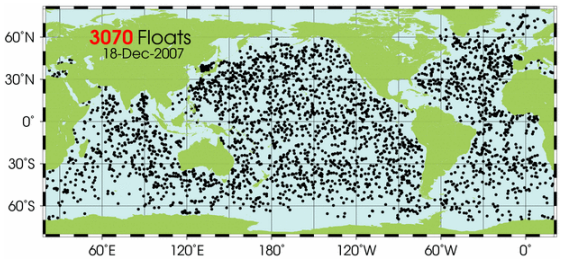
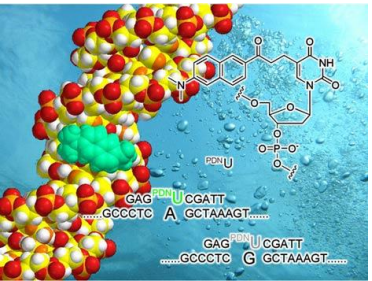
Data Management for Data Science
DATA 514

Lecture 1: Introduction



Class Goals

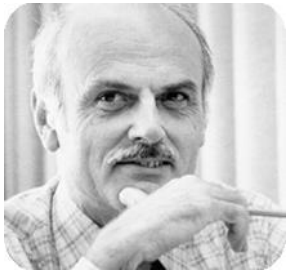
- The world is drowning in data!
- Need computer & data scientists to manage this data
  - Help domain scientists achieve new discoveries
  - Help companies provide better services (e.g., Facebook)
  - Help governments (and universities!) become more efficient
- Welcome to DATA 514: an Introduction to Data Management
  - Existing tools PLUS data management principles
  - Not just a class on SQL!



Turing Awards in Data Management

Charles Bachman, 1973

IDS and CODASYL



Ted Codd, 1981

Relational model

You could be next!!

Jim Gray, 1998

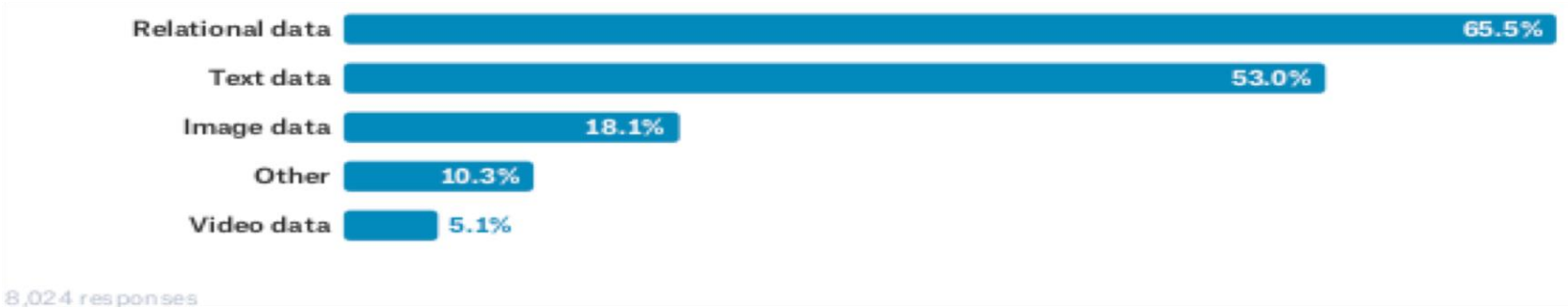
Transaction processing

Michael Stonebraker, 2014

INGRES and Postgres

Why Data Management for Data Scientists?

Most Enterprise AI Models Are Based on Relational Data*

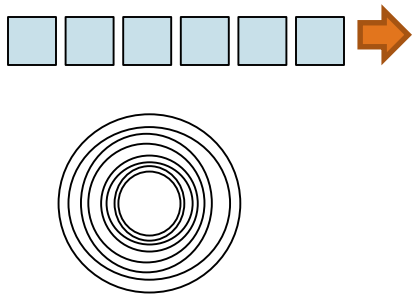


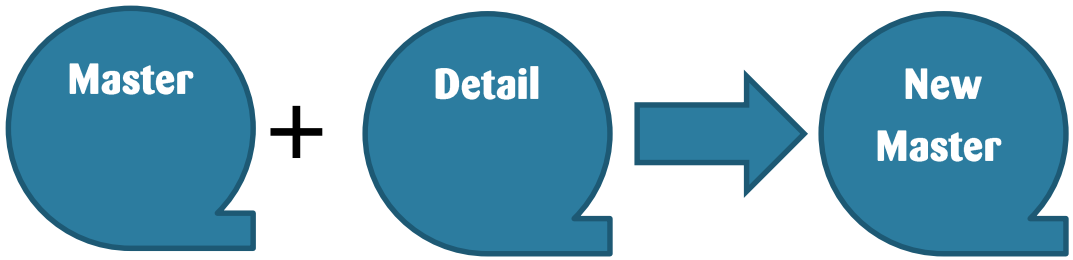
- Retail: > 86% relational
- Insurance: > 83% relational
- Marketing: > 81 % relational
- Financial: > 77% relational

* based on 2017 Kaggle survey of 16,000 ML practitioners
* some of us helped bring the relational model to databases

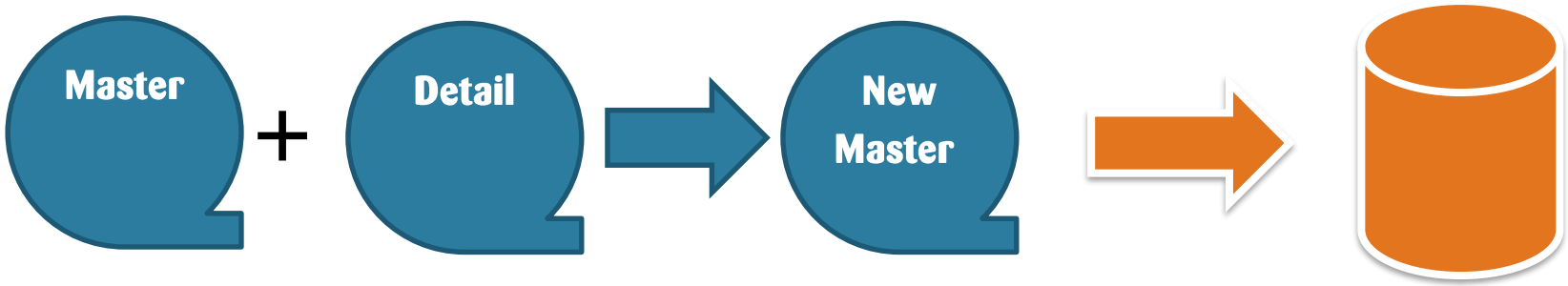
Data Management evolves!

- Begins with sequential media & access
- Direct access disks
  - CKD format
  - Access methods
    - Index by key
    - RRN, hashing
- Master + Detail files





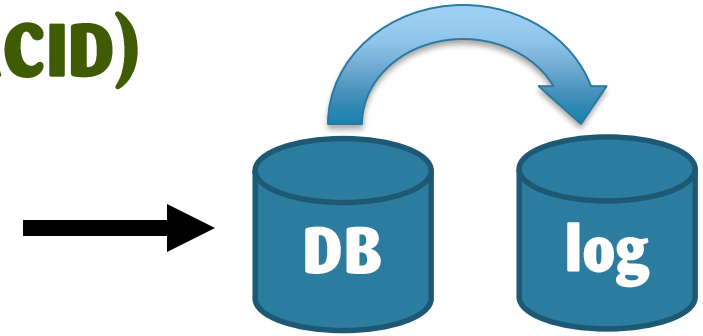
Data Management evolves!



