Bill Gates, the Revolution, and the Art of Gardening

Based on a true story
Outline

- What are databases about?
- A history of database systems (abridged)
- Why are databases so successful?
- Why does Bill Gates appear in the title?
Databases

- Database Management System (DBMS): an integrated suite of tools to manage (store, query, update) a large collection of data

- Databases are used everywhere
  - ATMs, Banking Systems, Web Applications, ...

- Lucrative career path
  - Skilled DB programmers and administrators are very well paid
It looks so simple...

- **Input:** a query (in SQL)
- **Output:** results

```sql
SELECT title
FROM Movies, Actors
WHERE Actors.mid = Movies.mid AND Actors.name="Pitt"
```

SQL Query → DBMS → Results
...but it is very complex!

- Transactional semantics
- Recovery
- Queries are declarative and must be optimized
- Data on disk is much larger than main memory
- Lots more!
Just how good is a DBMS?

Here are some examples:

- A DBMS can sort efficiently 700 gigabytes of data using 80MB of main memory.
- A DBMS can sustain hundreds of thousands of transactions per second.
- A DBMS can mine information from terabytes of data.
History
circa 1960

- Network Data Model (CODASYL)
- Key idea: a database is a network of records
- Tuple-at-a-time processing
1970: The revolution

Ted Codd proposes the Relational Model

Strong mathematical foundation

Set-a-time processing

Two query languages: RA and RC

<table>
<thead>
<tr>
<th>mid</th>
<th>year</th>
<th>genre</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2005</td>
<td>“action”</td>
</tr>
<tr>
<td>2</td>
<td>2004</td>
<td>“drama”</td>
</tr>
<tr>
<td>3</td>
<td>2000</td>
<td>“drama”</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>aid</th>
<th>sex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>male</td>
</tr>
<tr>
<td>2</td>
<td>female</td>
</tr>
<tr>
<td>3</td>
<td>male</td>
</tr>
</tbody>
</table>

σ_{name="Pitt"}(Actor) ⊗ Cast ⊗ Movie
CODASYL vs. Relational

**CODASYL:**
- Relational is too mathematical!
- Relational is not good for record-oriented apps!
- There is no efficient implementation for RA or RC!

**Relational:**
- CODASYL is too complex!
- Set-oriented queries are so much better!
- CODASYL does not have any math behind it!
mid 70s: the turn of the tide

- System-R and Ingres emerge as the first relational systems
- SQL and QUEL appear as user-friendly versions of RA and RC
- End result: the relational model starts looking more attractive!
Ingres

- Led by UC (Berkeley) researchers
- Research system
- Descendant: Postgres
System-R

- Developed at IBM San Jose lab
- High impact work
- Descendants: Oracle, DB2, ...
- SQL => System R version 1 => System R version 2
80s: Commercialization

- Oracle beats System-R to the market
- IBM releases many systems and settles on DB2
- Informix is started by Roger Sippl
- 1985: SQL Standard
90s–today: Domination

- Relational systems are the standard
- CODASYL survives only on legacy systems
- SQL standard is > 1000 pages (started at 20!)
- Billion dollar industry
- Major players: Oracle, IBM, Microsoft
Why are databases so good?
SQL

- Separate application code from data

```
SELECT title
FROM Movie, Actor
WHERE Movie.mid=Actor.mid AND Actor.name="Pitt"

SELECT Movie.year, COUNT(*)
FROM Movie, Actor
WHERE Movie.mid=Actor.mid AND Actor.name="Pitt"
GROUP BY Movie.year
```
Query Optimization

SQL is declarative => User describes *what* data to retrieve (but not how!)

The system determines the best plan to evaluate the query

Scalable processing algorithms
Data Independence

- **External Schema**
- **Conceptual Schema**
- **Physical Schema**

### External Schemas
- `YoungActors(name, age)`
- `PittMovies(title)`

### Conceptual Schemas
- `Movie(mid, title)`
- `Actor(name, mid, age)`

### Physical Schemas
- `Movie(mid, title) + Index on mid + Index on title`
- `Actor(name, mid, age) + Index on age`
ACID

Transactional Consistency:

- Atomicity
- Consistency
- Isolation
- Durability

Completely transparent to the application

Jim Gray won a Turing Award for his work on transactions
Database systems have strong theoretical foundations.

Prominent example: schema normalization.

Actor(name, mid) vs. Actor(name, aid)
Movie(mid, title, year) vs. Cast(aid, mid)
Movie(mid, title, year)
New Trends
Data Streams

Motivating example: run SQL query over the packet stream of a network router
  Compute the number of packets that originate from hosts A and B and target the same machine

Problem #1: data is infinite!

