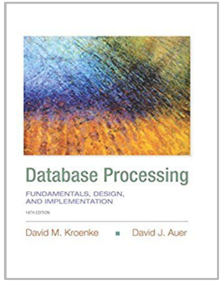
Fairfield University DOLAN School of Business



Databases for Analytics

Kroenke / Auer Chapter 1
Introduction to Database Concepts

Learning Objectives

- **Skills:** You should know how to ...
    - Identify the parts of a database table
    - Use keys to match records from separate tables
- **Theory:** You should be able to explain ...
    - Importance of databases for web and mobile apps
    - Features and components of database systems
    - Difference between mobile, desktop, and enterprise platforms
    - Functions of a DB Management System
    - Terminology like apps, layers, DBMS, SQL, metadata, etc.

Big Picture Stuff

Before we talk about relational databases

Why Study Databases?

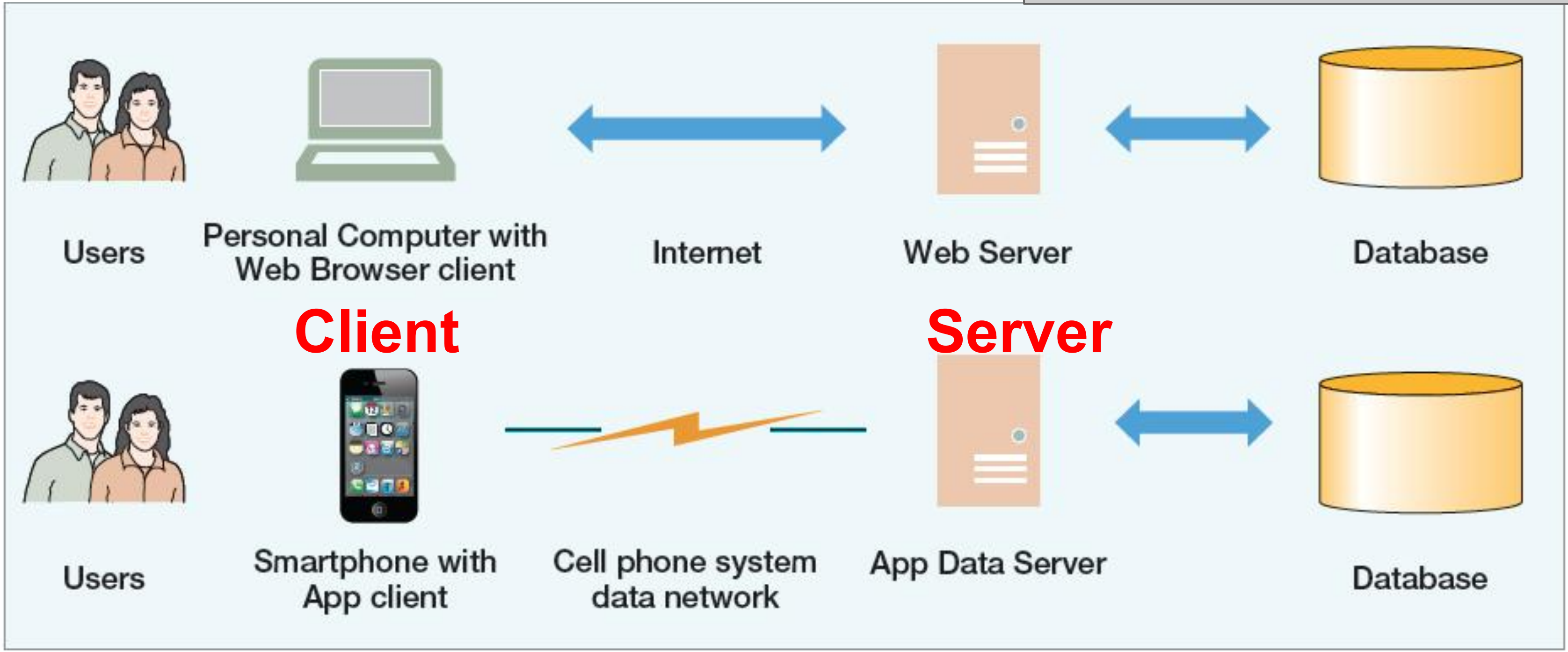
Access to data and information are fundamental to modern business

- Management is about decision making
- Good decisions require information
- Good information requires relevant, accurate, and timely data

Important to understand how databases work and interact with business applications

Enterprise Architecture

Access to data is indirect, through a server process rather than data files. **Every** mobile or web app has this **client-server architecture**.