- Master + Detail files on disk with keyed access
- became the 1st Hierarchical databases!
- CODASYL standard:
  - Keyed access
  - Hashing
  - Inverted File Systems
  - relationships

Data Management evolves!

- Online Transaction Processing (OLTP)
  - Transaction semantics (ACID)
    - Atomic
    - Consistent
    - Isolation
    - Durability
  - Lots of performance issues!
    - Concurrency, serialization & locking
  - Distributed transactions
    - 2-Phase Commit

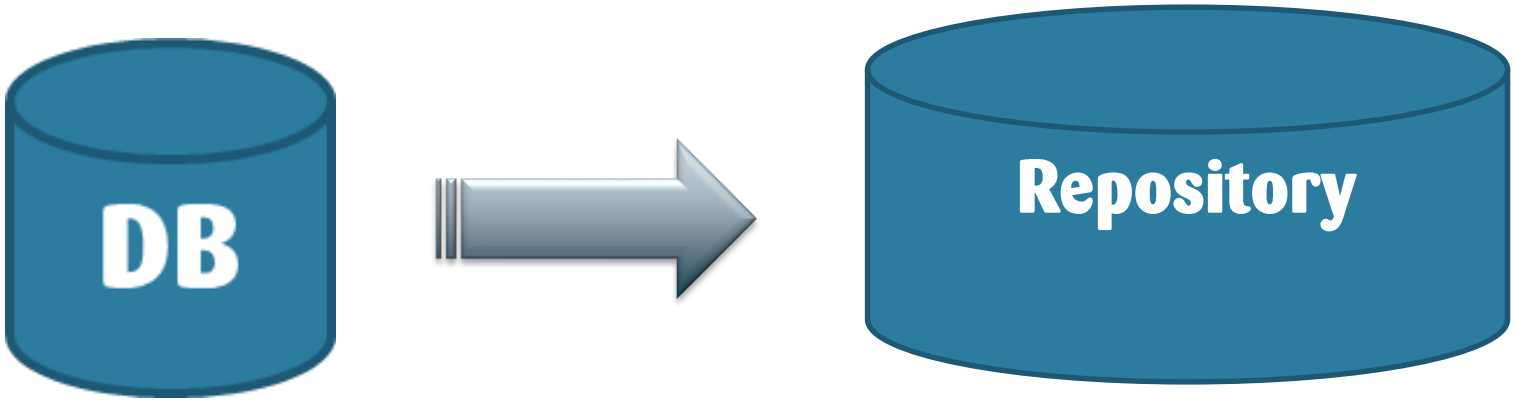


Data Management evolves!

- Relational Model
  - Based on Set theory & mathematical logic!
  - Simplified view of Data Management
    - Tables ⇨ Rows * (Strongly Typed) Columns
    - Query language based on Relational Algebra
  - Relies on technical advances:
    - b-tree Indexes
    - In-memory data buffering

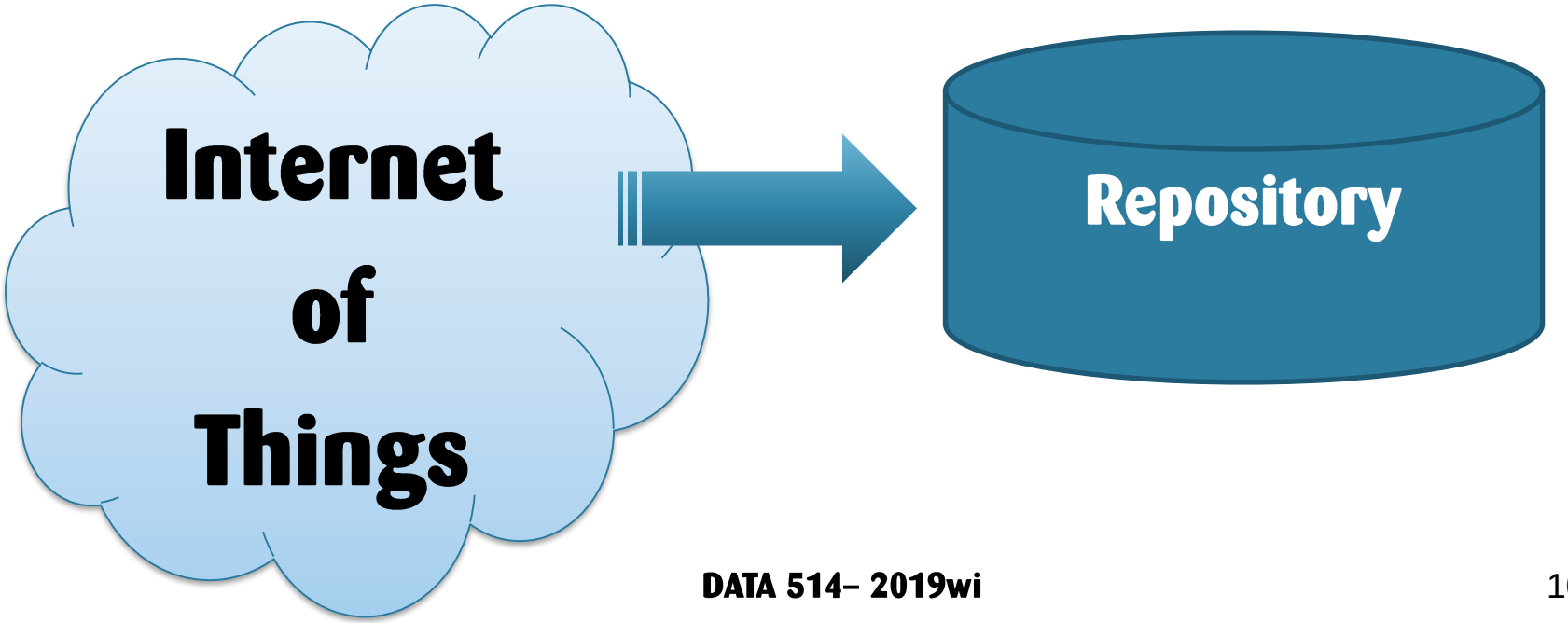
Data Management evolves!

- Data Warehousing
  - Bulk extract of OLTP DB to a Read-Only version!



Data Management evolves!

- Object Databases
  - Key-Value Pairs
  - JSON



Instructors

- Mark Friedman
  - mbfried@uw.edu
  - CSE 264
- Shrainik Jain

About me

- Professional software developer: 40+ years
  - Master's in CS ~1980
  - specialized in performance tool development, beginning around 1984
    - Landmark's The Monitor for MVS™ (1989)
    - Performance SeNTry, aka NTSMF (1997)
    - Architect, Microsoft Developer Division, 2006-2010

About me

- Industry analyst and technology entrepreneur
- Author and Instructor:
  - author of two books on Windows performance (2002, 2005)
  - blog
  - numerous technical articles published in journals and magazines
  - professional seminars, mainly on performance topics
  - CSE 590: Performance Engineering: UW 2018 au

So what I am doing teaching a class in Data Management?

- Computer Performance analysis is one the original Big Data applications
  - Data centers are awash in data!
  - e.g., Resource accounting for Cloud computing:
    - What processes are running; what resources are they using?
    - What are the Request rates and Response times for web pages?

So what I am doing teaching a class in Data Management?

- Computer Performance analysis is one the original Big Data applications
- I have extensive professional experience using repositories where this data is stored
  - as a user, developer, and designer
  - aka, Performance Databases (PDBs)

So what I am doing teaching a class in Data Management?