Problem #2: data arrives fast!

Need for new architecture:
  - New query languages
  - New optimization techniques
  - New paradigms
Data Mining

- **Goal:** extract knowledge from data
  - Example: Walmart data has revealed that customers who buy beer also buy diapers
- **Challenge #1:** scale!
  - Walmart is already at terabytes and going for petabytes
- **Challenge #2:** richer knowledge
  - Frequent sub-sequences, association rules, ...
  - What’s next?
Sensor Databases

- Goal: deploy a large number of small devices that measure “stuff” (temperature, illumination, humidity, ...)
- Use SQL to manipulate sensor data
- Challenge #1: Low power consumption
- Challenge #2: Limited processing power
Peer-to-Peer

- P2P systems have been popular for file sharing
- What about running SQL over P2P?
- Challenge #1: Volatility
- Challenge #2: Locating the data
- Challenge #3: Consolidating heterogeneity
XML
Letter to Bill Gates
“Microsoft mailing address”

Microsoft Address

Microsoft Office Assistance

USNPL - Newspaper Addresses

Sign-in Access Error

Mailing & Related Items - Find it on eBay!

www.eBay.com  eBay - The World’s Online Marketplace
"Microsoft address"

Google search results for "microsoft address".

- Worm dupes with fake Microsoft address | CNET News.com
- Microsoft kills Net address to foil worm | CNET News.com
- 1999 09 14: Intel, Compaq, Entrust, Ibm And Microsoft Address ...
- Portal: Portal Software and Microsoft Address WiFi Business ...

Tip: In most browsers you can just hit the return key instead of clicking on the search button.

Categories: Computers > Software > ... > Mac OS > Home Productivity Computers > Software > ...
Web Search Today

- Web document: bag of words
- HTML: presentation language

```html
<I>
  Microsoft
  One Microsoft Way
  Redmond, WA
</I>

```html
<I>
  Teriyaki sauce
  One egg
  New York steak
</I>

Difficult to identify structure/semantics
A first step - XML

Focus on structure/semantics instead of layout

```xml
<I>
  Microsoft<br>
  One Microsoft Way<br>
  Redmond, WA<br>
</I>

"Microsoft mailing address"

```xml
<address>
  <company name="Microsoft">
    <street>One Microsoft way</street>
    <city>Redmond</city>
    <state>WA</state>
  </company>
</address>

address[.*name="Microsoft"]
Example Query: “retrieve the experiments section of papers on XML, where the related work section references papers on histograms”
The Revolution
Challenges for XML

- Data is tree-structured
- Data is self-describing => Schema Chaos!
- Need to revise several aspects of the system
  - New query language
  - New optimization techniques
  - New system architecture
DB Research at UCSC
Group

- 3 Faculty
- Several PhD, MS, and undergraduates
Projects

- Data Integration
- Data Provenance
- Approximate Query Answering
- Top-k Queries
- Self-Tuning Systems
Conventional Queries

A conventional query retrieves all the results

SELECT r.name, h.name
FROM restaurant r, hotel h
WHERE r.city = 'Santa Cruz' AND r.city = h.city
Top-k Queries

- Top-k Query retrieves only the most "important" results

```
SELECT r.name, h.name
FROM restaurant r, hotel h
WHERE r.city = 'Santa Cruz' AND r.city = h.city
RANK BY .5/r.price + .5*h.rating
LIMIT 10
```
Obvious method: Generate all results, sort, retain top-k
  
  Why generate all the output to return only k tuples?

More efficient method: Rank Joins
  
  Generate the top-k results “on-the-fly”

```sql
SELECT r.name, h.name
FROM restaurant r, hotel h
WHERE r.city = 'Santa Cruz' AND r.city = h.city
RANK BY .3/r.price + .2*r.rating + .5*h.rating
LIMIT 10
```
Problem: users do not want to “lose” their data

Solution: make the system storage-less!

- The system processes data in-place
- No need to import/export

**CHAMELEON-DB**

![Diagram showing DBMS and Data in-box vs. out-box]
Challenges

- How can the DBMS ensure efficiency?
  - Data has to be indexed on-the-fly
- What about query optimization?
  - Need to build cost models on top of files
- What about system administration?
  - Make the system self-tuning
COLT

- Continuous On-Line Tuning
- Key idea: monitor the query load and install indexes on the fly
- Challenge #1: Identify useful indices
- Challenge #2: Control overhead
Τέλος!

- Database classes
  - CMPS 180
  - CMPS 181
- Weekly database seminar
  - Subscribe to db-seminar@soe
- Come and talk to us!