Basic Concepts of a Database System

• Database

A collection of interrelated data accessible by multiple users or multiple purposes

• Database Management System (DBMS)

Software that allows one or many persons to use and/or modify data

• Database system = Database + DBMS
Schemes and Instances

• Scheme of a database
  => structure of a database
    (structural skeleton)

• Instance of a database
  => current content of the database

• Programming Language analogy
  type customer = Record {…..}
  var customer1: customer
Data Model

• A formalism for describing the logical structure of a database and operation on the database

• Programming language analogy:

  var customer1: customer;
  ↑               ↑
  instance scheme

PASCAL(programming Language) ≡ data model
Levels of Abstraction in DBMS

User group1 ← View 1

User group2 ← View 2

User Group n ← View n

External

Conceptual Database

Implemented in data Definition Languages

Physical Database

Implemented on physical devices
Database Languages

• Data Definition Language (DDL)
  to describe a scheme of a database

• Data Manipulation Language (DML)
  to manipulate (retrieve, insert, delete & modify) a database instance
  → non-procedural (declarative)
  → procedural

• Query Language
  Interactive DML
• Host Language

Programming language in which Statements in a DML can be embedded (e.g. C)
Classification of DBMS

- From viewpoint of Data Models:
  - simple flat tables
  - Hierarchical DBMS
  - Network DBMS
  - Relational DBMS

- From a viewpoint of Control:
  - Centralized DBMS
  - Distributed DBMS (DDBMS)
    - heterogeneous DDBMS
    - homogeneous DDBMS
Important Database Properties

• Data Abstraction
  (hide storage detail from user)
  → current trend: operation abstraction in object-oriented systems

• Security and Authorization
  (file systems ok for all/nothing access)
  But often want to grant selected field access
  Grant student.advisor ACCESS(+R) to student.grade
• Control of Redundancy

1. Duplication of effort
   repetition of same data in multiple places
2. Waste of storage space
3. May lead to inconsistencies
   change (phone #) one place, change everywhere?
   (Date of Birth)
situation with
   Rosie Donnaldson  8/26/65   TA
   Rose  Donnaldson  8/26/65   Student Record

Controlled Redundancy may be useful.
   →case where difference values of field for different recs
A Brief history of Database Technology

• Flat Databases
  early 1900’s – The Punch card
    - fixed fields for storing data
    - (initially) 1 record per card

1945 – Magnetic Tape
  (Punchcards on tape, but allows faster search + sorting)
Flat databases motivated by Punch cards

• The Record key – allowed flat records to continue on multiple punchcards (facilitated sorting)

(Wider records on magnetic tape)
Flat Databases are still with us

• Popular PC database programs of 70’s/80’s
  Dbase                   (fixed field widths)
  DB2
Flat Databases are still with us
Flat Databases are still with us

<table>
<thead>
<tr>
<th>Num</th>
<th>Field Name</th>
<th>Field Type</th>
<th>Width</th>
<th>Dec</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACCOUNT</td>
<td>Character</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>COMPANY</td>
<td>Character</td>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ADDRESS</td>
<td>Character</td>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CITY</td>
<td>Character</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>STATE</td>
<td>Character</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ZIP</td>
<td>Character</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>TELEPHONE</td>
<td>Character</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CONTACT</td>
<td>Character</td>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>EQUIPMENT</td>
<td>Character</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>FIRSTORDER</td>
<td>Date</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>LASTORDER</td>
<td>Date</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>CREDITOK</td>
<td>Logical</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>YTOORDERS</td>
<td>Numeric</td>
<td>10</td>
<td>Z</td>
<td>N</td>
</tr>
<tr>
<td>14</td>
<td>TOTORDERS</td>
<td>Numeric</td>
<td>10</td>
<td>Z</td>
<td>N</td>
</tr>
<tr>
<td>15</td>
<td>COMMENTS</td>
<td>Memo</td>
<td>10</td>
<td></td>
<td>N</td>
</tr>
</tbody>
</table>
Flat Databases are still with us
Flatt Databases are still with us
Flat Databases – where else?
Flat Databases – where else?

<table>
<thead>
<tr>
<th>Name</th>
<th>SSN</th>
<th>HW1</th>
<th>HW2</th>
<th>HW3</th>
<th>FP</th>
<th>FE</th>
</tr>
</thead>
<tbody>
<tr>
<td>……</td>
<td>….</td>
<td>99</td>
<td>83</td>
<td>58</td>
<td>22</td>
<td>92</td>
</tr>
<tr>
<td>….</td>
<td>….</td>
<td>78</td>
<td>84</td>
<td>92</td>
<td>33</td>
<td>91</td>
</tr>
<tr>
<td>.....</td>
<td>.....</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• AWK/GAWK/Perl
  - Variable field widths
  - separated by tab character (or equiv.)

**Advantages:**
  easy to code (simple data model)
  efficient to index + access (data all in one place)
  deletion/insertion easy (if fixed width)
  → fewer dependencies

**Disadvantages:**
  - consistency management
  - Redundancy
History (continued)

• 1970  Relational Data Model
  
  Ted Codd, IBM research fellow
  
  square
  Sequel(SQL) \{ relational
  QBE \{ languages
  Quel

  System R – IBM \} Relational
  INGRES - Berkeley \} Research projects
History (continued)

• 1964 – GE Integrated Data Store (IDS)
  Bachman, *Network*. Data Model

• 1965 – IBM Information Management System (IMS)
  Hierarchical Data Mode

• Late 60’s – SABRE (IBM + American Airline)
  First large distributed database with intense concurrency and transaction control needs