
Departments
Branches
Companies
Invoices
Products
Customers
Suppliers
Transaction Events
Orders
Invoices

Corporate Database

Genes
Proteins
Biological Pathways
Genomic sequences
Organisms
Diseases
Chemical compounds
Clinical Trials
Assays
Functional Annotations

Bioinformatics Database

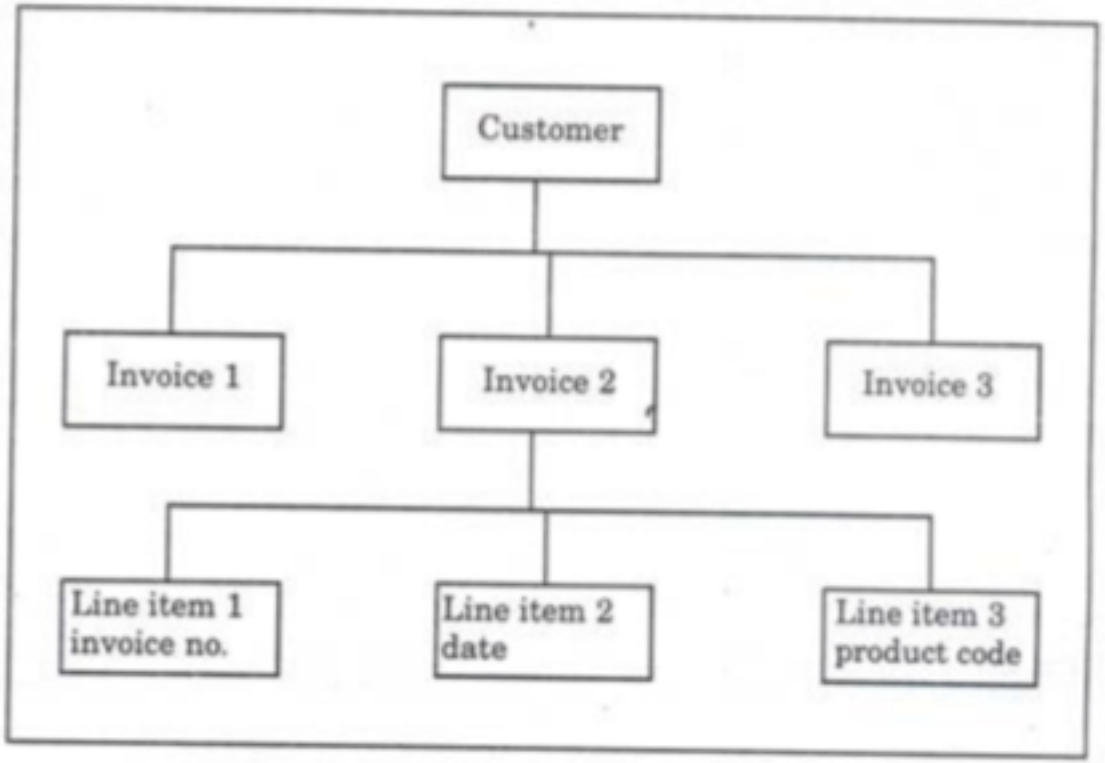
Movies
TV Shows
Actors
Directors
Writers
Ratings
User comments

IMDB

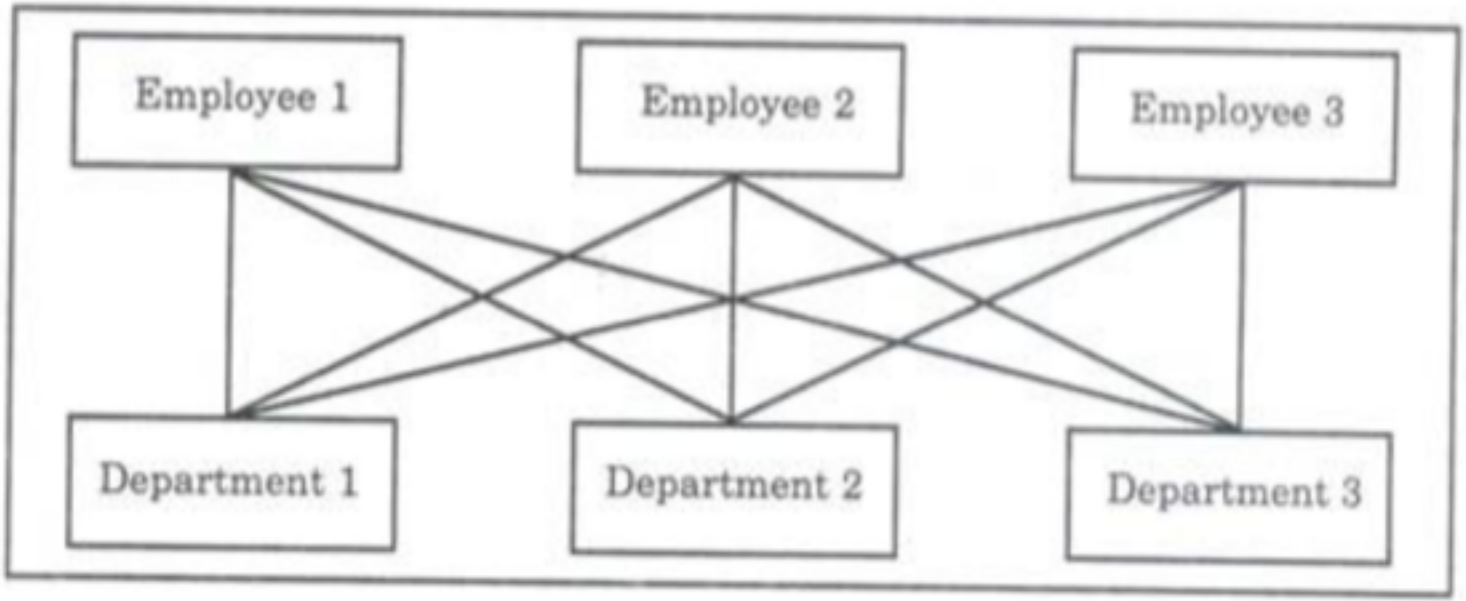
?

ebay

Hierarchical Data Model



Network Data Model



Codd, 1970.

Edgar Frank Codd
1923-2003

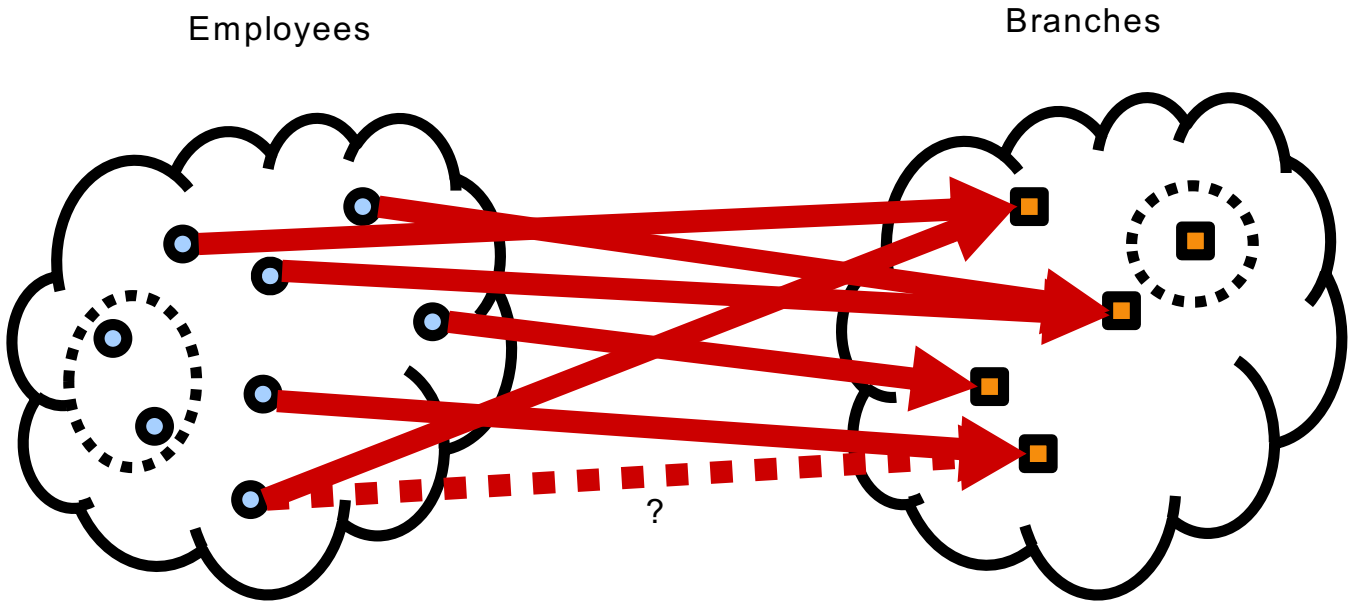
Information Retrieval

A Relational Model of Data for Large Shared Data Banks

E. F. Codd
IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

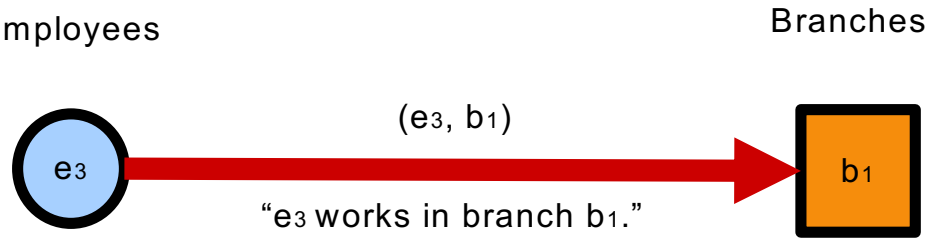
Relationships: Employees and Branches



E = {e_1, e_2, e_3, …. }

B= {b_1, b_2, b_3, …. }

An example: Employees and Branches



- In general, we can model the notion that an employee works in a particular branch with the tuple: (e,b) where e ∈ **E** and b ∈ **B**.
- Let **R** be the set of (e,b) tuples telling us which employees work at which branches. $R \subseteq E \times B$ where E X B is the *Cartesian Product* denoting all possible tuples (e_i, b_j)
- In set theory, **R** is called a **Relation**

Relational Databases (The dominant model)

Employee	Branch
e_2	b_5
e_3	b_5
e_7	b_2
e_{12}	b_9
e_1	b_9
.	.
.	.
.	.