- Performance Databases (PDBs)
  - e.g., Statistical analysis
    - linear & non-linear regression models
    - k-means cluster analysis for workload characterization/scheduling
    - statistical quality control & anomaly detection

About you

- Class survey: please e-mail me!
  - What is your day job?
  - How many classes in the PMP have you taken?
  - What is your professional & educational background?
  - What programming languages are you familiar with?
    - C++, C#, Java, JavaScript, Python, Ruby, R, etc.
  - What platform do the applications you work on target?
    - LAMP, MacOS, iPhone, Android, ASP.NET, etc.?
  - What DBMS technology & tools are you familiar with?
    - MySQL, MS SQL Server, Oracle, MongoDB, PostgreSQL, etc.

Course Format

- Lectures: Tuesdays, 5-7:50 pm
- Lab Sections: Tuesdays, 8-8:50 pm
  - Exercises, tutorials, questions
  - **bring your laptop!**

- 6 homework assignments
- 7 web quizzes

- Midterm and final

- Canvas Discussion Board: Post and answer questions

Grading

- Homeworks 30%
- Web quizzes 20%
- Midterm 20%
- Final 30%

Warning: This is all subject to change

Communications

- **Web page:** https://courses.cs.washington.edu/courses/csed514/19wi/
  - **Class materials are here**

- **Canvas**
  - **Announcements**
  - **Discussion Board**
  - **Homework submission**

Textbook

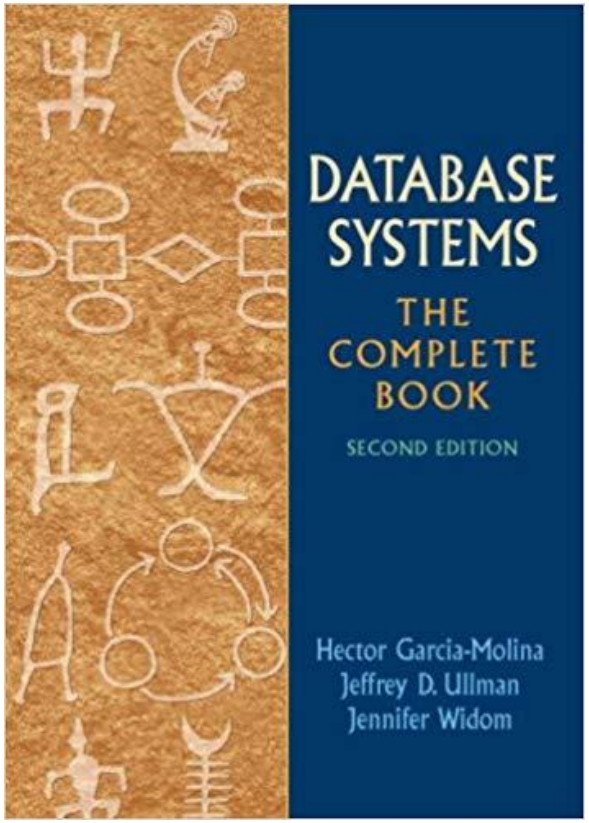
Textbook:

- ***Database Systems: The Complete Book,***
  - **authors: Hector Garcia-Molina, Jeffrey Ullman, Jennifer Widom**

  2nd edition

- **Read chapters 1,2 & 6 for next time.**

REQUIRED READING !



Six Homework Assignments

- H1: Sqlite
- H2: Basic SQL with SQLite
- H3: Advanced SQL with SQL Server
- H4: Conceptual Design
- H5: JSon
- H6: SQL in Java (JDBC)

Check calendar for due dates. Submit via using Canvas

About the Assignments

- You will learn/practice the course material
- You will also learn lots of new technology
  - Time spent learning it is useful!
  - Put everything on your resume!!!

  - SQL, SQLite, SQL Server, SQL Azure JDBC, JSon,…

Deadlines and Late Days

- You have up to 4 late days
  - No more than 2 on any one assignment
  - Use in 24-hour chunks

- Late days = safety net, not convenience
  - You should not plan on using them
  - If you use all 4 you are doing it wrong

Six Web Quizzes

- http://newgradiance.com/
- Create account
- Please use the same ID as your UW ID
- Provide token (on the whiteboard)
- Short tests, you may take them many times, best score counts
- No late days – closes at 11:00 pm deadline

Exams

- Midterm (Feb 5) and Final (March 12)
- You may bring letter-size piece of paper with notes
  - May write on both sides
  - Midterm: 1 sheet, Final: 2 sheets
- Final is closed book. No computers, phones, watches,...
- Location: in class

Academic Integrity

- Anything you submit for credit is expected to be your own work
  - OK to exchange ideas, not detailed solutions
  - We all know difference between collaboration and cheating
- I trust you implicitly, but will come down hard on any violations of that trust

Lecture Notes

- Will be available before class online
- Feel free to bring them to class to take notes
- Refresh often, since I revise them constantly, often at the last minute

Using Devices in Class

In the lectures:

- **Opened laptops may disturb neighbors**
- **Please sit in the back if you take notes on laptop; pads / surfaces are OK**
- **Please don't use your devices to check your email, youtube, fb, etc.**

In the lab sections:

- **Always bring your laptop (starting next week)**

Now onto the real stuff…

Outline of Today's Lecture

- Overview of database management systems
- Highlight course content:
  - Data Models
  - SQL (Relational DBMS): Relational Algebra
  - Logical and Physical Database design (E:R)
  - Query Plan execution
  - Transactions
  - noSQL (Object-oriented) approaches

Database

What is a database ?

Give examples of databases

Demo

Database

What is a database ?

- A collection of files storing related data

Give examples of databases

Database

What is a database ?

- A collection of files storing related data

Give examples of databases

- Accounts database; payroll database; UW's students database; Amazon's products database; airline reservation database

Database Management System

What is a DBMS ?

Give examples of DBMSs

Database Management System

What is a DBMS ?

- A big program (written by someone else) that allows us to manage a large database efficiently and allows it to persist over long periods of time

Examples of popular, commercial DBMSes

- Oracle, IBM DB2, Microsoft SQL Server, Vertica, Teradata
- Open source: MySQL (Sun/Oracle), PostgreSQL, CouchDB
- Open source library: SQLite

The class focuses on Relational DBMSes, except towards the end of the term

An Example: Online Bookseller

- What data do we need?
  -
  -
  -
  -
- What capabilities on the data do we need?
  -
  -
  -

An Example: Online Bookseller

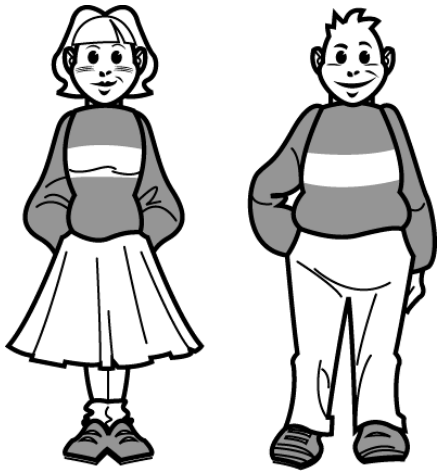
- What data do we need?
  - Data about books, customers, pending orders, order histories, trends, preferences, etc.
  - Data about sessions (clicks, pages, searches)
  - Note: data must be persistent! Outlive application
  - Also note that data is large… won't fit all in memory
- What data management capabilities does the Bookseller need?
  -
  -
  -

An Example: Online Bookseller

- What data do we need?
  - Data about books, customers, pending orders, order histories, trends, preferences, etc.
  - Data about sessions (clicks, pages, searches)
  - Note: data must be persistent! Outlive application
  - Also note that data is large… won't fit all in memory
- What capabilities on the data do we need?
  - Insert/remove books, find books by author/title/etc., analyze past order history, recommend books, …
  - Data must be accessed efficiently, by many users
  - Data must be safe from failures and malicious users