Data and Information

- Data = raw facts
- Information = Data + Metadata
- Metadata includes things like
  - Meaning of the data
  - Source, timing, and format of data
- Difference is mostly a matter of perspective
  - Information is the *product* of data processing
  - Databases are *designed* to provide information

Implications for Analysts

- **Data *Analysis*** = deriving information from data to support a task or decision
  - Database System → Required Information

We are more focused here ...

- **Database *Design*** = making decisions about how to generate, store, and retrieve data to best support data usage
  - Information Required → Database System

but we also need to know about this as well

What’s a Database? DBMS?

- A database is an integrated, shared *repository* of data + metadata
- A database management system (DBMS)
  - controls access (generation, storage, retrieval) to data and metadata
  - provides facilities to structure the data (with metadata)
- RDBMS vs NoSQL

Relational Databases

Before you go NoSQL like the cool kids, you should know about RDBMSs

What is a Relational Database?

- **Data is stored in tables,** which have rows and columns like a spreadsheet. A database may have multiple tables, where each table stores data about a different kind of entity ('thing').
- **Each row** in a table stores data about an occurrence or instance of the thing of interest. **Each column** ('field') represents an attribute of the thing.
- A database stores **data** (about the things) and **metadata** (data types, relationships, etc.).

Data in Tables

A **primary key** (PK) is a unique identifier within a table.

A **surrogate key** is a PK that is automatically assigned by the DBMS.

StudentNumber is both a PK **and** a surrogate key.

The STUDENT table

The CLASS table

The GRADE table—but who do these grades belong to?

Primary keys are shown in gold.

What's wrong with the GRADE table?

STUDENT

StudentNumber	LastName	FirstName	EmailAddress
1	Cooke	Sam	Sam.Cooke@OurU.edu
2	Lau	Marcia	Marcia.Lau@OurU.edu
3	Harris	Lou	Lou.Harris@OurU.edu
4	Greene	Grace	Grace.Greene@OurU.edu
(New)			

Record: 1 of 4 No Filter Search

CLASS

ClassNumber	ClassName	Term	Section
10	CHEM 101	2014-Fall	1
20	CHEM 101	2014-Fall	2
30	CHEM 101	2015-Spring	1
40	ACCT 101	2014-Fall	1
50	ACCT 101	2015-Spring	1