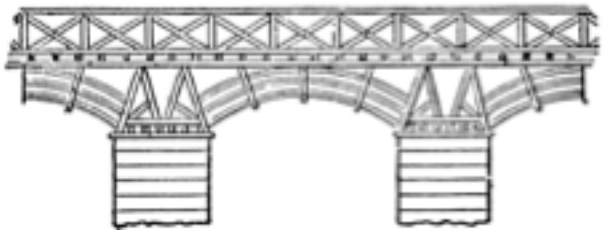
R = { (e_2,b_5), (e_3,b_5), (e_7,b_2), (e_{12},b_9), (e_1,b_9) … }

Key takeaways:

1. Relational databases store data in 2-dimensional tables
2. The tables establish connections between the different entities we want to model
3. The “Relational” in Relational Database (RDB) comes from the concept of a relation in set theory.

Why is creating a database a *design* problem?

Discussion: What does designing a database have in common with, say, designing a bridge, a car, a building, an algorithm, or data structure?



Corollary question:
Why are there so many different kinds of bridges??

Objectives and tradeoffs in database design

Storage efficiency – Have we eliminated unnecessary redundancy?

Query performance – Do our queries run efficiently?

Accuracy - Have we accurately modeled the real-word domain, e.g., the *business rules?*

Versatility - Does the database facilitate answering questions that we want to ask (or maybe some we haven't yet thought of)? When designing a database, it is always good practice to begin by **understanding the requirements**:

- What data do I need to manage?
- What operations do I need to be able to perform?
- What *questions* do I want to be able to ask?
- What insights can I hope to gain?

A Simple Relational Data Model

Name	Salary	Branch
John White	30000	22 Deer Rd, London
Ann Beech	12000	163 Main St, Glasgow
David Ford	18000	163 Main St, Glasgow
Mary Howe	9000	16 Argyll St, Aberdeen
Susan Brand	24000	163 Main St. Glasgow
Julie Lee	9000	22 Deer Rd, London

Discussion: What are the disadvantages of this design?

A Better Relational Data Model

Branch

branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

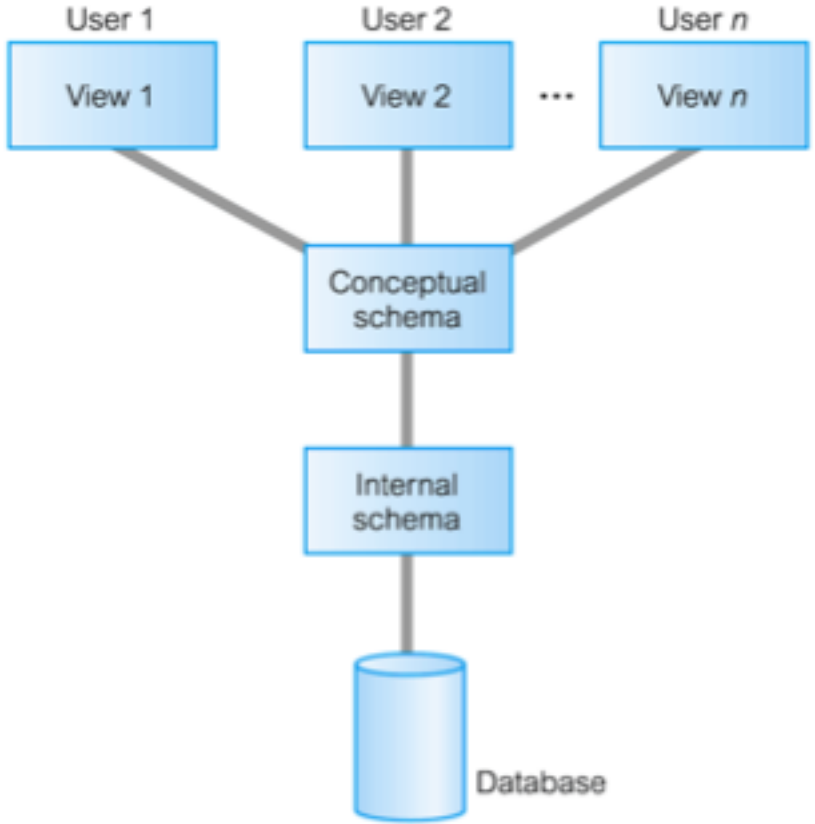
ANSI-SPARC Three-Level Architecture

Database Application

DBMS

DBMS

Operating System



End Users

Application programmers

Data administer

Logical DB designer

DB administer

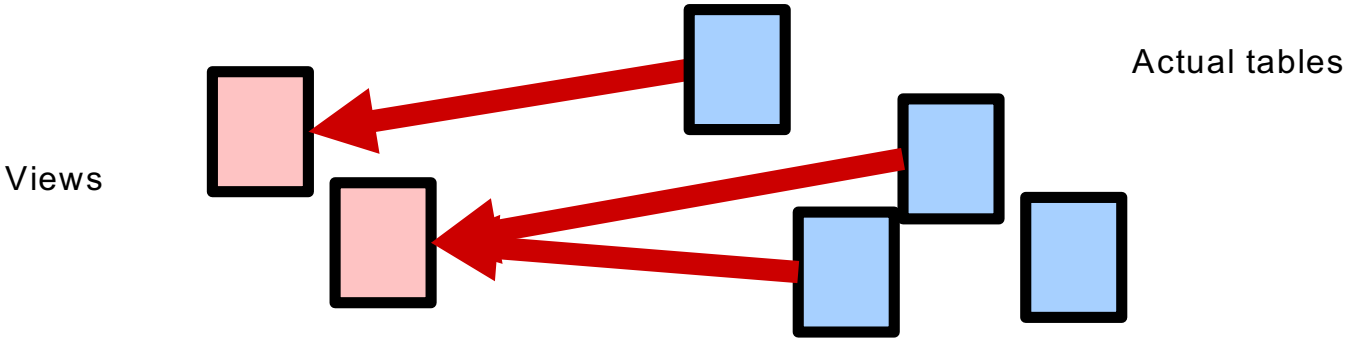
Physical DB designer

File Storage

Views

A view provides access to essentially some subset of the database. It allows each user to have his or her own view of the database. Benefits include:

- Reduce complexity
- Provide an additional level of security
- Provide a mechanism to customize the appearance of the database
- Present a consistent, unchanging picture of the structure of the database, even if the underlying database is changed



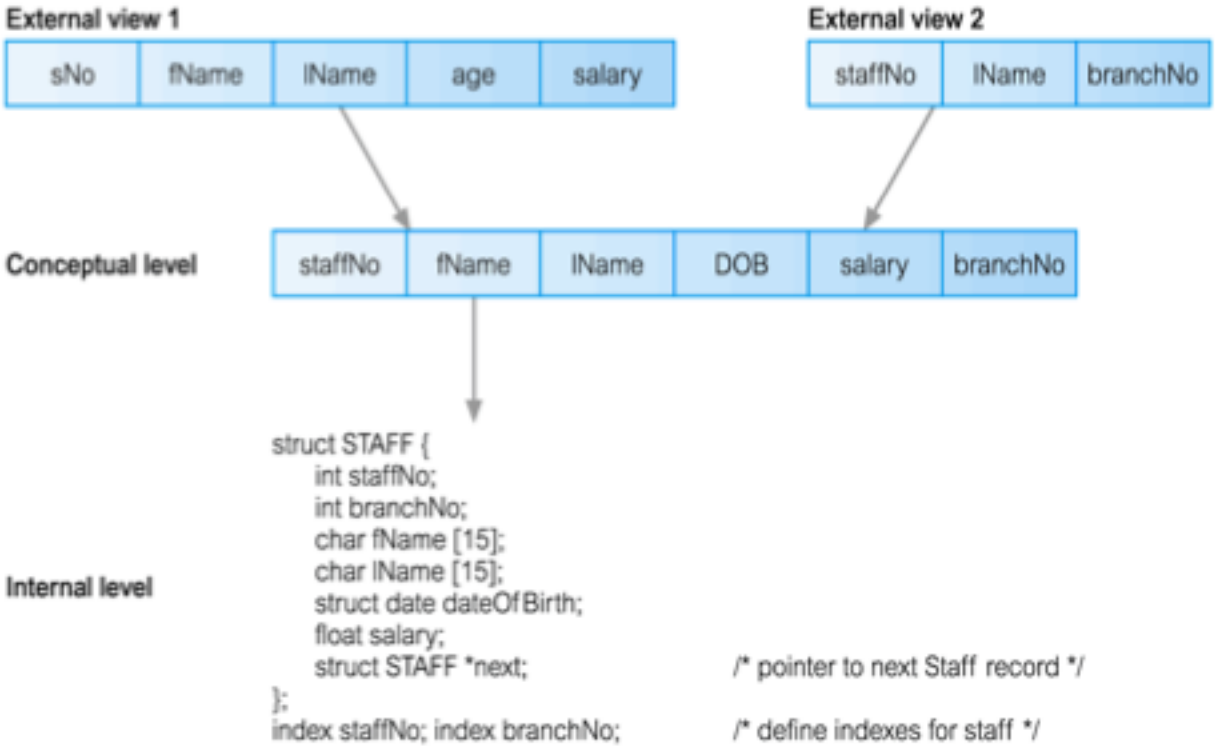
ANSI-SPARC Three-Level Architecture

External Level: The users' view of the database. Describes that part of database that is relevant to a particular user.