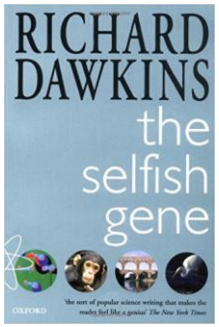
Challenges for a DBMS

Alice and Bob receive a $200 gift certificate as wedding gift

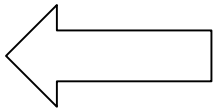


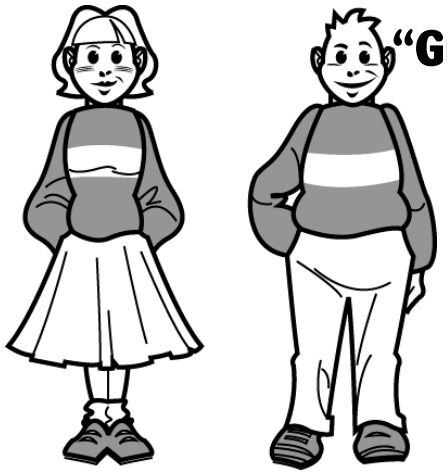
Challenges for a DBMS

Alice and Bob receive a $200 **gift** certificate as wedding gift



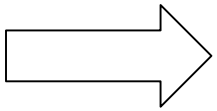
Alice @ her office orders "The Selfish Gene"

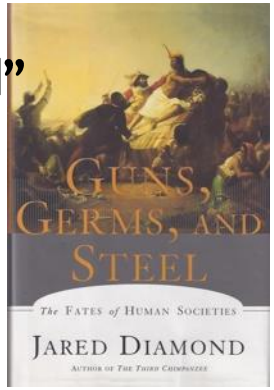




Alice Bob

Bob @ home orders "Guns, germs, and steel"



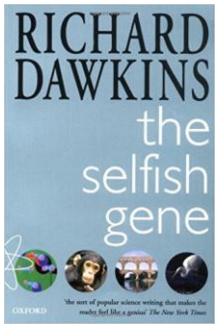


$80

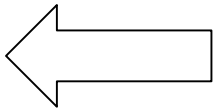
$100

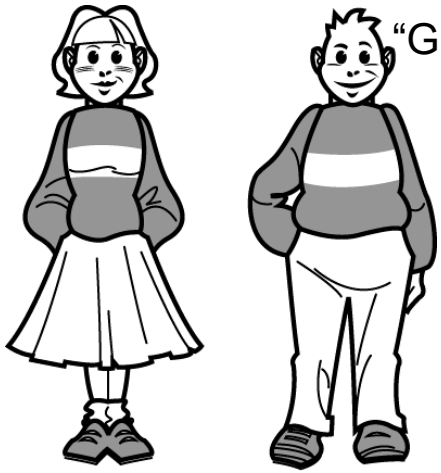
Challenges for a DBMS

Alice and Bob receive a $200 gift certificate as wedding gift



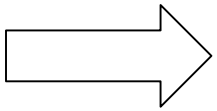
Alice @ her office orders
"The Selfish Gene"

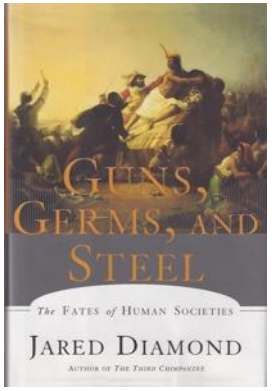




Alice Bob

Bob @ home orders
“Guns, germs, and steel”





$80

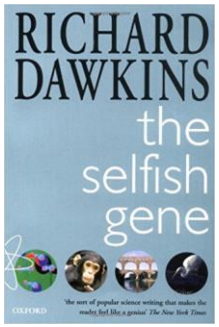
$100

Questions:

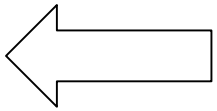
- What is the ending credit?
- What if second book costs $130?
- What if system crashes?

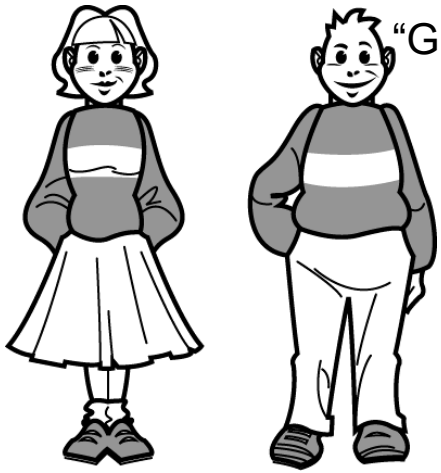
Challenges for a DBMS

Alice and Bob receive a $200 gift certificate as wedding gift



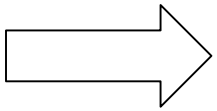
Alice @ her office orders
"The Selfish Gene"

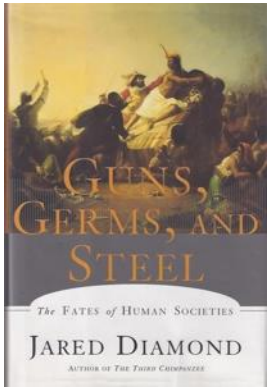




Alice Bob

Bob @ home orders
“Guns, germs, and steel”





$80

$100

Questions:

- What is the ending credit?
- What if second book costs $130?
- What if system crashes?

Lesson:

