Basic Database Concepts

• What is a *database*?
  A database is a collection of related data stored in a computer managed by a DBMS.

• What is a *DBMS, Database Management System*?
  A DBMS is a collection of programs for creating, searching, updating and maintaining large databases.
DBMS architecture
• DBMS technology *central* in the IT age
  More information => more need for DBMS tools
  DBMS companies leaders in IT industry (e.g. ORACLE)
• Examples of *services* provided by DBMSs
  *Creating meta-data* (database design) that
describes data in database as *schema*
  *Changing meta-data* (schema evolution)
  *Querying* database content in terms of schema
  *Updating* database content in terms of schema
  Simple-to-use *concurrent updates* (transactions)
  *Crash recovery, non-stop* functionality
  *Authorization, security, data integrity*
Meta-data definition

• DBMS always provides *data model* which is primitives (i.e. a language) to define a database *schema* (= *meta-data*) describing database contents.
• E.g. the *relational data model* represents everything as *tables* and *relational database schemas* describe *table* properties.
• *Conceptual Data Model*:
  A very high level and user-oriented (graphical) data model.
  A common one is *Extended Entity-Relationship* (ER) data model.
• *NOTICE*: The term *Data Model* sometimes used to mean what DBMS people call *Schema* or *Information Model*. 
Extended ER schema
Database Design

- **Conceptual Database Design**
  Define ER-schema based on application area

- **Logical Database Design**:
  Translate ER-schema to relational tables

- **Physical Database Design**
  Provide storage information to make database fast
Chapter 2: Entity-Relationship Model

- Entity Sets (kinds)
- Relationship Sets (kinds)
- Design Issues
- Mapping Constraints (cardinality constraints)
- Keys
- E-R Diagram

NOTICE: Object-oriented schemas also have *inheritance*
Entity Sets (kinds)

- A database can be modeled as:
  - a collection of entities,
  - relationship among entities.
- An entity is an object that exists and is distinguishable from other objects.
  
  Example: specific person, company, event, plant
- Entities have attributes
  
  Example: people have names and addresses
- An entity set (kind, type) is a set of entities of the same type that share the same properties.
  
  Example: set of all persons, companies, trees, holidays
## Entity Sets *customer* and *loan*

<table>
<thead>
<tr>
<th>customer-id</th>
<th>customer-name</th>
<th>customer-street</th>
<th>customer-city</th>
<th>loan-number</th>
<th>loan-amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>321-12-3123</td>
<td>Jones</td>
<td>Main</td>
<td>Harrison</td>
<td>L-17</td>
<td>1000</td>
</tr>
<tr>
<td>019-28-3746</td>
<td>Smith</td>
<td>North</td>
<td>Rye</td>
<td>L-23</td>
<td>2000</td>
</tr>
<tr>
<td>677-89-9011</td>
<td>Hayes</td>
<td>Main</td>
<td>Harrison</td>
<td>L-15</td>
<td>1500</td>
</tr>
<tr>
<td>555-55-5555</td>
<td>Jackson</td>
<td>Dupont</td>
<td>Woodside</td>
<td>L-14</td>
<td>1500</td>
</tr>
<tr>
<td>244-66-8800</td>
<td>Curry</td>
<td>North</td>
<td>Rye</td>
<td>L-19</td>
<td>500</td>
</tr>
<tr>
<td>963-96-3963</td>
<td>Williams</td>
<td>Nassau</td>
<td>Princeton</td>
<td>L-11</td>
<td>900</td>
</tr>
<tr>
<td>335-57-7991</td>
<td>Adams</td>
<td>Spring</td>
<td>Pittsfield</td>
<td>L-16</td>
<td>1300</td>
</tr>
</tbody>
</table>
Attributes

- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set.
  