Conceptual Level: Community view of the database. Describes what data is stored in database and relationships among the data. (The "Schema")

Internal Level: Physical representation of the database on the computer. Describes how the data is stored in the database.

Differences between Three Levels of ANSI-SPARC Architecture



Benefits of Three-Level Architecture

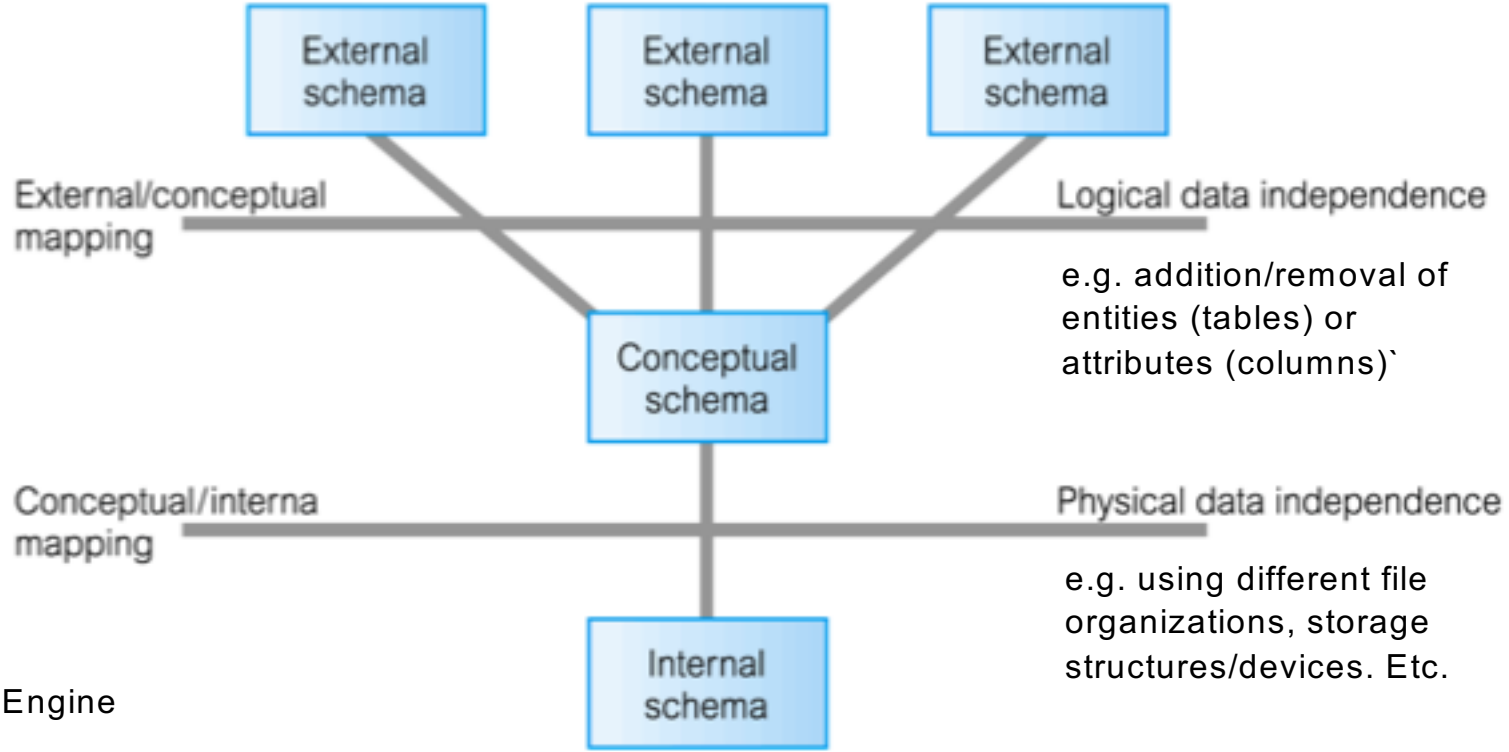
- All users should be able to access same data but have a customized view.
- A user's view is immune to changes made in other views.
- Users should not need to know physical database storage details.

ANSI-SPARC Three-Level Architecture

The View

The Schema

The Storage Engine



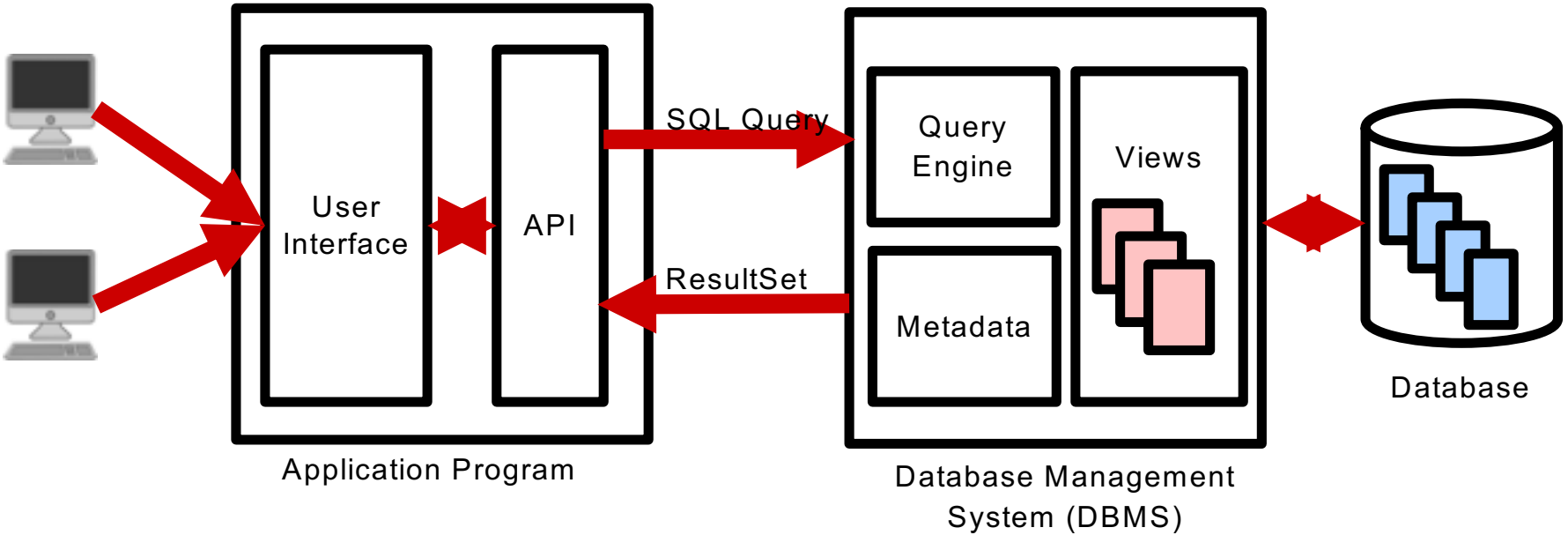
Benefits of Three-Level Architecture

- **Physical data independence**: DBA should be able to change database storage structures without affecting the users' views. (Might impact performance, but not application logic.)

- **Logical data independence:** DBA should be able to change conceptual structure of database without affecting all users.

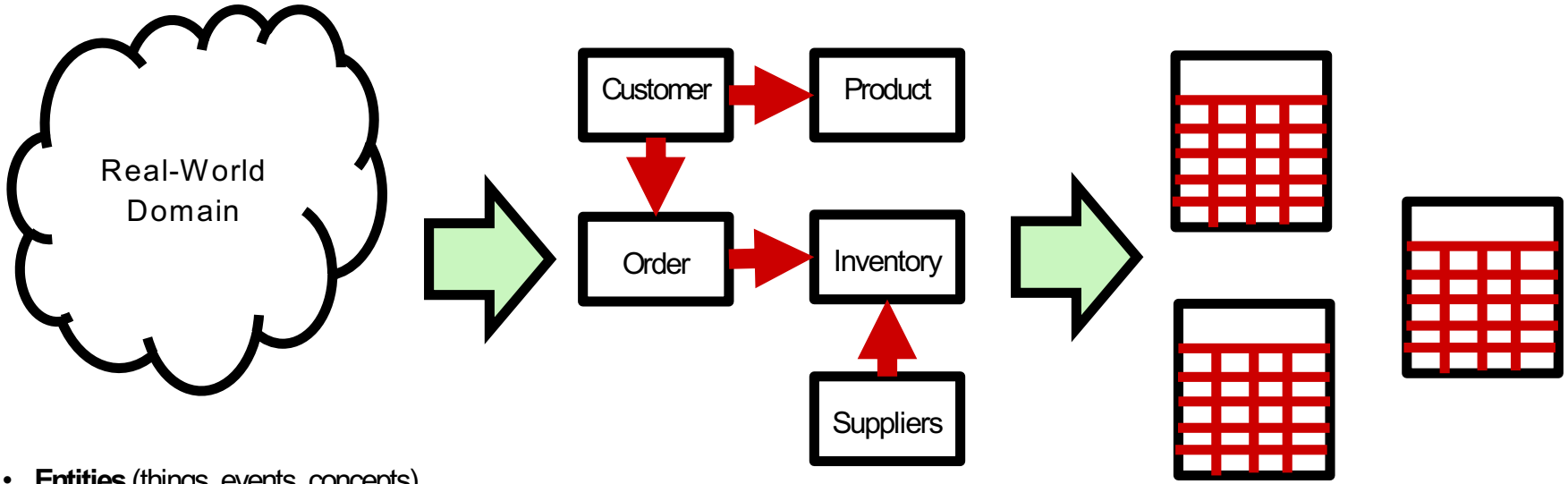
- **Platform independence**: Internal structure of database should be unaffected by changes to physical aspects of storage.