a DBMS needs to handle data integrity scenarios

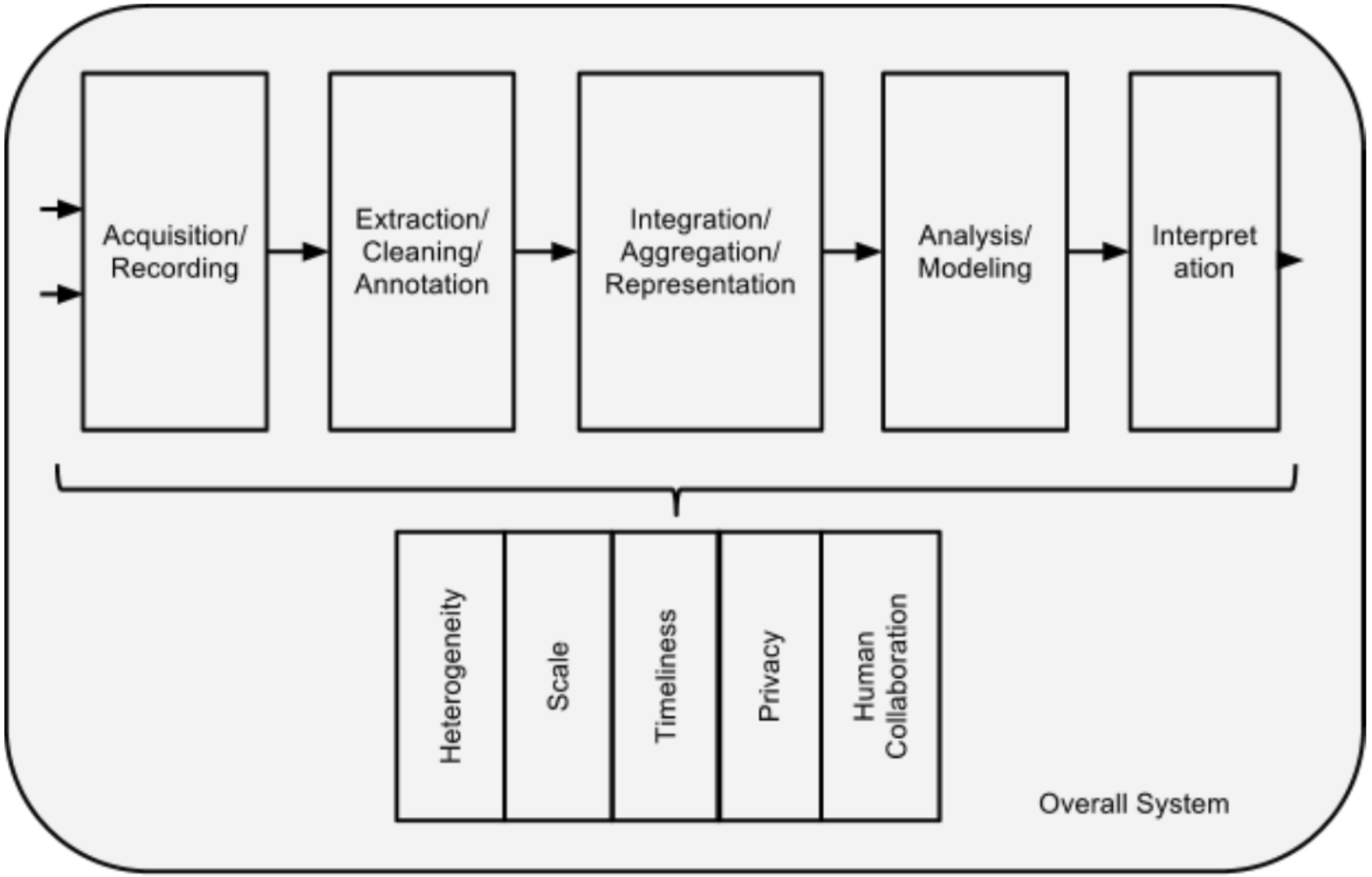
What a DBMS Does

- Describe real-world entities
- Store large datasets persistently
- Query & update efficiently
- Change structure (e.g., add attributes)
- Handle concurrent updates
- Crash recovery
- Provide security and data integrity

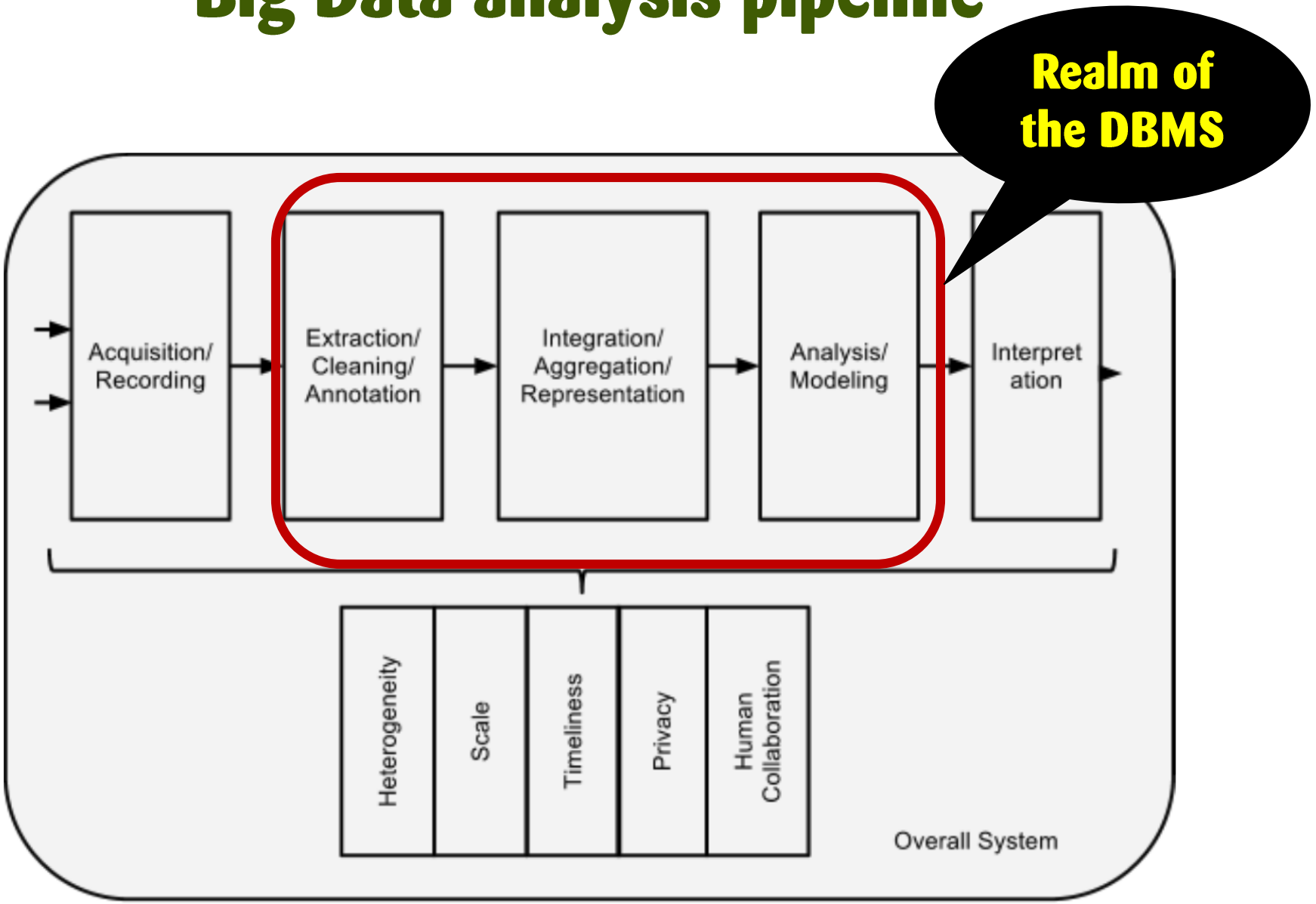
Key DM Roles

- DB application developer: writes programs that query and modify data
- DB designer: establishes schema
- DB administrator: loads data, tunes system, keeps whole thing running
- Data analyst: data mining, data integration

Big Data analysis pipeline



Big Data analysis pipeline



Some DBMS benefits

- Data is stored independently of the applications that access it
- Security & Auditability
- High Availability (logging and recovery)
- Scalability
- Concurrency
- Rich development tools

Specific RDBMS benefits

- Data is stored in **Tables**: organized as uniform data rows and columns (fields)
- Access and control using the **SQL** language
  - SQL = Structured Query language
  - ANSI standard(s) for syntax
  - based on the Relational Algebra (set theory)
    - Table = Relation
    - subsets, Union, Intersection, etc.
- Any Table can be indexed for fast access

Specific RDBMS benefits

Access and control using SQL language

SELECT col1, col2, …
FROM table1, table2, …
WHERE cond1, cond2, …
GROUP BY col*m*, col*n*, …
ORDER BY sortcol1, sortcol2, …

Specific RDBMS benefits

Access and control using SQL language

SELECT col1, col2, …
FROM table1, table2, …
WHERE cond1, cond2, …
GROUP BY col*m*, col*n*, …
ORDER BY sortcol1, sortcol2, …

- SQL Queries specify what data you want to retrieve, not how to retrieve it
  - Query Optimizer function figures out how to retrieve the data
  - Data independence!

