The Relational Model

T. M. Murali

October 5, 2009
Course Outline

- Weeks 1–5, 13: Query/Manipulation Languages
  - The relational model
  - Relational Algebra
  - SQL
  - Data definition
  - Programming with SQL

- Weeks 6–8: Data Modelling
  - Entity-Relationship (E/R) approach
  - Good E/R design
  - Specifying Constraints
  - Converting E/R model to relational model.
Course Outline

- **Weeks 1–5, 13: Query/Manipulation Languages**
  - The relational model
  - Relational Algebra
  - SQL
  - Data definition
  - Programming with SQL

- **Weeks 6–8: Data Modelling**
  - Entity-Relationship (E/R) approach
  - Good E/R design
  - Specifying Constraints
  - Converting E/R model to relational model.
The Relational Model

- Built around a single concept for modelling data: the relation or table.
- Supports high-level programming language (SQL).
- Has an elegant mathematical design theory.
- Most current DBMS are relational.
The Relation

▷ A relation is a two-dimensional table:
  ▷ Relation $\equiv$ table.
  ▷ Attribute $\equiv$ column name.
  ▷ Tuple $\equiv$ row (not the header row).
  ▷ Database $\equiv$ collection of relations.

<table>
<thead>
<tr>
<th>CoursesTaken</th>
<th>Student</th>
<th>Course</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hermione Grainger</td>
<td>Potions</td>
<td>A-</td>
</tr>
<tr>
<td></td>
<td>Draco Malfoy</td>
<td>Potions</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>Harry Potter</td>
<td>Potions</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>Ron Weasley</td>
<td>Potions</td>
<td>C</td>
</tr>
</tbody>
</table>
The schema of a relation is the name of the relation followed by a paranthetised list of attributes.

\[
\text{CoursesTaken}(\text{Student}, \text{Course}, \text{Grade})
\]

A design in a relational model consists of a set of schemas.

- Such a set of schemas is called a relational database schema.
Converting E/R Diagrams to Relational Designs

- Entity set → relation.
  - Attribute of an entity set → attribute of a relation.

- Relationship → relation whose attributes are
  - Attribute of the relationship itself.
  - Key attributes of the connected entity sets.

- Several special cases:
  - Weak entity sets.
  - Combining relations (especially for many-one relationships).
  - *Isa* relationships and subclasses.
Example for Conversion

```
Courses
  Offer
    Departments
      Name

Students
  Take
    Courses
      Number
      Classroom
    Enrollment
      Name

Evaluation
  Grade
    Professor
      Name
      Age
      Office
```
For each entity set, create a relation with the same name and with the same set of attributes.

- Departments
  - Name
- Offer
- Students
  - Name
  - Address
- Take
- Courses
  - Name
  - Classroom
  - Number
  - Enrollment
- Evaluation
  - Grade
- Teach
- Professors
  - Name
  - Age
  - Office
For each entity set, create a relation with the same name and with the same set of attributes.

Students(Name, Address)
For each entity set, create a relation with the same name and with the same set of attributes.

Students(Name, Address)
Professors(Name, Office, Age)
Schemas for Non-Weak Entity Sets

For each entity set, create a relation with the same name and with the same set of attributes.

Students(Name, Address)
Professors(Name, Office, Age)
Departments(Name)
For each weak entity set $W$, create a relation with the same name whose attributes are

- Attributes of $W$ and
- Key attributes of the other entity sets that help form the key for $W$. 
For each weak entity set \( W \), create a relation with the same name whose attributes are

- Attributes of \( W \) and
- Key attributes of the other entity sets that help form the key for \( W \).

\[
\text{Courses(\texttt{Number}, \texttt{DepartmentName}, \texttt{CourseName}, \texttt{Classroom}, \texttt{Enrollment})}
\]
Schemata for Non-Supporting Relationships (1)

- For each relationship, create a relation with the same name whose attributes are
  - Attributes of the relationship itself.
  - Key attributes of the connected entity sets (even if they are weak).
Schemas for Non-Supporting Relationships (2)