Record: 1 of 5 No Filter Search

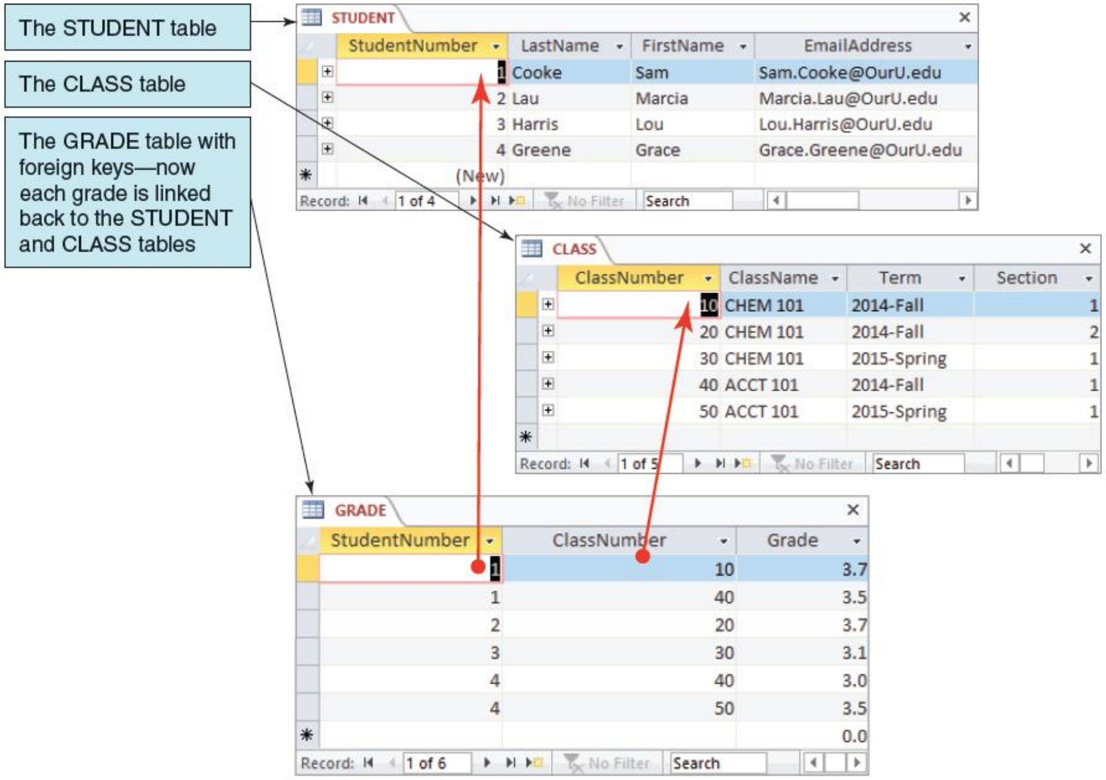
GRADE

Grade
3.7
3.5
3.7
3.1
3.0
3.5
0.0

Record: 1 of 6

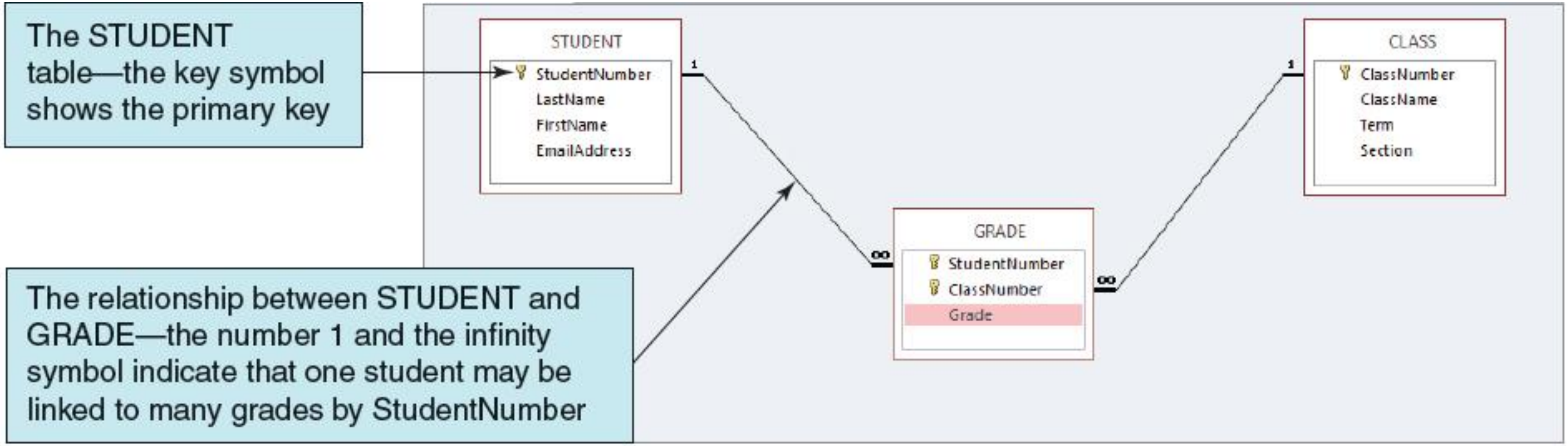
Table Relationships

A **foreign key** (FK) is a link (red arrow) from one record to another record. The FK matches up with the PK of the other record (usually in a different table).