Database Application



(Usually running on a separate server)

The Modeling Process

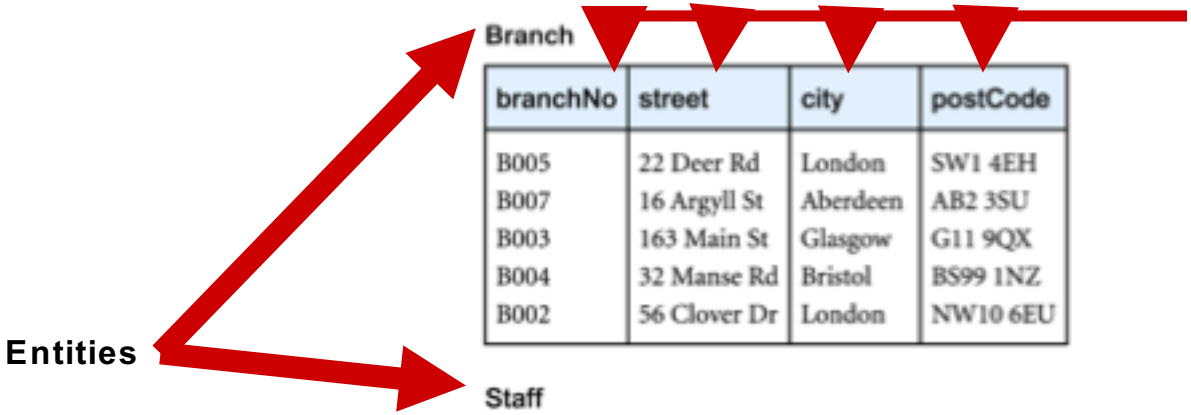


- **Entities** (things, events, concepts)
- **Attributes** (annotations on entities)
- **Relationships** (connections or associations between entities)

Entity-Relationship Diagram
(A *conceptual* data model)

Relational Data Model
(A *logical* data model)

Relational Data Model



Branch

branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

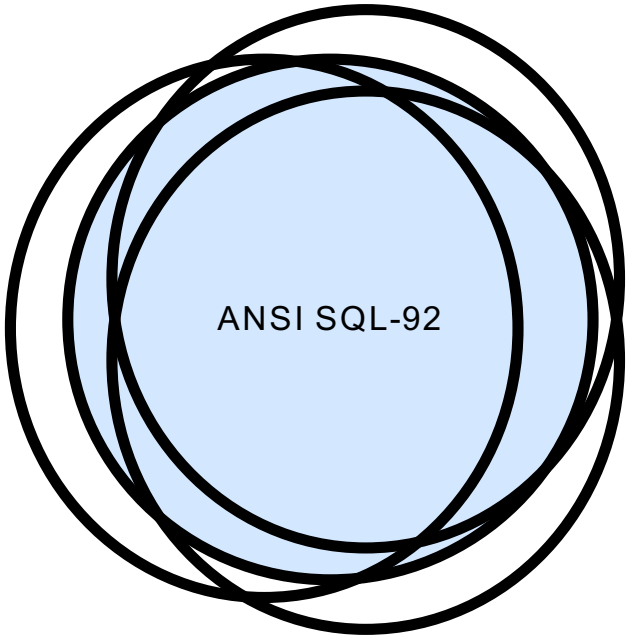
A column or ***attribute*** or **field** providing more details about the particular entity.

- Name
- Datatype
- Other attributes

A row or ***record*** describing one instance of the entity

SQL: The language of databases

- SQL = Structured Query Language
- Pronounced *either* S.Q.L. or *See*-quel (both are perfectly acceptable.)
- The language was *standardized* over time (SQL-86, SQL-89, SQL-92) but most vendors do not implement the standard completely or they introduce vendor-specific features / dialects.



One standard, many dialects

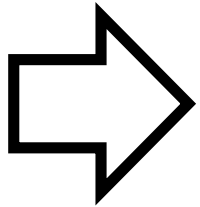
What is *structured* about SQL?

Simplified syntax for selecting data:

```
SELECT  <Columns to retrieve>
FROM    <Source tables>
WHERE   <Conditions on individual rows>
ORDER BY  <Sorting criteria>
```

An example of how SQL is a declarative language

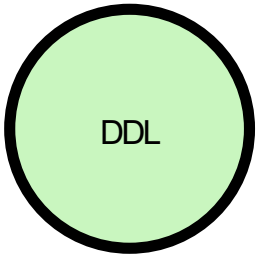
```
SELECT  invoice_id, invoice_total
FROM    invoices
WHERE   invoice_total > 100.0
ORDER BY  invoice_total
```



	invoice_id	invoice_total
▶	1	125.00
	4	2199.99

The Language Hierarchy

Data Definition Language

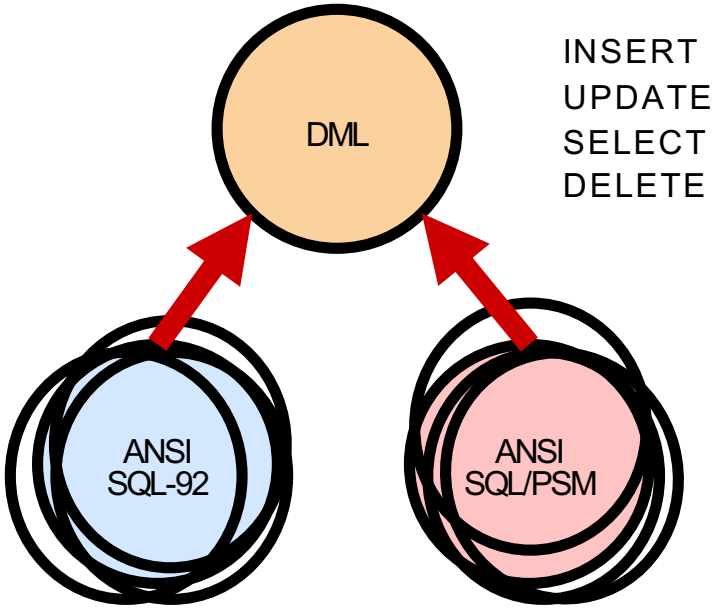


CREATE
DROP
ALTER

- Operations that create or modify the database Schema

- Metadata-focused
- Allows the DBA or user to describe and name entities, attributes, and relationships required for the application plus any associated integrity and security constraints.

Data Manipulation Language



INSERT
UPDATE
SELECT
DELETE

Declarative (4GL)

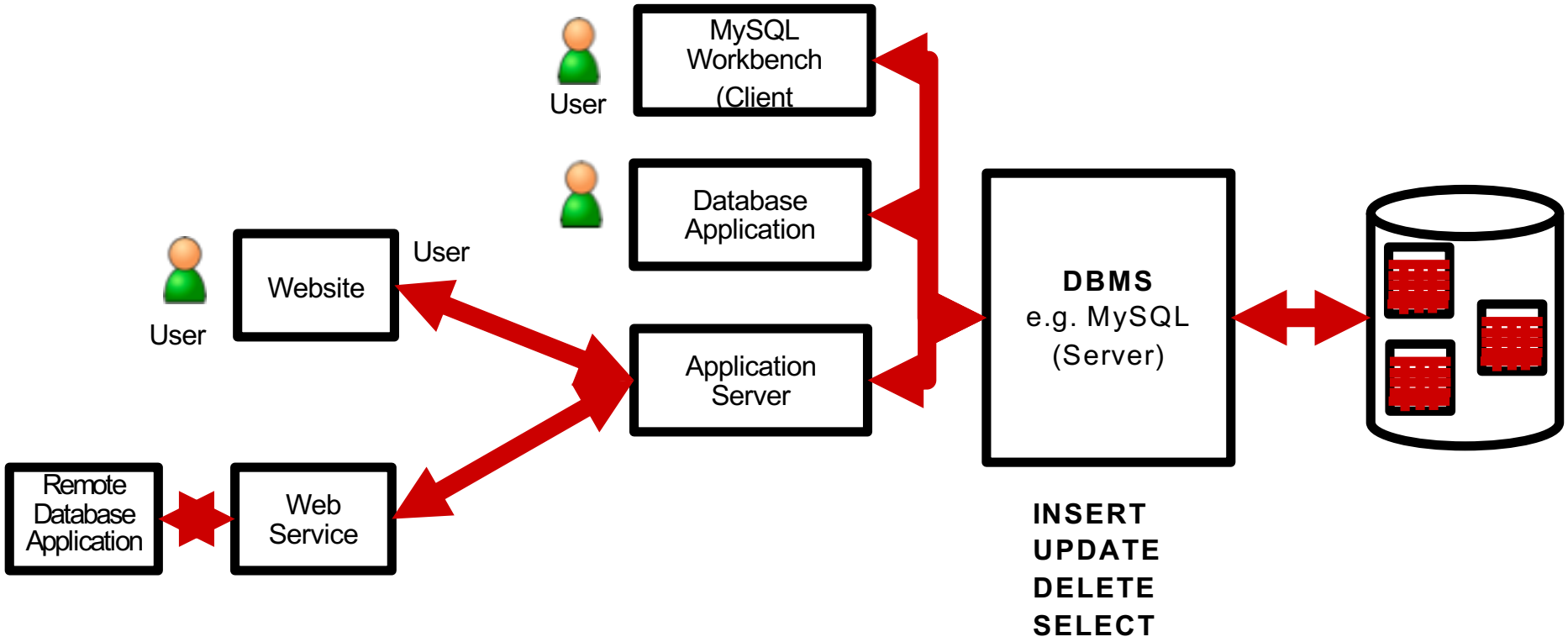
Procedural (3GL)

