CS360
Business Information Systems Analysis and Modeling

System Models and Entity Relationship Models
Modeling

- the capture of a subset of system characteristics relevant to a level or understanding or detail
System Models

- System Models emphasize processing user interfaces

  - User’s System Diagram
    - icon based
    - clearly describe user familiar system interfaces / controls
    - a gentle introduction to formal system specifications

- Menu Trees
  - based on commonly used GUI control interfaces
  - focuses on actions / commands available to the user
  - categorizes and groups related system functions

- System Flow Charts
  - ISO system components
  - focuses on relationships between data and software components
  - identifies configuration management items that must be maintained and controlled
User’s System Diagram

1. Jelly Bean Company
2. Jelly Bean Order
3. Ordering
   - Weekly
   - Daily
4. Inventory
5. Backup
6. Customer History
7. Retail Sales
8. Order Manager
9. Customers
10. Purchase Request
11. Review/Reorder
Menu Tree

User interface function list

Main Jelly Bean Menu

SALES
- Purchase
  - Cash
  - Credit
- Refund

INVENTORY
- Item Inquiry
  - Reorder
  - Order Status

MAINTENANCE
- Backup
  - Reload
  - Recover
- Purge Data
System Flowchart

Relationships between data and software inputs and outputs

Orders Screen → orders program → pending orders

Sales Screen → purchase program → sales receipt

orders program → inventory

inventory → customer history

customer history → recovery program

recovery program → backup program

backup program → backup

backup → orders program
Subsystem Detailing

- Subsystems allow more focus on related operations in the context of the whole

Diagram:
- Orders Screen
- Sales Screen
- Orders program
- purchase program
- pending orders
- inventory
- customer history
- sales receipt
- backup program
- recovery program
- backup
Data Modeling

* Information Structure Support
  * facts
    * measurements / description
    * state designation
    * identifying
  * relationship
  * association
  * dependency
  * aggregation / membership

* Information Processing Support
  * collection
    * validation (existence, metric, range)
    * distinction (replacement, addition of facts)
  * retrieval
    * by identity
    * by classification
  * maintenance
    * by identity and classification
Data Modeling Paradigms

* **Hierarchical**
  * relationships based on parent/child
  * organizational chart
  * family tree

* **Relational**
  * relationships based upon information content
  * shared / common descriptive characteristics
  * identifying facts
  * formally consistent table structure
  * formal mathematics for information operations
  * relational algebra / calculus
  * non-procedural relational languages (SQL, QUEL, QBE)

* **Network**
  * relationships based upon association
  * telephone network
  * network of friends or business associates
Relational Data Model

* **Formal Data Structuring Discipline**
  * columns representing simple facts (attributes)
  * rows representing related facts (instances)
  * related facts defining an “entity”
  * rows + columns form TABLE aka RELATION

* **Mathematical Operations on Relations**
  * Projection: copying columns from tables
  * Selection: copying rows from tables
  * Join: building new rows by copying rows from two tables based on shared information

* **Information Protection through Integrity Constraints**
  * attributes are atomic
  * instances must be “distinguished” in a table
  * joins limited to “join compatible” attributes
  * attributes “clusters” based on functional dependency
Zoo Database

* Zoo keeper
  * feeds animals according to their diet
  * animals are housed in designated cages
  * meals are determined by a ration of foods
  * rations are taken from available food stores
Relational Table