- **Departments**
  - Name

- **Offer**
  - Name
  - Address

- **Students**
  - Name
  - Address
  - Grade

- **Courses**
  - Name
  - Number
  - Classroom
  - Enrollment

- **Professors**
  - Name
  - Age
  - Office

- **Take**
  - (StudentName, Address, Number, DepartmentName)

- **Teach**
  - (ProfessorName, Office, Number, DepartmentName)

- **Evaluation**
  - (StudentName, Address, ProfessorName, Office, Number, DepartmentName, Grade)
Schemata for Non-Supporting Relationships (2)

- **Take**

  ![Diagram of relational schema](image)
Schemas for Non-Supporting Relationships (2)

- Take(StudentName, Address, Number, DepartmentName)

- Teach(ProfessorName, Office, Number, DepartmentName)

- Evaluation(StudentName, Address, ProfessorName, Office, Number, DepartmentName, Grade)
Schemas for Non-Supporting Relationships (2)

- Take(StudentName, Address, Number, DepartmentName)
- Teach
Schemas for Non-Supporting Relationships (2)

- Take(StudentName, Address, Number, DepartmentName)
- Teach(ProfessorName, Office, Number, DepartmentName)
Schemas for Non-Supporting Relationships (2)

- Take(StudentName, Address, Number, DepartmentName)
- Teach(ProfessorName, Office, Number, DepartmentName)
- Evaluation
Schemas for Non-Supporting Relationships (2)

- Take(StudentName, Address, Number, DepartmentName)
- Teach(ProfessorName, Office, Number, DepartmentName)
- Evaluation(StudentName, Address, ProfessorName, Office, Number, DepartmentName, Grade)
Roles in Relationships

If an entity set $E$ appears $k > 1$ times in a relationship $R$ (in different roles), the key attributes for $E$ appear $k$ times in the relation for $R$, appropriately renamed.
Roles in Relationships

- If an entity set $E$ appears $k > 1$ times in a relationship $R$ (in different roles), the key attributes for $E$ appear $k$ times in the relation for $R$, appropriately renamed.

PreReq(RequirerNumber, RequirerDeptName, RequirementNumber, RequirementDeptName)
Consider many-one Teach relationship from Courses to Professors.

Schemas are

- Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)
- Professors(Name, Office, Age)
- Teach(Number, DepartmentName, ProfessorName, Office)
Combining Relations

- Consider many-one Teach relationship from Courses to Professors.
- Schemas are:
  - Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)
  - Professors(Name, Office, Age)
  - Teach(Number, DepartmentName, ProfessorName, Office)
- The key for Courses uniquely determines all attributes of Teach.
Combining Relations

- Consider many-one Teach relationship from Courses to Professors.

- Schemas are

  Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)
  Professors(Name, Office, Age)
  Teach(Number, DepartmentName, ProfessorName, Office)

- The key for Courses uniquely determines all attributes of Teach.

- We can combine the relations for Courses and Teach into a single relation whose attributes are
  - All the attributes for Courses,
  - Any attributes of Teach, and
  - The key attributes of Professors.
Rules for Combining Relations

- We can combine into one relation $Q$
  - The relation for an entity set $E$ and
  - all many-to-one relationships $R_1, R_2, \ldots, R_k$ from $E$ to other entity sets $E_1, E_2, \ldots, E_k$, respectively.
**Rules for Combining Relations**

- We can combine into one relation $Q$
  - The relation for an entity set $E$ and
  - all many-to-one relationships $R_1, R_2, \ldots R_k$ from $E$ to other entity sets $E_1, E_2, \ldots, E_k$, respectively.

- The attributes of $Q$ are
  - all the attributes of $E$,
  - any attributes of $R_1, R_2, \ldots R_k$, and
  - the key attributes of $E_1, E_2, \ldots, E_k$. 