Procedural Languages for DB are numerous

Source	Common name	Full name
ANSI/ISO Standard	SQL/PSM	SQL/Persistent Stored Modules
Interbase / Firebird	PSQL	Procedural SQL
IBM DB2	SQL PL	SQL Procedural Language (implements SQL/PSM)
IBM Informix	SPL	Stored Procedural Language
IBM Netezza	NZPLSQL[18]	(based on Postgres PL/pgSQL)
Microsoft / Sybase	T-SQL	Transact-SQL
Mimer SQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MySQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MonetDB	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
NuoDB	SSP	Starkey Stored Procedures
Oracle	PL/SQL	Procedural Language/SQL (based on Ada)
PostgreSQL	PL/pgSQL	Procedural Language/PostgreSQL Structured Query Language (implements SQL/PSM)
Sybase	Watcom-SQL	SQL Anywhere Watcom-SQL Dialect
Teradata	SPL	Stored Procedural Language
SAP	SAP HANA	SQL Script

Source:
https://en.wikipedia.org/wiki/SQL

Data Storage, Retrieval, and Update



Access to metadata

- **Metadata** is data about data. It is pretty much everything but the data itself.
- Sometimes called the **data dictionary** or the **system catalog**
- Things we find in the system catalog include

  - Names of available databases
  - The names of each table in each database
  - The name and type of each column in each table and any column-specific constraints (Uniqueness, non-empty, etc.)
  - Referential constraints (The values in column X of table A must occur in column Y of table B)
  - Users and access privileges.
  - Usage statistics, open connections, etc.

Branch

branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Metadata

Data

Transaction Support

- A **transaction** is a set of updates that must either succeed together or fail together. If only some of the updates were to succeed, the database could be left in an inconsistent state.

- If transaction fails, the database is **rolled back** to its original state.

Examples:

Insert a new staff member into the staff table.

Remove a staff member and re-assign the properties she managed to other staff members.

Recovery Services

- Database hardware can fail: Servers, disk drives, and software can crash.
- Rollbacks need to occur in the event of these types of failure.
- Backups are your friend.

```
$ mysqldump --user root –p myprojectdb > myproject_01DEC017.sql
```

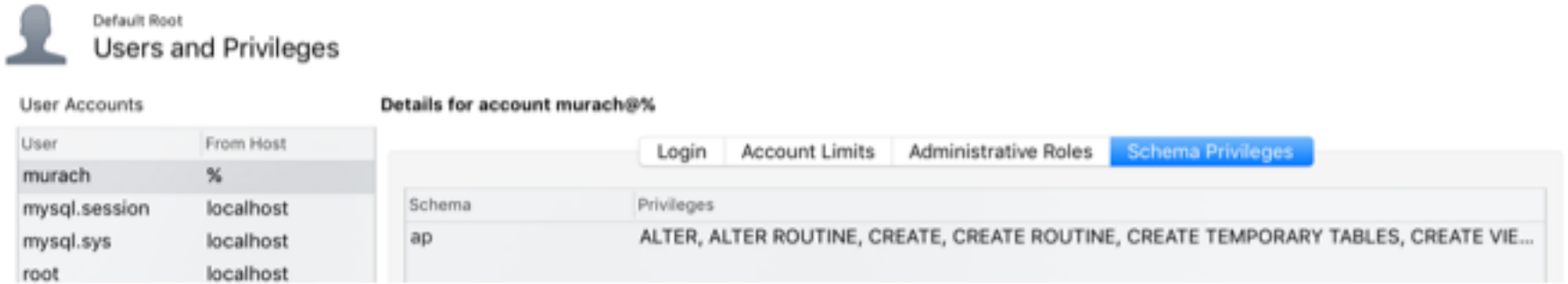
Concurrency Support

- Allow multiple users to perform changes to the database simultaneously
- The *lost update* problem: Multiple users sharing a single bank account doing deposits and withdrawals at different ATMs at the same time.

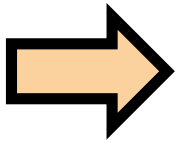
Time	Withdraw $50	Deposit $50	Balance
t_1		x = readBalance()	$100
t_2	x = readBalance()	x = x + 50	$100
t_3	x = x - 50	writeBalance(x)	$150
t_4	writeBalance(x)		$50
t_5			$50

Authorization / Access Control

- Only authorized users can access a database
- Specific kinds of privileges are **granted** or **revoked**



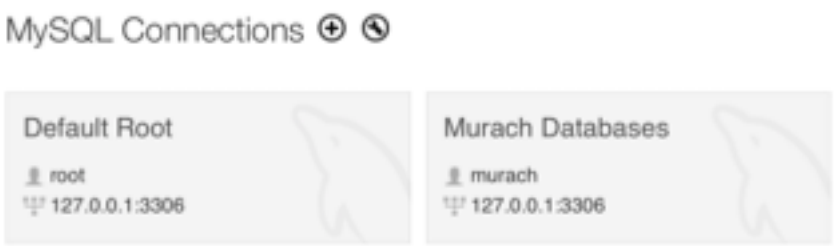

```
SELECT *
FROM mysql.user
```

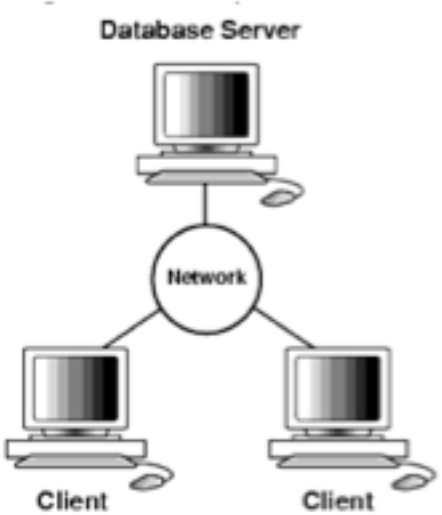


Error Code: 1142. SELECT command denied to user 'murach'@'localhost' for table 'user'

Data Communication Support

- Users should be able to connect to a remote database over a network
- This is called *distributed processing* (upcoming lectures)





Database Integrity

- Stored data must be consistent and correct
- Operations on the data must adhere to specified constraints
- These constraints are often derived from an understanding of the **business rules** and are reflected in the database design.

Some examples of constraints supported by MySQL:

NON NULL: A value must be provided for a particular field
ENUM/SET: The value must be one of the specified values
Datatype: The stored value must adhere to the datatype of the column
Uniqueness: Values in the column must be unique (no duplicates!)
Primary Key: Primary keys must be unique and non-null
Foreign Key: The entity referenced in another table must exist

Services to promote data independence*

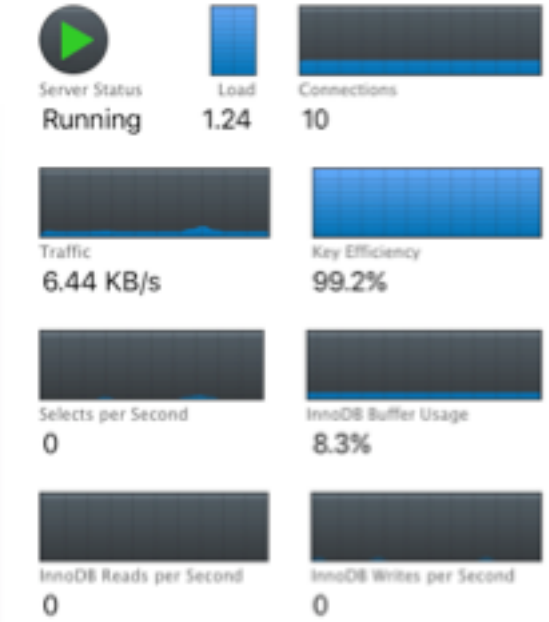
- A DBMS must include facilities to support the independence of programs from the actual structure of the database
- Often achieved by having support for VIEWS
- Good application design is key. Applications should have a separate and well-defined API for **object-relational mapping** to ensuring that underlying changes to the database design (the relations) have minimal impact on the application components (the objects).

* Supplemented by Connolly and Begg

Utility Services*

- May be third-party tools or provided by the vendor

Service	Supported by MySQL Workbench	Alternative MySQL Utility
Import / Export from flat files (.csv for example) and general backup	YES	mysqlimport mysqldump Mysqlpumpdf (5.7.8+)
Usage monitoring	YES	-
Statistical analysis	YES	-
Validation / Repair	YES	mysqlcheck
Garbage collection	YES	mysqlcheck



* Supplemented by Connolly and Begg

MySQL Workbench Dashboard

A simplified view of the history of computing

Early Electronic Computing (1950-1980)

The PC Revolution (1980-1995)

The Internet Revolution (1995 - 2010)

The Smartphone Revolution (2007-present)

The AI / Cloud Revolution (2010 -)

Database Architectures- Objectives

- The meaning of the client–server architecture and the advantages of this type of architecture for a DBMS.
- The difference between two-tier, three-tier and *n*-tier client–server architectures.
- Distributed DBMSs
- About cloud computing and data as a service (DaaS) and database as a service (DBaaS).
- Software components of a DBMS.