Another Perspective ...

The Relationship graph below can be drawn **before we have data**.



Notes on Naming Conventions ...

- **Table names** are written with all capital letters and underscores between words:
  - STUDENT, CLASS, GRADE, COURSE_INFO
- **Column names** are written in "CamelCase," with no spaces and a an initial capital letter each word (including the first):
  - Term, Section, ClassNumber, StudentName

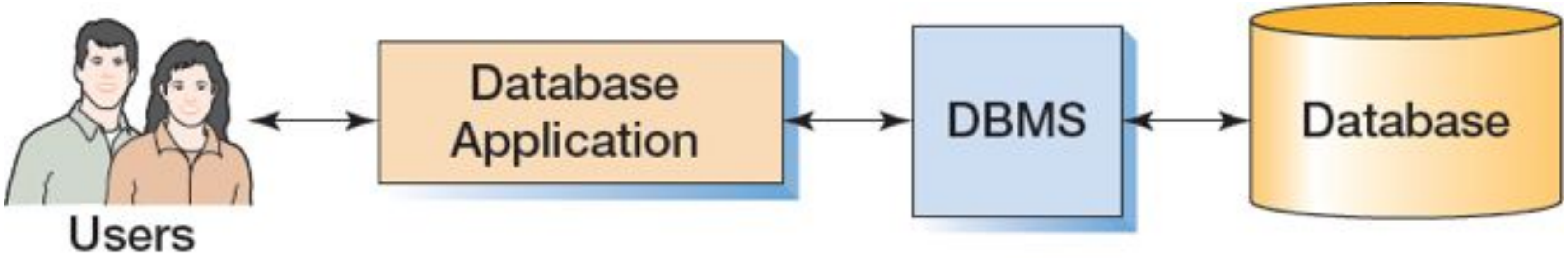
Database Application Examples

Application	Example Users	Number of Users	Typical Size	Remarks
Sales contact manager	Salesperson	1	2,000 rows	Products such as GoldMine and Act! are database centric.
Patient appointment (doctor, dentist)	Medical office	15 to 50	100,000 rows	Vertical market software vendors incorporate databases into their software products.
Customer relationship management (CRM)	Sales, marketing, or customer service departments	500	10 million rows	Major vendors such as Microsoft and Oracle PeopleSoft Enterprise build applications around the database.
Enterprise resource planning (ERP)	An entire organization	5,000	10 million+ rows	SAP uses a database as a central repository for ERP data.
E-commerce site	Internet users	Possibly millions	1 billion+ rows	Drugstore.com has a database that grows at the rate of 20 million rows per day!
Digital dashboard	Senior managers	500	100,000 rows	Extractions, summaries, and consolidations of operational databases.
Data mining	Business analysts	25	100,000 to millions+	Data are extracted, reformatted, cleaned, and filtered for use by statistical data mining tools.