Rules for Combining Relations

- We can combine into one relation $Q$
  - The relation for an entity set $E$ and
  - all many-to-one relationships $R_1, R_2, \ldots R_k$ from $E$ to other entity sets $E_1, E_2, \ldots, E_k$, respectively.
- The attributes of $Q$ are
  - all the attributes of $E$,
  - any attributes of $R_1, R_2, \ldots R_k$, and
  - the key attributes of $E_1, E_2, \ldots, E_k$.
- Can we combine $E$ and $R$ if $R$ is a many-many relationship from $E$ to $F$?
Supporting Relationships

- Schema for `Departments` is `Departments(Name)`.
- Schema for `Courses` is `Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)`.

What is the schema for `Offer`?
- `Offer(Name, Number, DepartmentName)`.
- But `Name` and `DepartmentName` are identical, so the schema for `Offer` is `Offer(Number, DepartmentName)`.
- The schema for `Offer` is a subset of the schema for the weak entity set, so we can dispense with the relation for `Offer`.
Supporting Relationships

- Schema for Departments is Departments(Name).
- Schema for Courses is Courses(Number, DepartmentName, CourseName, Classroom, Enrollment).
- What is the schema for Offer?
Supporting Relationships

- Schema for Departments is Departments(Name).
- Schema for Courses is Courses(Number, DepartmentName, CourseName, Classroom, Enrollment).
- What is the schema for Offer?
  - Offer(Name, Number, DepartmentName).
Supporting Relationships

- Schema for Departments is Departments(Name).
- Schema for Courses is Courses(Number, DepartmentName, CourseName, Classroom, Enrollment).
- What is the schema for Offer?
  - Offer(Name, Number, DepartmentName).
  - But Name and DepartmentName are identical, so the schema for Offer is Offer(Number, DepartmentName).
Supporting Relationships

- Schema for Departments is Departments(Name).
- Schema for Courses is Courses(Number, DepartmentName, CourseName, Classroom, Enrollment).
- What is the schema for Offer?
  - Offer(Name, Number, DepartmentName).
  - But Name and DepartmentName are identical, so the schema for Offer is Offer(Number, DepartmentName).
  - The schema for Offer is a subset of the schema for the weak entity set, so we can dispense with the relation for Offer.
Summary of Weak Entity Sets

- If $W$ is a weak entity set, the relation for $W$ has a schema whose attributes are:
  - all attributes of $W$,
  - all attributes of supporting relationships for $W$, and
  - for each supporting relationship for $W$ to an entity set $E$, the key attributes of $E$.

- There is no relation for any supporting relationship for $W$. 
ISA to Relational

Three approaches:
1. E/R viewpoint
2. Object-oriented viewpoint
3. “Flatten” viewpoint
Rules Satisfied by an ISA Hierarchy

- The hierarchy has a root entity set.
- The root entity set has a key that identifies every entity represented by the hierarchy.
- A particular entity can have components that belong to entity sets of any subtree of the hierarchy, as long as that subtree includes the root.
Example ISA hierarchy

Students
- ID
- Name

Courses
- Offered_by
- Departments
- Name
- Number

Undergraduates
- UTA_for
- Major

Graduates
- GTA_for
- Major

Masters
- Thesis_title_MS

PhDs
- Thesis_title_PhD
ISA to Relational Method I: E/R Approach

- Create a relation for each entity set.
- The attributes of the relation for a non-root entity set $E$ are
  - the attributes forming the key (obtained from the root) and
  - any attributes of $E$ itself.
- An entity with components in multiple entity sets has tuples in all the relations corresponding to these entity sets.
- Do not create a relation for any isa relationship.
- Create a relation for every other relationship.
ISA to Relational Method I: E/R Approach Example
ISA to Relational Method I: E/R Approach Example

The diagram illustrates the conversion of an ISA hierarchy to the relational model. Here's a breakdown of the entities and their attributes:

**Students**
- ID
- Name
- Undergraduates
- Graduates
- Masters
- PhDs
- UTA_for
- GTA_for
- Offered_by
- Departments
- Thesis_title_MS
- Thesis_title_PhD
- Major

**Courses**
- Name
- Number

The diagram shows the relationships and attributes involved in converting an ISA hierarchy into a relational model.
ISA to Relational Method I: E/R Approach Example