* Tables, attributes, keys, instances

**Entity w/attributes**

<table>
<thead>
<tr>
<th>ANIMAL</th>
<th>ASPECIES</th>
<th>CLOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEO</td>
<td>LION</td>
<td>B2.1</td>
</tr>
<tr>
<td>MEO</td>
<td>LION</td>
<td>B2.2</td>
</tr>
<tr>
<td>ZEO</td>
<td>LION</td>
<td>B2.2</td>
</tr>
<tr>
<td>BLACKIE</td>
<td>PANTHER</td>
<td>B2.3</td>
</tr>
<tr>
<td>MOLLY</td>
<td>PANTHER</td>
<td>F2.3</td>
</tr>
<tr>
<td>JEFFREY</td>
<td>GIRAFFE</td>
<td>C1.1</td>
</tr>
<tr>
<td>MOLLY</td>
<td>DONKEY</td>
<td>C1.1</td>
</tr>
<tr>
<td>CONRAD</td>
<td>CAMEL</td>
<td>C1.1</td>
</tr>
<tr>
<td>FANNY</td>
<td>ELEPHANT</td>
<td>A2.1</td>
</tr>
<tr>
<td>MORRIS</td>
<td>ELEPHANT</td>
<td>A2.1</td>
</tr>
<tr>
<td>CHUBBLES</td>
<td>HIPPO</td>
<td>B4.1</td>
</tr>
<tr>
<td>SUZY</td>
<td>MONKEY</td>
<td>A3.1</td>
</tr>
<tr>
<td>SALLY</td>
<td>MONKEY</td>
<td>A3.1</td>
</tr>
<tr>
<td>SAM</td>
<td>MONKEY</td>
<td>A3.1</td>
</tr>
<tr>
<td>BOSS</td>
<td>APE</td>
<td>A3.2</td>
</tr>
</tbody>
</table>

Relational Table w/instances

**ANIMAL**

Key → {ANAME, ASPECIES, AGENDER, CLOC}
Relationships

* Relationships associate information found in two (or more) tables and define “cardinality” of the relationship.
Relating Tables

* Keys logically connect information

<table>
<thead>
<tr>
<th>ANIMAL</th>
<th>IS CAGED IN &quot;CLOC&quot;</th>
<th>CAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IS FED WITH &quot;ANAME, ASPECIES&quot;</td>
<td></td>
</tr>
<tr>
<td>RATION</td>
<td>IS DELIVERED TO</td>
<td>FOOD</td>
</tr>
<tr>
<td></td>
<td>CONTAINS &quot;FDESC&quot;</td>
<td></td>
</tr>
</tbody>
</table>

these all happen to be many to one relationships
Relating Tables

- Linkage notation
- Note the join columns on the relationship symbol
Relatability / Join Compatible

All connections are based on table contents

ANIMAL
- ANAME
- ASPECIES
- AGENDER
- CLOC

CAGE
- CLOC
- CTYPE
- CSIZE

RATION
- ANAME
- ASPECIES
- FDESC
- RHOWMUCH
- RHOWOFTEN
- CLOC

FOOD
- FDESC
- FINV
- FUNITS
Primary / Foreign Keys

- **Primary** keys uniquely distinguish rows in a table.
- **Foreign** keys identify rows in another table for the purposes of joining.

**ANIMAL**
- ANAME
- ASPECIES
- AGENDER
- CLOC

**CAGE**
- CLOC
- CTYPE
- CSIZE

**RATION**
- ANAME
- ASPECIES
- FDESC
- RHOWMUCH
- RHOWOFTEN
- CLOC

**FOOD**
- FDESC
- FINV
- FUNITS
Data Dictionary

* Capture the full breadth of data definition
  * catalogs tables
  * purpose
  * authority
  * data sources
  * catalogs attributes
  * type
  * valid value ranges
  * formatting

* Defines database
  * Oracle
  * Access
  * Filemaker
  * etc.

Animal = \( \text{aname} + \text{aspecies} + \text{agender} + \text{cloc} \)

Cage = \( \text{cloc} + \text{ctype} + \text{csize} \)

Food = \( \text{fdesc} + \text{finv} + \text{funits} \)

Ration = \( \text{aname} + \text{aspecies} + \text{fdesc} + \text{rhowmuch} + \text{rhowoften} \)

LJ Waguespack, Ph.D. 2017
Normalization

* Normal forms confirm that the structure of the relational tables protects against various anomalies in retrieval / update

* First normal form
  * all attributes are “atomic”
  * no repeating groups

* Second normal form
  * all non-key attributes are determined by the primary key of that table
  * if the primary key is composite: the whole key

* Third normal form
  * there are no “transitive” relationships in any table
  * dependencies are on the primary key only
CASE Support

* Data base administration
  * many case tools not only allow the analyst to document the structure and contents of the data flows and stores in their model but define the data representation in production database tools
  * maintaining current documentation of database structure and contents is essential to protecting the information assets that a database represents
  * many case tools provide migration tools to move an applications data from one database tool to another with operational obsolescence become a problem
  * visual programming environment require an automated data dictionary to support the “drag and drop” programming of queries and reports