Database Systems

Databases in Context

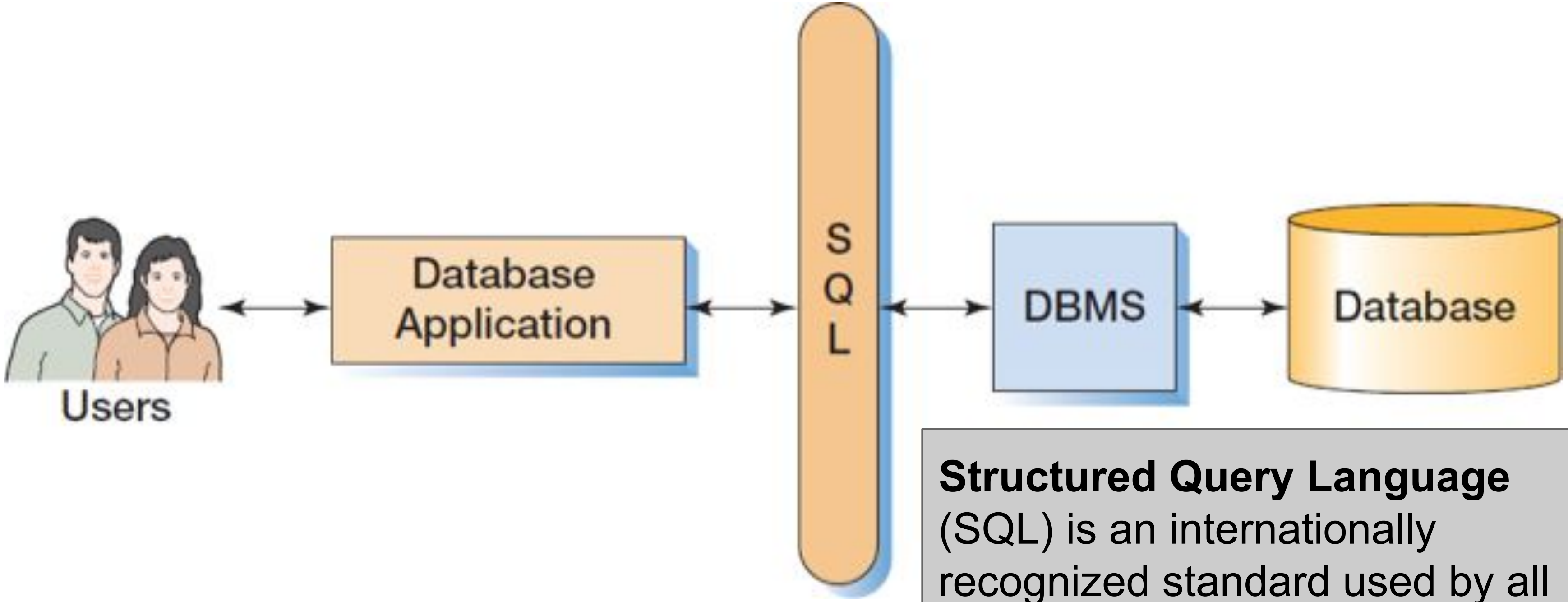
DB System Components ("layers")



Applications are the computer programs that users work with.

The **Database Management System** (DBMS) *creates*, *processes*, and *administers* databases.

SQL as an Interface to the DBMS



Structured Query Language (SQL) is an internationally recognized standard used by all commercial DBMSs.

Application layer vs DBMS layer

Basic Functions of Application Programs
Create and process forms
Process user queries
Create and process reports
Execute application logic
Control the application itself

Visible to the end user as *system use cases* with results shown on the screen.

Functions of a DBMS
Create database
Create tables
Create supporting structures (e.g., indexes)
Modify (insert, update, or delete) database data
Read database data
Maintain database structures
Enforce rules
Control concurrency
Perform backup and recovery

Invisible to the end user but required to carry out the system use cases.

The Database (again)

- A relational **database** is a self-describing collection of integrated tables.
- The tables are called **integrated** because they store data about the relationships between the rows of data.
- A database is called **self-describing** because it stores a description of itself.
- The self-describing data is called **metadata**, which is data about data.

Metadata

Metadata is stored in tables, just like any other data, except it is about the tables and columns.

The **USER_TABLES** table has metadata about tables.

The **USER_COLUMNS** table has metadata about columns.

TableName is used as a PK/FK pair to relate columns to tables.

Can you guess how this works?