(Conceptual) Data Model

- Entity : Relation (E:R) approach
  - Entities (sets)
    - e.g., customers, products, orders, suppliers, etc.
    - Entities have attributes (properties)
    - custID, name, ship address, e-mail, billing, …
    - prodID, product-name, supplier, price, …
    - orderID, custID, prodID, …

Entity : Relation

- **Entity sets**
  - **e.g., *customers, products, orders, suppliers*, etc.**
- **Relations between Entities**
  - **e.g., *IsA, HasA, Contains, Owns*, …**

Entity	Relationship	Entity
Customer	<orders>	Titles
Publisher	<supplies>	Titles
Customer Order	<contains>	Titles
Warehouse	<ships>	Customer Order

E:R data model

- Conceptual model of the data to be represented in the Database
  - Entities
    - **customer**: custID, name, shipping address, e-mail, billing address, credit card, …
  - Relations between Entities
    - **orders**: custID, orderID, productid
  - Note: Relations have **multiplicity**
    - 1:1, 0..1:1, 1..n:1, m:n, (+ ***constraints***)

E:R conceptual data model

- **multiplicity** of Relations
  - e.g.,
    - 1:1 required relationship (person, SSid)
    - 0:n optional relationship (driver, auto)
    - m:n many-to-many (authors, books)

  - chicken : egg
  - rooster : egg
  - chicken : wing

E:R conceptual data model

- Conceptual Model ⇨ database schema
  - Entities and Relationships are both represented in the database schema
  - Relational DBMS: entities and relationships are stored as Tables (Relations)
    - Entities (id, col1, col2, col3, …)
    - Relationship tables ($entity_1$id, $entity_2$id, …)
  - Alternative representations:
    - JSON objects (tree-structured)
    - Key-Value pairs (NoSQL)
    - Graph nodes and edges

Logical data model

- Entity attributes in Tables (columns) are strongly typed
  - Data type determines physical storage
    - logical/bit fields
    - char
    - string (fixed length character arrays)
    - varchar
    - decimal
    - floating point
    - datetime
    - money
    - etc.

Physical data model

- Indexes
  - Physically implemented as balanced Trees
    - tree Search (based on index fan-out ratio)
    - insertions operations into b-Trees
  - Unique
    - Primary Key (clustered Index determines the physical sort order)
    - Auto-generated (e.g., custID)
    - Can use multiple columns in combination
  - Non-unique (non-clustered)
    - tree Search (based on index fan-out ratio)

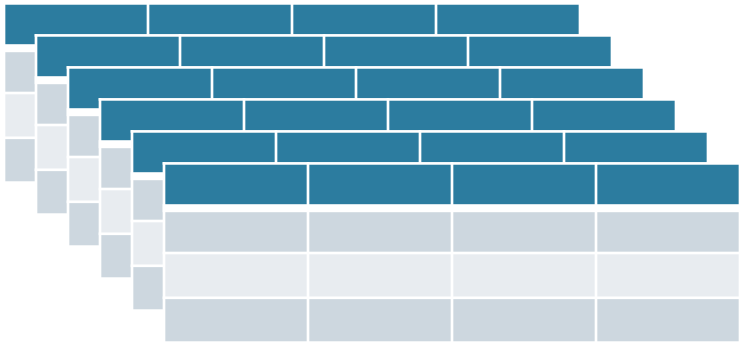
Physical data model

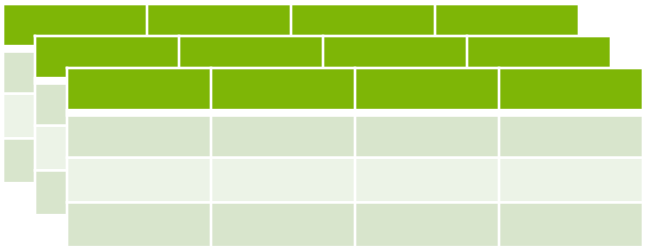
- Foreign Key
  - column in $Relation_1$ that points to a designated Row (or Rows) in $Relation_2$
  - Facilitates JOIN operations
  - example:
    - Country (countryCode, countryName)
    - Order (… , countryCode, …)

- Rationale: save physical space in DB; represent data common to multiple Relations without redundancy

Demo

- (Semantics) Schema ⇨ set of Tables
  - (Entities ⇨ Attributes)
  - Physical DB design:
    - (unique) Keys, Indexes, Foreign Keys
- Relationships between these Entities
  - customers : orders : <customer-orders: custid, ordered>
- Data Access Language (SQL) based on relational algebra (Set theory + Logic)

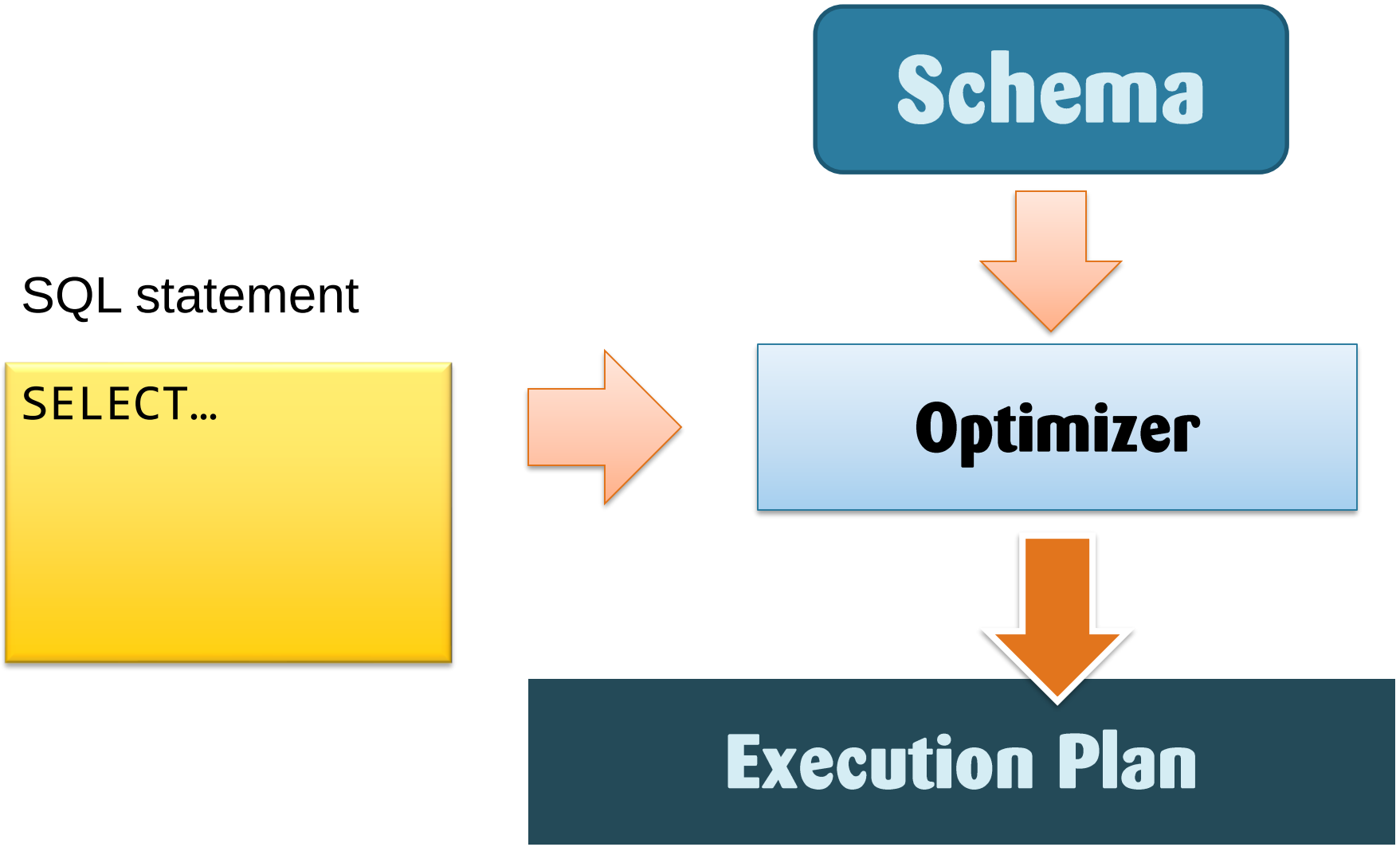




SQL

- **SELECT** col1, col2, …
- **FROM** rel1, rel2, …
- **WHERE** cond1 And cond2 Or cond3 …
- **GROUP BY** aggregate1, aggregate2, …
- **HAVING** cond1 And cond2 Or cond3 …
- **ORDER BY** sortkey1, sortkey2, …

