Data Model

- **Entities**: noun phrases, have attributes
- **Relationships**: characterize relationships amongst entities
- **Attributes**: Think of them as features
Attributes

Default attributes are our initial interpretation of attributes, really just features like birthdate, eye color, etc.

Multivalued attributes are features entities might have more than one of.

Derived attributes are implicit. If we’re given birthdate then we know age.
## Relationship Cardinalities

<table>
<thead>
<tr>
<th>Type</th>
<th>Illustrated</th>
<th>Multiple Loans?</th>
<th>Joint Loans?</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-to-One (1:1)</td>
<td><img src="image" alt="Borrows" /></td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Many-to-one (n:1)</td>
<td><img src="image" alt="Borrows" /></td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>One-to-many (1:n)</td>
<td><img src="image" alt="Borrows" /></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Many-to-many (n:m)</td>
<td><img src="image" alt="Borrows" /></td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Relationship Cardinalities Cont.

1:1

n:1

1:n

n:m
## Relationship Cardinalities Cont.

<table>
<thead>
<tr>
<th>Relationship Cardinality</th>
<th>Relational Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>n:m</td>
<td>E₁ = (a₁, ..., aₙ)</td>
</tr>
<tr>
<td></td>
<td>E₂ = (b₁, ..., bₚ)</td>
</tr>
<tr>
<td></td>
<td>R = (a₁, b₁, c₁, ..., cₖ)</td>
</tr>
<tr>
<td>n:1</td>
<td>E₁ = (a₁, ..., aₙ, b₁, c₁, ..., cₖ)</td>
</tr>
<tr>
<td></td>
<td>E₂ = (b₁, ..., bₚ)</td>
</tr>
<tr>
<td>1:n</td>
<td>E₁ = (a₁, ..., aₙ)</td>
</tr>
<tr>
<td></td>
<td>E₂ = (b₁, ..., bₚ, a₁, c₁, ..., cₖ)</td>
</tr>
</tbody>
</table>
Keys

- **Superkey**: any key that distinguishes each tuple
- **Candidate**: “minimal” super key. The super key with the fewest possible attributes.
- **Primary**: Candidate key chosen by the DBA
Existence Dependencies

A **weak entity set** depends on the existence of some other entity set. They have no *inherent* superkeys and are discriminated by the fact that each of the is related to some loan. Use discriminators instead of keys.

The **identifying relationship** connects an entity set with its weak entity set.

Here, the payment is **existence dependent** on the loan.

**Total participation** between an entity and relationship indicates that every tuple in the entity has some association with the relationship.
Specialization v Generalization

**Overlapping**: a person can belong to two subclasses

**Disjoint**: a person doesn’t fit in any of the subclasses

**Total Membership**: every person belongs to some subclass

**Partial Membership**: some people don’t belong to any subclass

**User Defined**: The user arbitrarily decides who fits into each subclass

**Condition Defined**: The database can automatically assign a record to its subclass (think age)
Example
Schema

Strong Entity Sets

classroom (building, room_number, capacity)
department (dept_name, building, budget)
course (course_id, title, credits)
instructor (ID, name, salary)
student (ID, name, tot_cred)

Relationship Sets

teaches (ID, course_id, sec_id, semester, year)
takes (ID, course_id, sec_id, semester, year, grade)
prereq (course_id, prereq_id)
advisor (s_ID, i_ID)
sec_course (course_id, sec_id, semester, year)
sec_time_slot (course_id, sec_id, semester, year, time_slot_id)
sec_class (course_id, sec_id, semester, year, building, room_number)
inst_dept (ID, dept_name)
stud_dept (ID, dept_name)
course_dept (course_id, dept_name)