USER_TABLES Table

TableName	NumberColumns	PrimaryKey
STUDENT	4	StudentNumber
CLASS	4	ClassNumber
GRADE	3	(StudentNumber, ClassNumber)

USER_COLUMNS Table

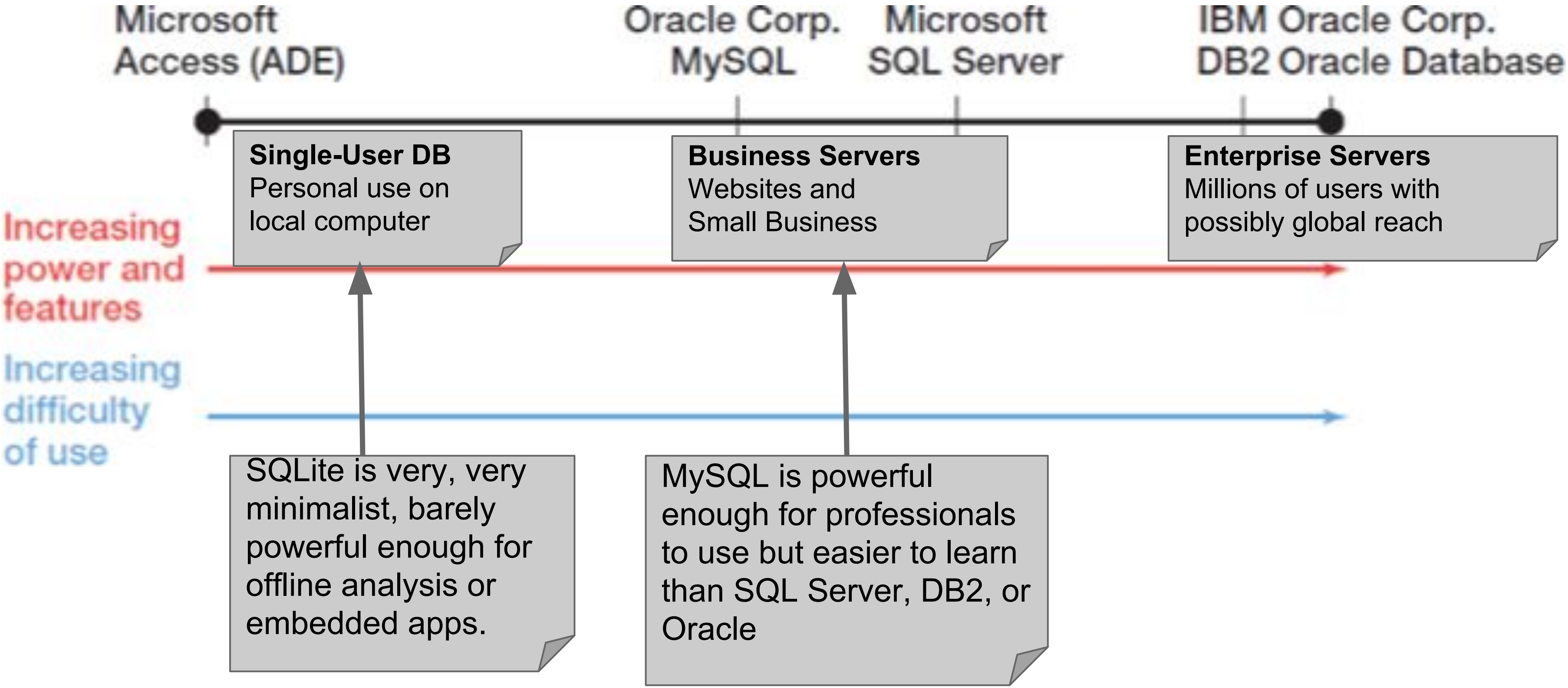
ColumnName	TableName	DataType	Length (bytes)
StudentNumber	STUDENT	Integer	4
LastName	STUDENT	Text	25
FirstName	STUDENT	Text	25
EmailAddress	STUDENT	Text	100
ClassNumber	CLASS	Integer	4
Name	CLASS	Text	25
Term	CLASS	Text	12
Section	CLASS	Integer	4
StudentNumber	GRADE	Integer	4
ClassNumber	GRADE	Integer	4
Grade	GRADE	Decimal	(2, 1)

Commonly-used DBMS Products

- Microsoft Access
- Microsoft SQL Server
- Oracle Corporation Oracle Database
- IBM DB2
- MySQL Server
- SQLite

We will be using **MySQL Server** and **SQLite** in this class, but will try to avoid anything nonstandard that wouldn't also apply to the others.