Students

<table>
<thead>
<tr>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undergraduates</td>
<td>ID, Major</td>
</tr>
<tr>
<td>Graduates</td>
<td>ID, Thesis_title_MS</td>
</tr>
<tr>
<td>Masters</td>
<td>ID, Thesis_title_MS</td>
</tr>
<tr>
<td>PhDs</td>
<td>ID, Thesis_title_PhD</td>
</tr>
<tr>
<td>UTA_for</td>
<td>ID, CourseNumber, DepartmentName</td>
</tr>
<tr>
<td>GTA_for</td>
<td>ID, CourseNumber, DepartmentName</td>
</tr>
</tbody>
</table>
ISA to Relational Method I: E/R Approach Example

Students

Undergraduates

Graduates

Masters

GTA_for

UTA_for

Courses

Departments

Offered_by

Thesis_title_MS

Thesis_title_PhD

Major

Number

Name

Name
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates

UTA_for (ID, CourseNumber, DepartmentName)
GTA_for (ID, CourseNumber, DepartmentName)

Masters
PhDs

Thesis_title_MS
Thesis_title_PhD
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)

Students
- ID
- Name
- Undergraduates
  - ID
  - Major
- Graduates
  - ID
  - Major
- Masters
  - Thesis_title_MS
- PhDs
  - Thesis_title_PhD

Courses
- Offered_by
  - Name
  - Number
- Departments
  - Name

UTA_for
- ID
- CourseNumber
- DepartmentName

GTA_for
- ID
- CourseNumber
- DepartmentName
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters

Courses
Offered_by
Name
Number
Departments

Students

Undergraduates

UTA_for

GTA_for

Graduates

Thesis_title_MS
Masters

Thesis_title_PhD
PhDs
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
PhDs
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
PhDs(ID, Thesis_title_PhD)
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
PhDs(ID, Thesis_title_PhD)
UTA_for
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
PhDs(ID, Thesis_title_PhD)
UTA_for(ID, CourseNumber, DepartmentName)
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
PhDs(ID, Thesis_title_PhD)
UTA_for(ID, CourseNumber, DepartmentName)
GTA_for
ISA to Relational Method I: E/R Approach Example

Students(ID, Name)
Undergraduates(ID, Major)
Graduates(ID, Major)
Masters(ID, Thesis_title_MS)
PhDs(ID, Thesis_title_PhD)
UTA_for(ID, CourseNumber, DepartmentName)
GTA_for(ID, CourseNumber, DepartmentName)
ISA to Relational Method II: “Flatten” Approach

- Create a *single* relation for the entire hierarchy.
- Attributes are
  - the key attributes of the root and
  - the attributes of each entity set in the hierarchy.
- Handle relationships as before.
ISA to Relational Method II: “Flatten” Approach

- Create a *single* relation for the entire hierarchy.
- Attributes are
  - the key attributes of the root and
  - the attributes of each entity set in the hierarchy.
- Handle relationships as before.

Students(ID, Name, UGMajor, GMajor, Thesis_title_MS, Thesis_title_PhD).
ISA to Relational Method III: Object-Oriented Approach (1)

- Treat entities as objects belonging to a single class.
- “Class” ≡ subtree of the hierarchy that includes the root.
- Enumerate all subtrees of the hierarchy that contain the root.
- For each such subtree,
  - Create a relation that represents entities that have components in exactly that subtree.
  - The schema for this relation has all the attributes of all the entity sets in that subtree.
- Schema of the relation for a relationship has key attributes of the connected entity sets.
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

- Students
- Undergraduates
- Graduates
- Courses
- UTA_for
- GTA_for
- Thesis_title_MS
- Thesis_title_PhD
- Masters
- PhDs
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

Students

Students

ID
Name

UTA_for
GTA_for

Undergraduates
Graduates

Masters
PhDs

Offered_by
Departments

Name
Number

Major

Thesis_title_MS

Major

Thesis_title_PhD
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

Students(ID)
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

Students(ID)

StudentsUGs
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

- Students(ID)
- StudentsUGs(ID, Major)
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are:
- Students(ID)
- StudentsUGs(ID, Major)
- StudentsGs

![Diagram showing the relationship between Students, Undergraduates, Graduates, Courses, and Departments]
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