Multi-user DBMS Architectures

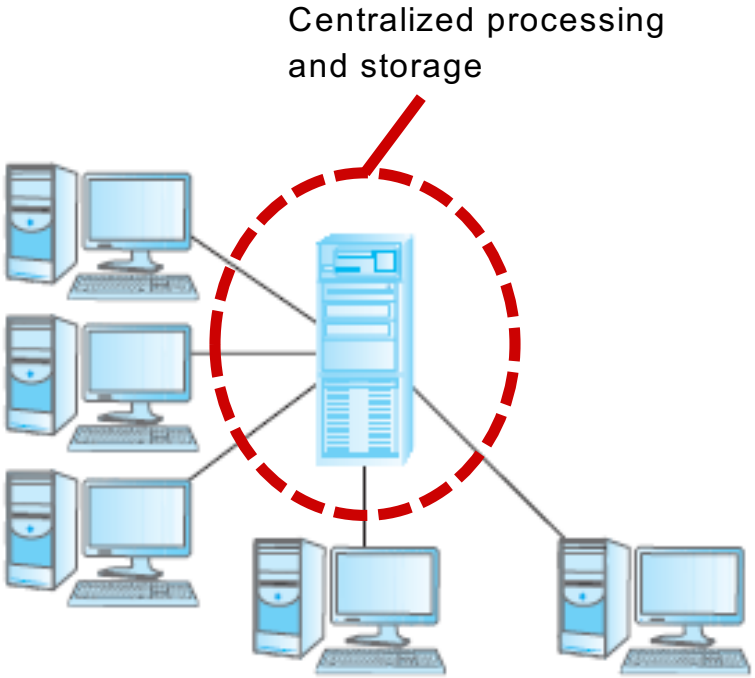
As we saw last time, supporting multiple concurrent users is a key requirements for a DBMS.

The common architectures that are used to implement multi-user database management systems include:

- Teleprocessing
- File-Server
- Client-Server

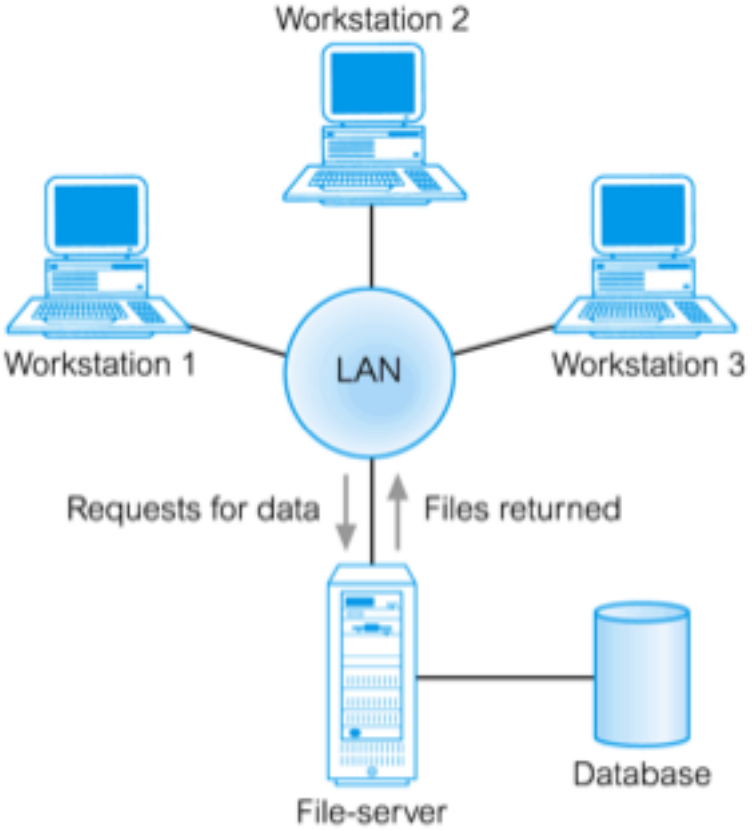
Teleprocessing

- One computer with a single CPU and a number of terminals.
- Processing performed within the same physical computer. User terminals are typically “dumb”, incapable of functioning on their own, and cabled to the central computer.
- Central computer does everything: DBMS, display formatting.



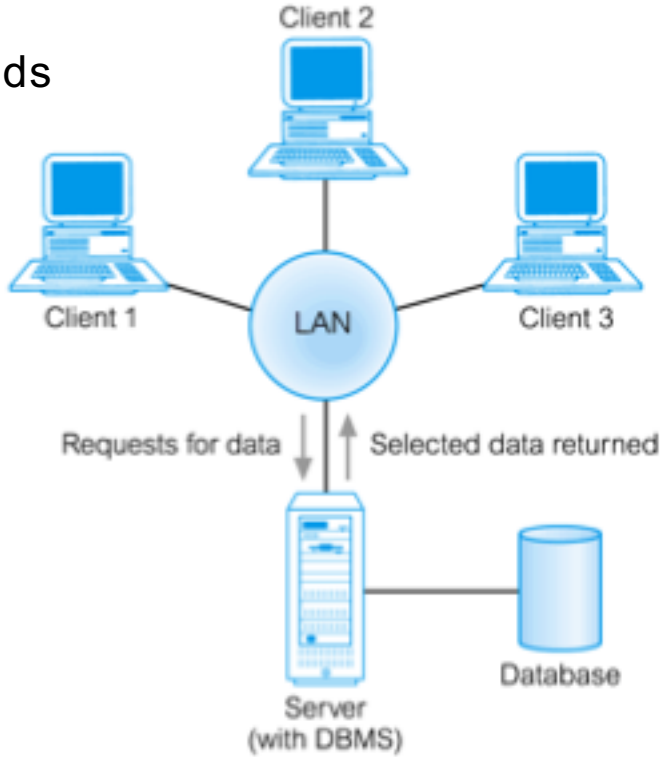
File-Server

- File-server is connected to several workstations across a network.
- Database resides on file-server.
- DBMS and applications run on each workstation.
- Centralized storage, distributed processing
- Disadvantages include:
  - Significant network traffic.
  - Copy of DBMS on each workstation.
  - Concurrency, recovery and integrity control more complex.



Traditional *Two-Tier* Client-Server

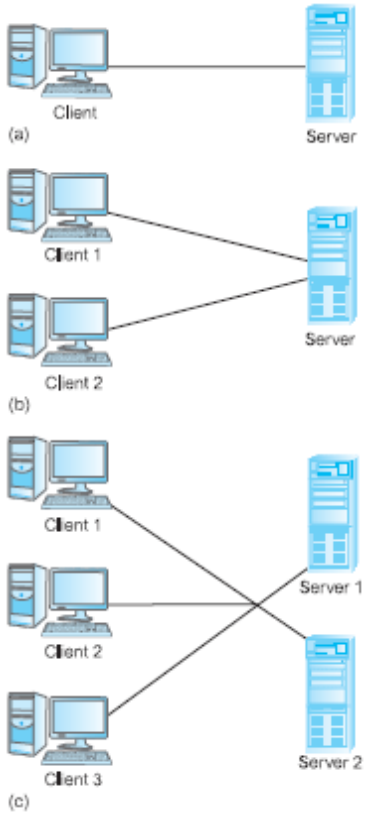
- **Client** (tier 1) manages user interface and runs applications (application logic). **Server** (tier 2) holds database and DBMS.
- Advantages include:
  - wider access to existing databases;
  - increased performance;
  - possible reduction in hardware costs;
  - reduction in communication costs;
  - increased consistency.



Summary of Client-Server Functions

CLIENT	SERVER
Manage user-interface	Accept DB Requests
Accept and check syntax of user input	Check authorization
Generate DB requests and transmits to server	Ensure integrity constraints not violated
Process application logic	Performs query/update processing. Transmits response to client
Passes back response to user	Maintains system catalog. Support concurrency. Etc.

Alternative Client-Server Topologies

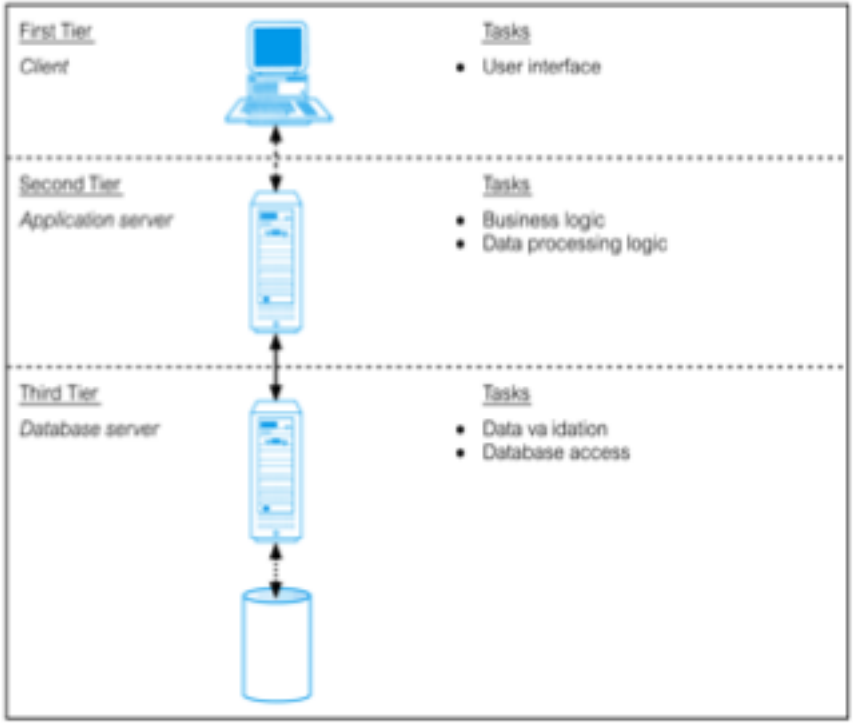


Remaining challenges to scalability:

- 'Fat' client, requiring considerable resources on client's computer to run effectively.
- Significant client side administration overhead. For example, client updates have to be pushed out in some coordinated fashion.