Power vs Ease of Use



Some Light History

Era	Years	Important Products	Remarks
Predatabase	Before 1970	File managers	All data were stored in separate files. Data integration was very difficult. File storage space was expensive and limited.
Early database	1970–1980	ADABAS, System2000, Total, IDMS, IMS	First products to provide related tables. CODASYL DBTG and hierarchical data models (DL/I) were prevalent.
Emergence of relational model	1978–1985	DB2, Oracle	Early relational DBMS products had substantial inertia to overcome. In time, the advantages weighed out.
Microcomputer DBMS products	1982–1992+	dBase-II, R:base, Paradox, Access	Amazing! A database on a micro. All micro DBMS products were eliminated by Microsoft Access in the early 1990s.
Object-oriented DBMS	1985–2000	Oracle ODBMS and others	Never caught on. Required relational database to be converted. Too much work for perceived benefit.

Era	Years	Important Products	Remarks
Web databases	1995–present	IIS, Apache, PHP, ASP.NET, and Java	Stateless characteristic of HTTP was a problem at first. Early applications were simple one-stage transactions. Later, more complex logic developed.
Open source DBMS products	1995–present	MySQL, PostgresQL, and other products	Open source DBMS products provide much of the functionality and features of commercial DBMS products at reduced cost.
XML and Web services	1998–present	XML, SOAP, WSDL, UDDI, and other standards	XML provides tremendous benefits to Web-based database applications. Very important today. May replace relational databases during your career. See Chapter 11 and Appendix K.
Big Data and the NoSQL movement	2009–present	Hadoop, Cassandra, Hbase, CouchDB, MongoDB, and other products	Web applications such as Facebook and Twitter use Big Data technologies, often using Hadoop and related products. The NoSQL movement is really a NoRelationalDB movement that replaces relational databases with non-relational data structures. See Chapter 12 and Appendix K.

Implications of Database Design

Why Data Analysts should know about design

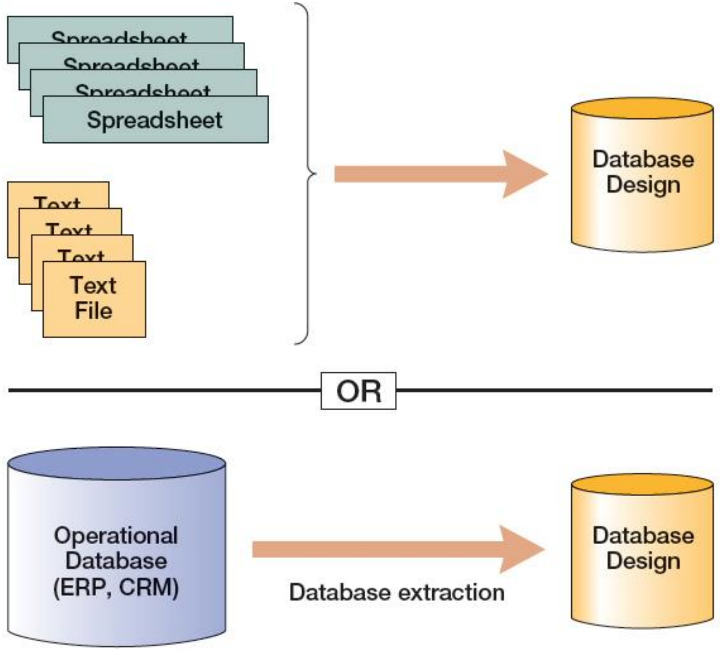
Importance of Design

Databases must fit the expected users, business operations, facilities, and needs of the organization.

- Good design provides
  - Shared data management (across whole org)
  - Access to accurate, timely, and relevant information
- Bad design leads to
  - Difficult-to-trace errors
  - Degraded ability to make and execute decisions

Effect of Data Sources

How a database is designed depends somewhat on the **provenance**, **timeliness**, **integrity**, and **organization** of the source data.