SQL execution Plan



SQL example

```
SELECT wsRaceHeader.RaceTypeId, RaceTypeDescription,
   Count(RaceId) As NumRaces
FROM wsRaceHeader, wsRaceTypes
Where wsRaceHeader.RaceTypeId = wsRaceTypes.RaceTypeId
GROUP BY wsRaceHeader.RaceTypeId, RaceTypeDescription
Order by NumRaces DESC
```

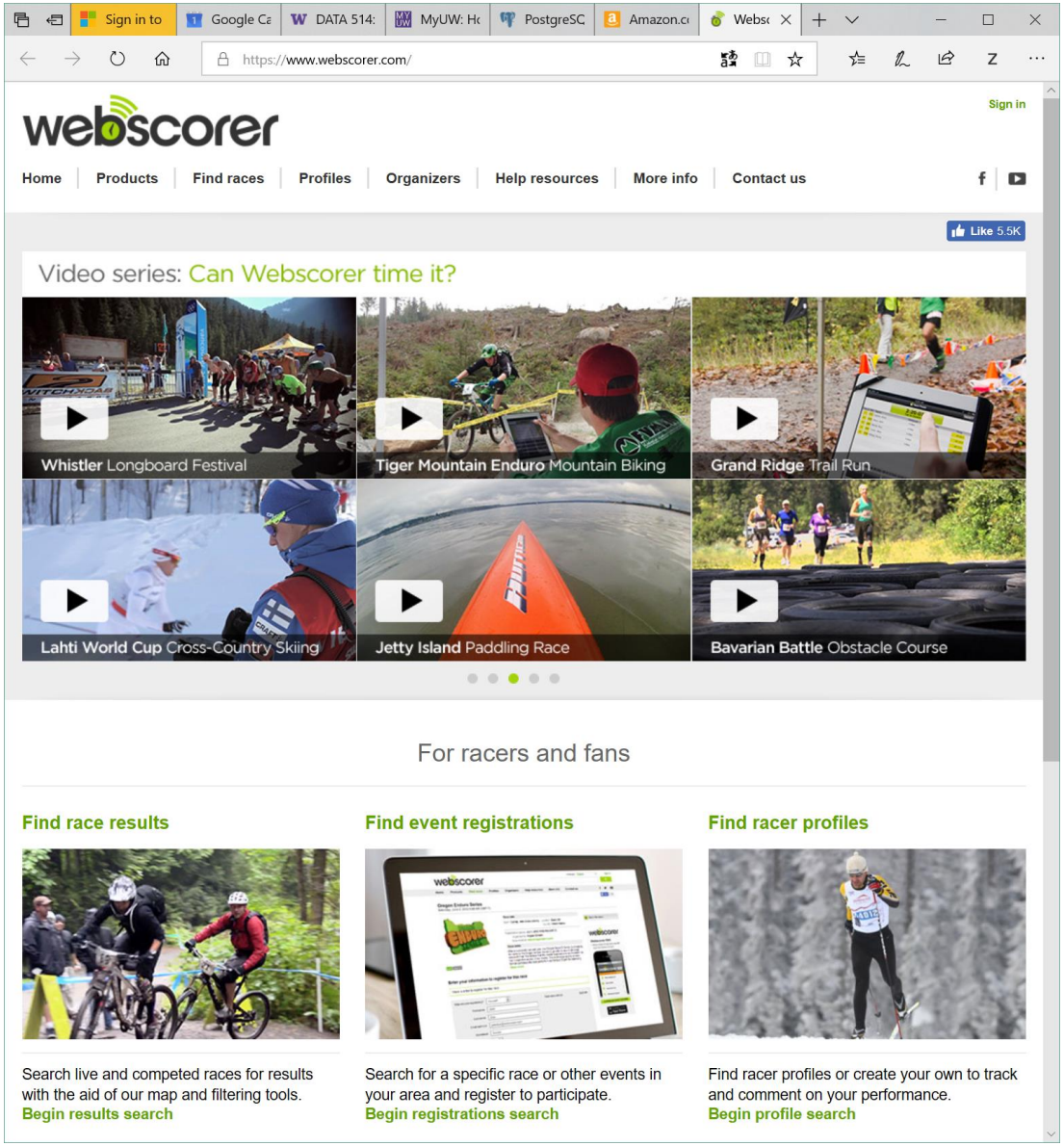
Using SQL in practice

- Data Administration
  - Data independence (in principle)
  - Security considerations
    - e.g., Row-level security
  - Views ⇦
    - projections/selections/aggregations of actual Tables
  - Stored Procedures

Demo

- navigate to www.Webscorer.com

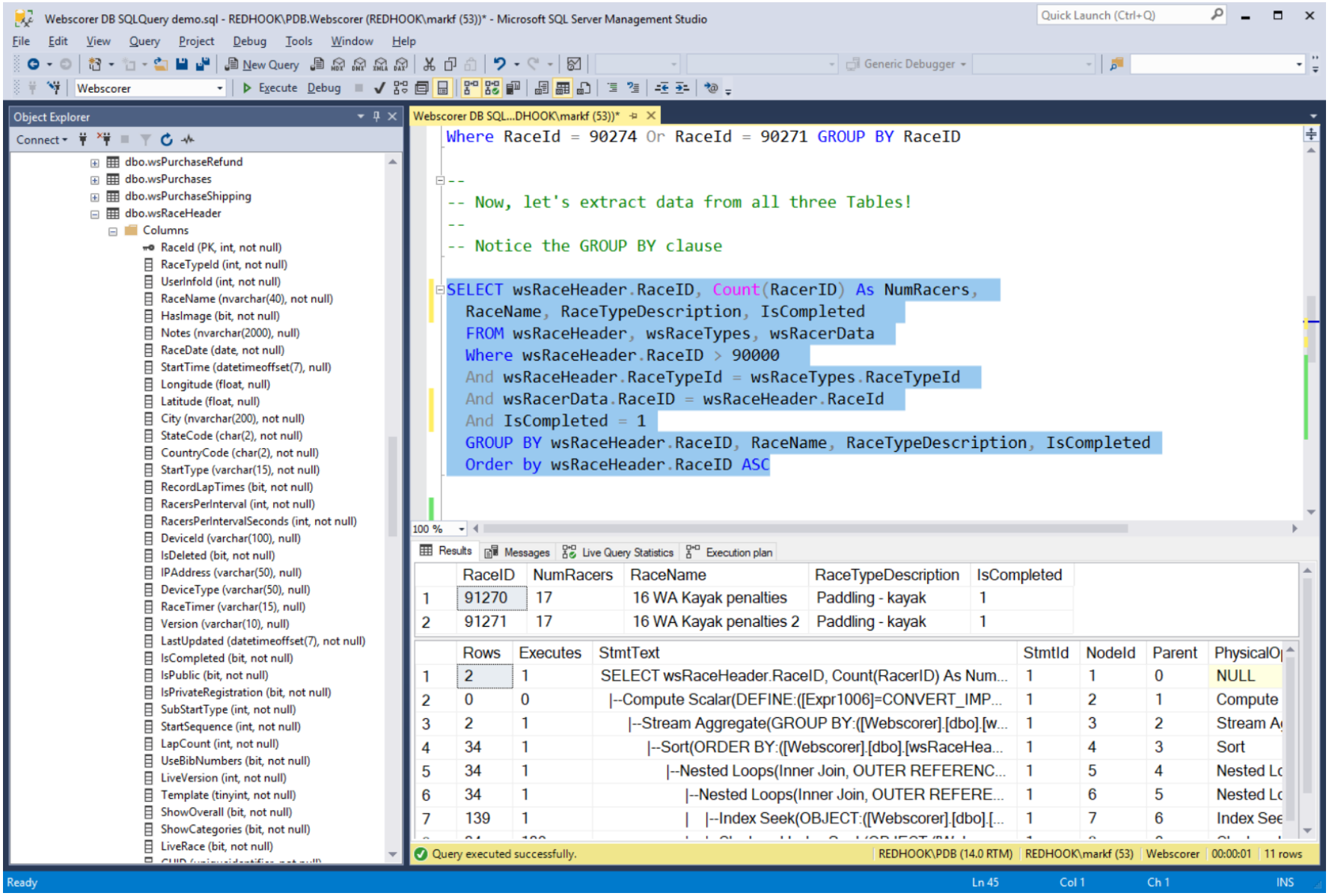
- Conceptual Model
  - Races
  - Racers
  - Race organizers
  - Race Results