Students(ID)
StudentsUGs(ID, Major)
StudentsGs(ID, Major)
Subtrees are

- Students(ID)
- StudentsUGs(ID, Major)
- StudentsGs(ID, Major)
- StudentsGsMasters

ISA to Relational Method III: Object-Oriented Approach (2)
ISA to Relational Method III: Object-Oriented Approach (2)

- Subtrees are
  Students(ID)
  StudentsUGs(ID, Major)
  StudentsGs(ID, Major)
  StudentsGsMasters(ID, Major, Thesis_title_MS)
ISA to Relational Method III: Object-Oriented Approach (2)

▶ Subtrees are

Students(ID)
StudentsUGs(ID, Major)
StudentsGs(ID, Major)
StudentsGsMasters(ID, Major, Thesis_title_MS)
StudentsGsPhDs
ISA to Relational Method III: Object-Oriented Approach (2)

Subtrees are

- Students(ID)
- StudentsUGs(ID, Major)
- StudentsGs(ID, Major)
- StudentsGsMasters(ID, Major, Thesis_title_MS)
- StudentsGsPhDs(ID, Major, Thesis_title_PhD)
ISA to Relational Method III: Object-Oriented Approach (2)

- Subtrees are
  - Students(ID)
  - StudentsUGs(ID, Major)
  - StudentsGs(ID, Major)
  - StudentsGsMasters(ID, Major, Thesis_title_MS)
  - StudentsGsPhDs(ID, Major, Thesis_title_PhD)
  - StudentsUGsGsMasters
Subtrees are

- Students(ID)
- StudentsUGs(ID, Major)
- StudentsGs(ID, Major)
- StudentsGsMasters(ID, Major, Thesis_title_MS)
- StudentsGsPhDs(ID, Major, Thesis_title_PhD)
- StudentsUGsGsMasters(ID, UGMinor, GradMinor, Thesis_title_MS)
Subtrees are

- Students(ID)
- StudentsUGs(ID, Major)
- StudentsGs(ID, Major)
- StudentsGsMasters(ID, Major, Thesis_title_MS)
- StudentsGsPhDs(ID, Major, Thesis_title_PhD)
- StudentsUGsGsMasters(ID, UGMinor, GradMinor, Thesis_title_MS)

What other subtrees exist?
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.

- Number of relations for \( n \) entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten:
  - E/R:
  - OO: can be
Answering queries
- It is expensive to answer queries involving several relations.
- Queries about Students in general.
- Queries about a particular subclass of Students.

Number of relations for $n$ entities in the hierarchy.
- We like to have a small number of relations.
- Flatten: 1.
- E/R:
- OO: can be
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.

- Number of relations for \( n \) entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten: 1.
  - E/R: \( n \).
  - OO: can be
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.

- Number of relations for \( n \) entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten: 1.
  - E/R: \( n \).
  - OO: can be \( 2^n \).
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.

- Number of relations for \( n \) entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten: 1.
  - E/R: \( n \).
  - OO: can be \( 2^n \).

- Redundancy and space usage
  - OO:
  - Flatten:
  - E/R:
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.

- Number of relations for $n$ entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten: 1.
  - E/R: $n$.
  - OO: can be $2^n$.

- Redundancy and space usage
  - OO: Only one tuple per entity.
  - Flatten: .
  - E/R: .
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.
- Number of relations for $n$ entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten: 1.
  - E/R: $n$.
  - OO: can be $2^n$.
- Redundancy and space usage
  - OO: Only one tuple per entity.
  - Flatten: May have a large number of NULLs.
  - E/R:
ISA to Relational: Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations.
  - Queries about Students in general.
  - Queries about a particular subclass of Students.

- Number of relations for $n$ entities in the hierarchy.
  - We like to have a small number of relations.
  - Flatten: 1.
  - E/R: $n$.
  - OO: can be $2^n$.

- Redundancy and space usage
  - OO: Only one tuple per entity.
  - Flatten: May have a large number of NULLs.
  - E/R: Several tuples per entity, but only key attributes repeated.