  Example:

  \[
  \text{customer} = (\text{customer-id, customer-name, customer-street, customer-city}) \\
  \text{loan} = (\text{loan-number, amount})
  \]

- Domain – the set of permitted values for each attribute

- Attribute types:
  - Simple and composite attributes.
  - Single-valued and multi-valued attributes
    - E.g. multi-valued attribute: phone-numbers

NOTICE: Relational database only allow simple attributes
Composite Attributes

NOTICE: Relational database do not permit composite attributes
=> Need to convert composite attributes to atomic ones (1st normal form)
Relationship Sets (kinds, types)

- A relationship is an association between several entities
  Example:
  \[
  \text{Hayes} \quad \text{depositor} \quad \text{A-102}
  \]
  customer entity relationship set account entity

- A relationship set (kind, type) is a mathematical relation among \( n \geq 2 \) entities, each taken from entity sets
  \[
  \{(e_1, e_2, \ldots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}
  \]

  where \((e_1, e_2, \ldots, e_n)\) is a relationship

  - Example:
    \[
    (\text{Hayes, A-102}) \in \text{depositor}
    \]
Relationship Set *borrower*

```
<table>
<thead>
<tr>
<th></th>
<th>customer</th>
<th>loan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jones</td>
<td>321-12-3123</td>
<td>L-17</td>
</tr>
<tr>
<td>Smith</td>
<td>019-28-3746</td>
<td>L-23</td>
</tr>
<tr>
<td>Hayes</td>
<td>677-89-9011</td>
<td>L-15</td>
</tr>
<tr>
<td>Jackson</td>
<td>555-55-5555</td>
<td>L-14</td>
</tr>
<tr>
<td>Curry</td>
<td>244-66-8800</td>
<td>L-19</td>
</tr>
<tr>
<td>Williams</td>
<td>963-96-3963</td>
<td>L-11</td>
</tr>
<tr>
<td>Adams</td>
<td>335-57-7991</td>
<td>L-16</td>
</tr>
</tbody>
</table>
```
Relationship Sets (Cont.)

- An *attribute* can also be property of a relationship set.
- For instance, the *depositor* relationship set between entity sets *customer* and *account* may have the attribute *access-date*
Degree of a Relationship Set

- Refers to number of entity sets that participate in a relationship set.
- Relationship sets that involve two entity sets are *binary* (or degree two). Generally, most relationship sets in a database system are binary.
- Relationship sets may involve more than two entity sets.
  - E.g. Suppose employees of a bank may have jobs (responsibilities) at multiple branches, with different jobs at different branches. Then there is a ternary relationship set between entity sets *employee, job and branch*.
- Relationships between more than two entity sets (ternary relationships) are rare. Most relationships are binary. Ternary relationships can always be converted to binary ones.
Mapping Cardinalities
(Cardinality Constraints)

- Express the number of entities to which another entity can be associated via a relationship set.
- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to one
  - One to many
  - Many to one
  - Many to many
Mapping Cardinalities

Note: Some elements in A and B may not be mapped to any elements in the other set.
Mapping Cardinalities

(a) Many to one

(b) Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set.
E-R Diagrams

- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets (kinds).
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
  - **Double ellipses** represent multivalued attributes.
  - **Dashed ellipses** denote derived attributes.
- **Underline** indicates primary key attributes (will study later)
E-R Diagram With Composite, Multivalued, and Derived Attributes
Relationship Sets with Attributes
Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (→), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set.

- E.g.: One-to-one relationship:
  - A customer is associated with at most one loan via the relationship `borrower`
  - A loan is associated with at most one customer via `borrower`
One-To-Many Relationship

- In the one-to-many relationship a loan is associated with at most one customer via borrower, a customer is associated with several (including 0) loans via borrower.
Many-To-One Relationships

- In a many-to-one relationship a loan is associated with several (including 0) customers via borrower, a customer is associated with at most one loan via borrower.
A customer is associated with several (possibly 0) loans via borrower

A loan is associated with several (possibly 0) customers via borrower
Keys

- A super key of an entity set is a set of one or more attributes whose values uniquely determine each entity.
- A candidate key of an entity set is a minimal super key
  - Customer-id is candidate key of customer
  - account-number is candidate key of account
- Although several candidate keys may exist, one of the candidate keys is selected to be the primary key and it is underlined in ER-diagram.
E-R Diagram with a Ternary Relationship

NOTICE: Ternary Relationships can always be converted to binary ones by creating new entities from relationships