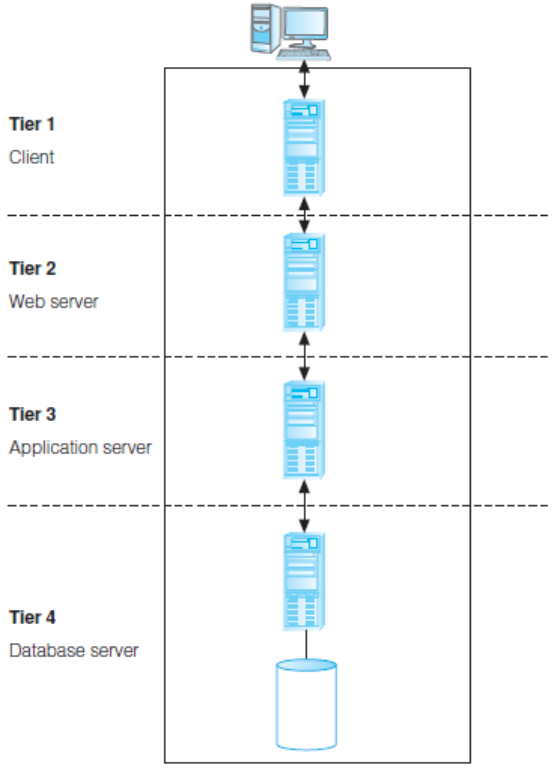
Three-Tier Client-Server

- The need for enterprise scalability challenged the traditional two-tier client–server model.
- By 1995, three layers proposed, each potentially running on a different platform.
- Advantages:
  - ‘Thin’ client, requiring less expensive hardware.
  - Application maintenance centralized.
  - Easier to modify or replace one tier without affecting others.
  - Separating business logic from database functions makes it easier to implement load balancing.
  - *Maps quite naturally to Web environment.*



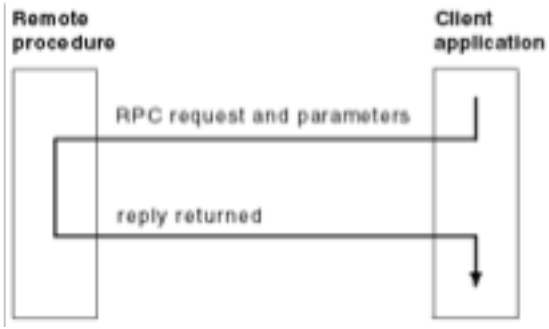
n-Tier Client-Server (e.g. 4-Tier)

- The three-tier architecture can be expanded to *n* tiers, with additional tiers providing more flexibility and scalability.
- Applications servers host API to expose business logic and business processes for use by other applications.
- Web server / Application server is a common separation in n-Tier architectures.

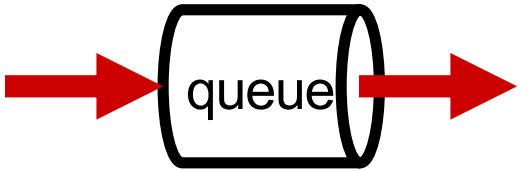


Middleware

Middleware is a generic term used to describe software that mediates with other software and allows for communication between disparate applications in a heterogeneous system. Examples include:

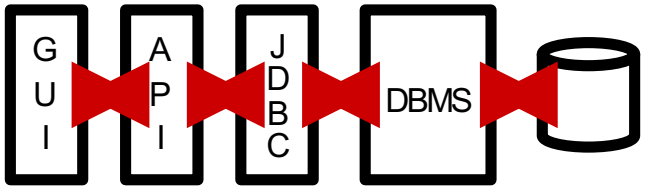


- **Remote Procedure Call (RPC)** is a protocol that one program can use to request a service from a program located in another computer on a network without having to understand the network's details.



- **Messaging Queues**: A **publisher** sends a message to the queue while **subscribers** listen to the queue and respond to incoming messages.

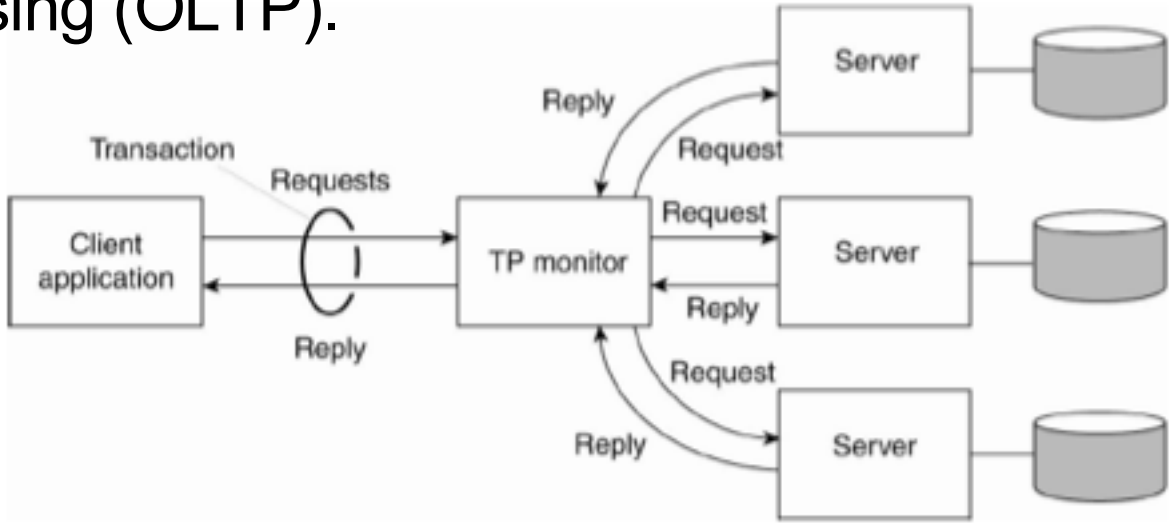
- **SQL-oriented data access** such as **JDBC (Java Database Connectivity)** which provides a common interface for connecting Java to different DBMS's including MySQL.



Transaction Processing Monitors

TP monitor is a program that controls data transfer between clients and servers in order to provide a consistent environment, particularly for online transaction processing (OLTP).

- Load balancing
- Routing

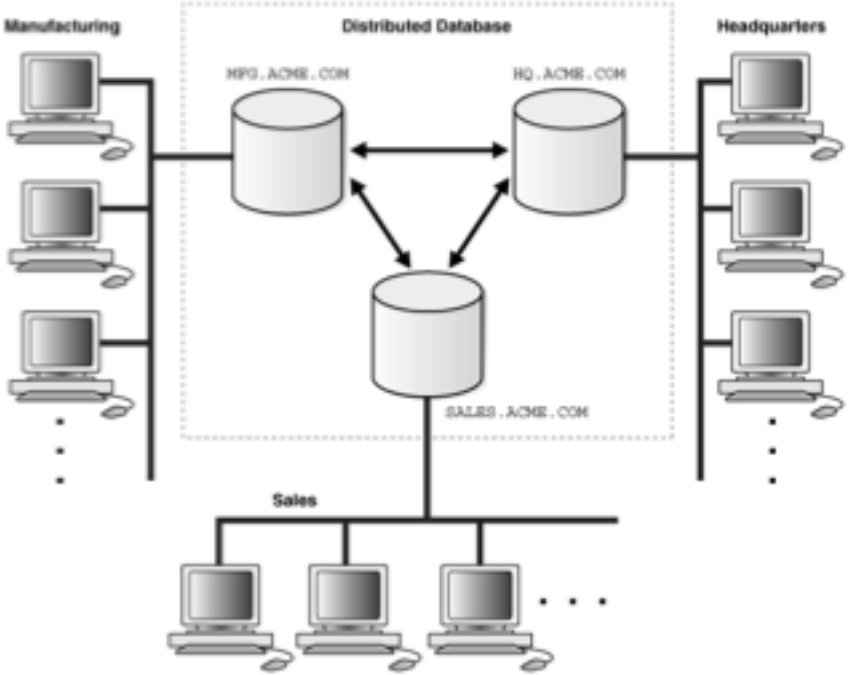


Distributed DBMSs

- A **distributed database** is a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network.

- A **distributed DBMS (DDBMS)** is the software system that permits the management of the distributed database and makes the distribution transparent to users.

- The *fundamental principle* of DDBMSs: Make the distributed nature of the data transparent to the user (i.e., it seems just like a centralized database).



Source: docs.oracle.com

Distributed DBMSs

- A DDBMS consists of a single logical database split into a number *of fragments.*

- Each fragment is stored on one or more computers *(replicas)* under the control of a separate DBMS, with the computers connected by a network.

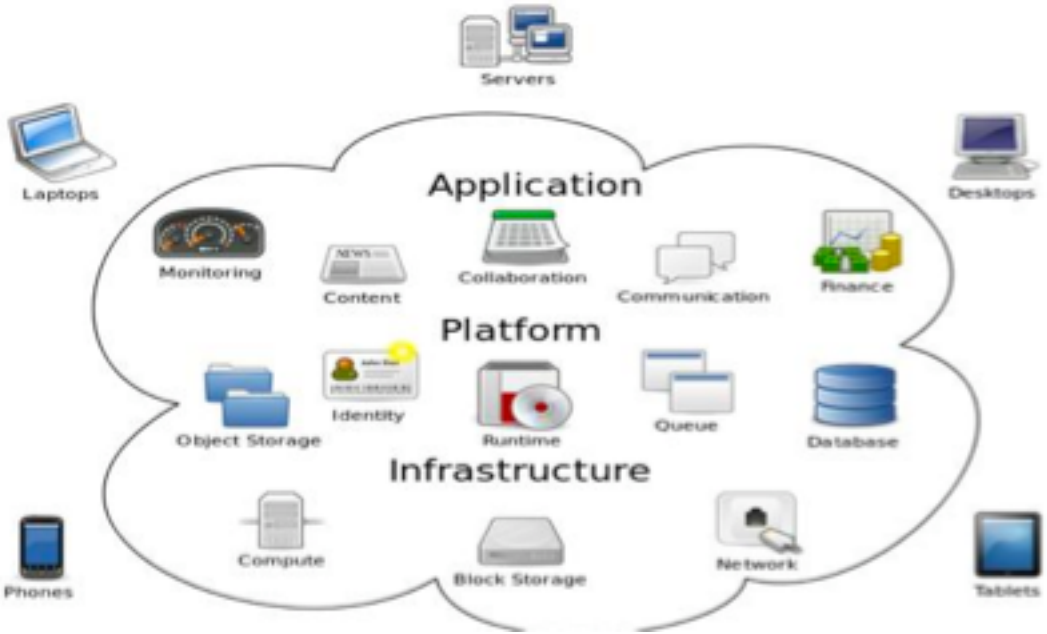
- Each site is capable of independently processing user requests that require access to local data (that is, each site has some degree of local autonomy) and is also capable of processing data stored on other computers in the network.