Normalization: One Table vs Two Tables

One-table design has plenty of redundancy (and potential typos) but is darn convenient.

Two-table design has higher integrity (less redundancy) but requires logic to connect the tables.

EmpNum	EmpName	DeptNum	DeptName
100	Jones	10	Accounting
150	Lau	20	Marketing
200	McCauley	10	Accounting
300	Griffin	10	Accounting

(a) One-Table Design

Denormalized Design
Use this one-table design only if there is a single data source that ensures data integrity.

OR?

DeptNum	DeptName
10	Accounting
20	Marketing

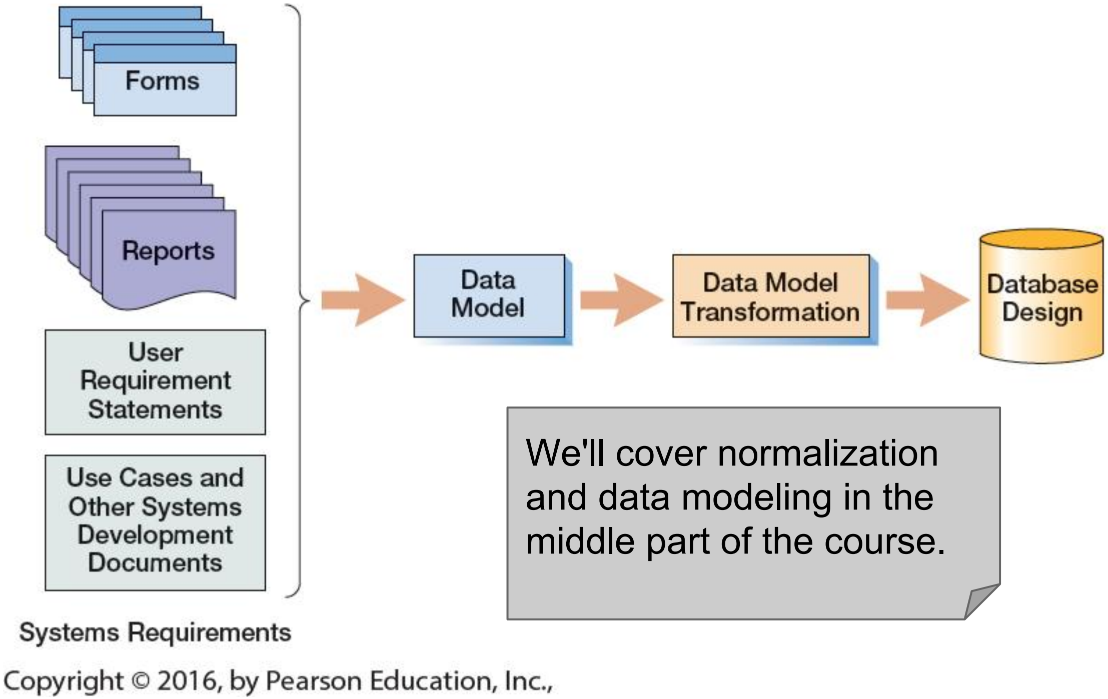
EmpNum	EmpName	DeptNum
100	Jones	10
150	Lau	20
200	McCauley	10
300	Griffin	10

(b) Two-Table Design

Normalized Design
For all other cases, break the data out into multiple tables.

The Need for Data Modeling

Database design has to consider all the possible uses and sources of data, not just those the business analyst cares about.

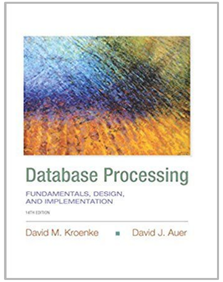


We'll cover normalization and data modeling in the middle part of the course.

Homework

- Read K/A 1 and K/A 2 (skipping anything that is vendor-specific).
- Complete any assigned DataCamp exercises
  - Take note of due dates and plan accordingly
- Study for Quiz 1, which covers K/A 1 plus Deals DB (part 1)

Fairfield University DOLAN School of Business



Databases for Analytics

Kroenke / Auer
Chapter 1