Demo www.Webscorer.com

- Semantic/Conceptual (E:R) Model:
  - Races, Racers, organizers, Race Results, Race Types
- Primary Keys (clustered Index)
- Foreign Keys
- Constraints
- JOINs
- Calculated fields
- Aggregates (Group by)
- Sorting
- Execution Plan

Demo


Webscorer DB SQLQuery demo.sql - REDHOOK\PDB.Webscorer (REDHOOK\markf (53))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Debug Tools Window Help
New Query
Generic Debugger
Webscorer
Execute Debug
Object Explorer
Connect
dbo.wsPurchaseRefund
dbo.wsPurchases
dbo.wsPurchaseShipping
dbo.wsRaceHeader
Columns
RaceId (PK, int, not null)
RaceTypeId (int, not null)
UserInfoId (int, not null)
RaceName (nvarchar(40), not null)
HasImage (bit, not null)
Notes (nvarchar(2000), null)
RaceDate (date, not null)
StartTime (datetimeoffset(7), null)
Longitude (float, null)
Latitude (float, null)
City (nvarchar(200), not null)
StateCode (char(2), not null)
CountryCode (char(2), not null)
StartType (varchar(15), not null)
RecordLapTimes (bit, not null)
RacersPerInterval (int, not null)
RacersPerIntervalSeconds (int, not null)
DeviceId (varchar(100), null)
IsDeleted (bit, not null)
IPAddress (varchar(50), null)
DeviceType (varchar(50), null)
RaceTimer (varchar(15), null)
Version (varchar(10), null)
LastUpdated (datetimeoffset(7), not null)
IsCompleted (bit, not null)
IsPublic (bit, not null)
IsPrivateRegistration (bit, not null)
SubStartType (int, not null)
StartSequence (int, not null)
LapCount (int, not null)
UseBibNumbers (bit, not null)
LiveVersion (int, not null)
Template (tinyint, not null)
ShowOverall (bit, not null)
ShowCategories (bit, not null)
LiveRace (bit, not null)
Webscorer DB SQL...DHOOK\markf (53))*
Where RaceId = 90274 Or RaceId = 90271 GROUP BY RaceID
--
-- Now, let's extract data from all three Tables!
--
-- Notice the GROUP BY clause
SELECT wsRaceHeader.RaceID, Count(RacerID) As NumRacers,
RaceName, RaceTypeDescription, IsCompleted
FROM wsRaceHeader, wsRaceTypes, wsRacerData
Where wsRaceHeader.RaceID > 90000
And wsRaceHeader.RaceTypeId = wsRaceTypes.RaceTypeId
And wsRacerData.RaceID = wsRaceHeader.RaceId
And IsCompleted = 1
GROUP BY wsRaceHeader.RaceID, RaceName, RaceTypeDescription, IsCompleted
Order by wsRaceHeader.RaceID ASC
100 %
Results Messages Live Query Statistics Execution plan
RaceID NumRacers RaceName RaceTypeDescription IsCompleted
1 91270 17 16 WA Kayak penalties Paddling - kayak 1
2 91271 17 16 WA Kayak penalties 2 Paddling - kayak 1
Rows Executes StmtText StmtId NodeId Parent PhysicalO
1 2 1 SELECT wsRaceHeader.RaceID, Count(RacerID) As Num... 1 1 0 NULL
2 0 0 |--Compute Scalar(DEFINE:([Expr1006]=CONVERT_IMP... 1 2 1 Compute
3 2 1 |--Stream Aggregate(GROUP BY:([Webscorer].[dbo].[w... 1 3 2 Stream A
4 34 1 |--Sort(ORDER BY:([Webscorer].[dbo].[wsRaceHea... 1 4 3 Sort
5 34 1 |--Nested Loops(Inner Join, OUTER REFERENC... 1 5 4 Nested Lo
6 34 1 |--Nested Loops(Inner Join, OUTER REFERE... 1 6 5 Nested Lo
7 139 1 | |--Index Seek(OBJECT:([Webscorer].[dbo].[... 1 7 6 Index See
Query executed successfully.
REDHOOK\PDB (14.0 RTM) REDHOOK\markf (53) Webscorer 00:00:01 11 rows
Ready
Ln 45 Col 1 Ch 1 INS

Demo www.Webscorer.com

- Semantic/Conceptual (E:R) Model:
  - Races, Racers, organizers, Race Results, Race Types
- Primary Keys (clustered Index)
- Foreign Keys
- Constraints
- JOINs
- Calculated fields
- Aggregates (Group by)
- Sorting
- Execution Plan

General form of Grouping and Aggregation

```
SELECT    S
FROM      R1,…,Rn
WHERE     C1
GROUP BY  a1,…,ak
HAVING    C2
```

S = may contain attributes $a_1,\ldots,a_k$ and/or any aggregates but NO OTHER ATTRIBUTES

C1 = is any condition on the attributes in $R_1,\ldots,R_n$

C2 = is any condition on aggregate expressions and on attributes $a_1,\ldots,a_k$

Semantics of SQL With Group-By

```
SELECT     S
FROM       R1,…,Rn
WHERE      C1
GROUP BY   a1,…,ak
HAVING     C2
```

FWGHOS

Evaluation steps:

1. Evaluate FROM-WHERE using Nested Loop Semantics
2. Group by the attributes $a_1,\ldots,a_k$
3. Apply condition C2 to each group (may have aggregates)
4. Compute aggregates in S and return the result

Next Steps

web site: https://courses.cs.washington.edu/courses/csed514/19wi/

- Homework 1 is posted
  - Simple queries in SQL Lite
  - Due on Monday, 1/14
- Webquiz 1 is open
  - Create account at http://newgradiance.com/
  - Sign up for class online
  - Due on Monday, 1/14
- Read textbook: chapters 1,2 & 6
- First lab sections to follow: simple queries in SQL Lite