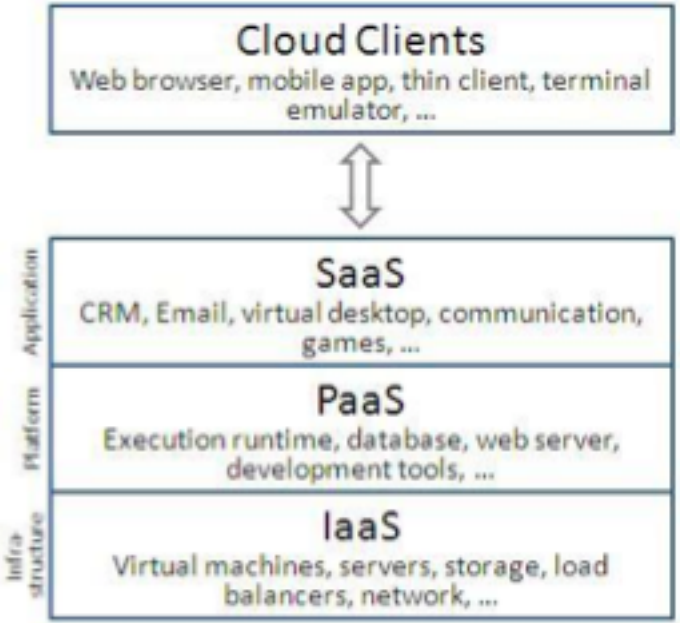
Cloud Computing

A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction (NIST).

The practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer (Dictionary.com).

Cloud Computing





Source: salesforcetutorial.com

Cloud Computing – Key Characteristics

- ***On-demand self-service**:* Consumers can obtain, configure and deploy cloud services without help from provider.

- ***Broad network access**:* Accessible from anywhere, from any standardized platform (e.g. desktop computers, laptops, mobile devices).

- ***Resource pooling**:* Provider's computing resources are pooled to serve multiple consumers. Resources are dynamically assigned and reassigned according to consumer demand.

- ***Rapid elasticity**:* Provider's capacity caters for customer's spikes in demand and reduces risk of outages and service interruptions. Capacity can be automated to scale rapidly based on demand.

- ***Measured service**:* Provider uses a metering capability to measure usage of service (e.g. storage, processing, bandwidth, and active user accounts).

Let’s watch some videos!

- A short TED talk about Cloud Computing

  https://youtu.be/jeOb0rKrt7A

- An introduction to Amazon EC2

  https://youtu.be/TsRBftzZsQo

- Amazon RDS (Relational Database Service)

  https://www.youtube.com/watch?v=yjH10T3Miag

Cloud-based database solutions

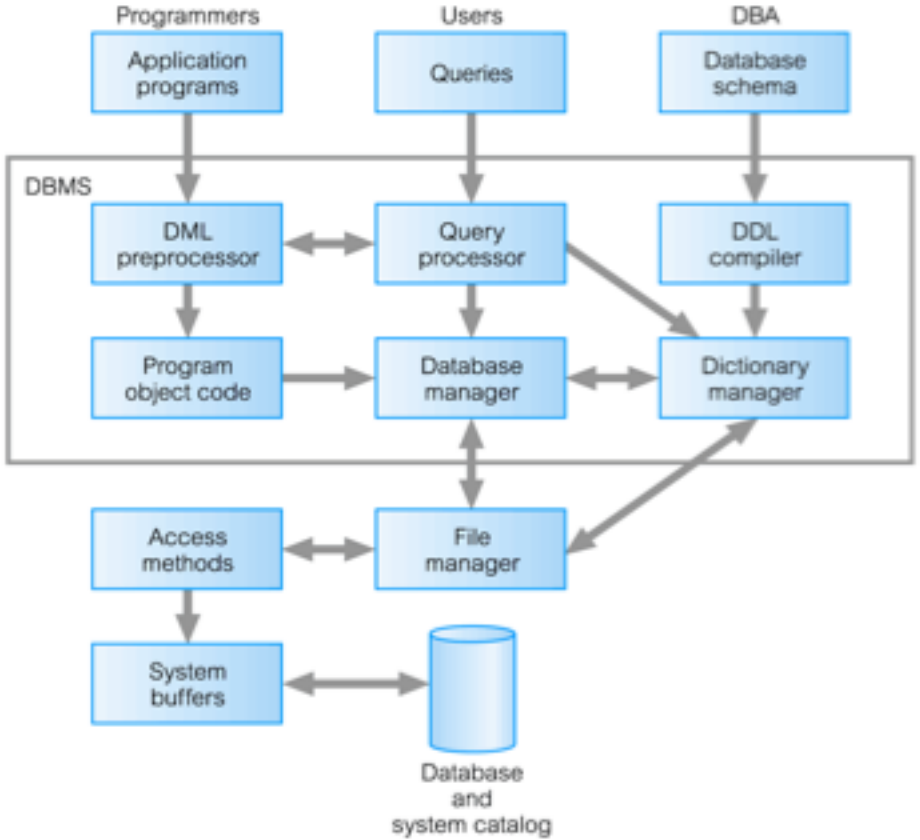
Database as a service (DBaaS)

- Offers full database functionality to application developers.
- Provides a management layer for continuous monitoring and configuring of the database.

Data as a service (DaaS)

- Services provide a data definition in the cloud and access via APIs (rather than SQL)
- Enables organization with valuable data to offer access to others. (e.g., Geomapping, navigation, financial services, business data, etc.)

Components of a DBMS



- **DDL compiler** converts DDL statements into a set of tables (metadata).
- **Dictionary manager** manages the system catalog containing the metadata.
- **Query Processor** converts input queries into low-level instructions.
- **DML Processor** converts DML statements embedded in application to host-specific function calls.
- **Database manager** receives instructions and determines with help of dictionary manager what specific data it needs.
- **File manger** carries out data requests by the database manager.

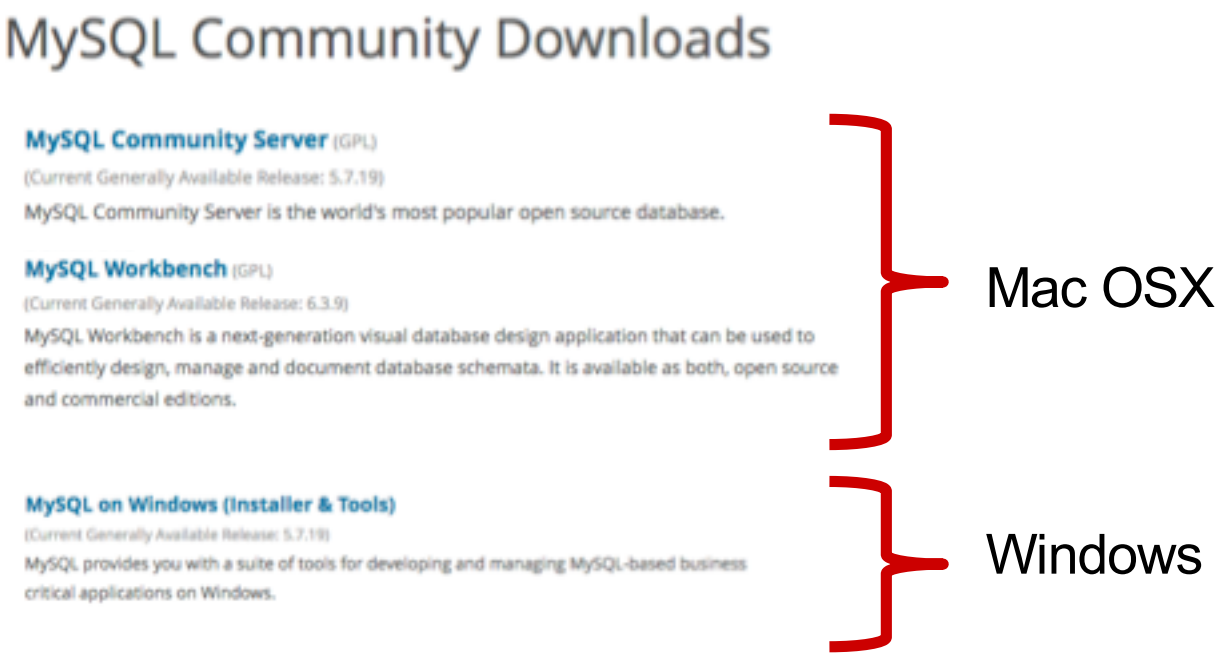
MySQL Setup and Account Configuration

Objectives for today

- Download install the MySQL server and MySQL workbench
- Install Murach's sample database (Posted on blackboard)
- Create a user other than "root" that only has access to the Murach databases. (Why?)
- I have posted a step-by-step set of instructions on blackboard:
  Course Material > MySQL Setup
  > MySQL Installation Step-by-step

Installation

https://dev.mysql.com/downloads/



Mac OSX

Windows