Chapter 7

Advanced Topics in Database Systems

Emmanuel Stefanakis
http://www2.unb.ca/~estef/
Advanced Conceptual Modeling

Emmanuel Stefanakis
estef@unb.ca
Topics

Advanced Database Design...

- Special Cases of ER Diagrams
- Enhanced ER-model
- UML
Conceptual Schema Design

- Special cases of ER-diagrams
  - Recursive relationships…
Conceptual Schema Design

- **Special cases of ER-diagrams**
  - Recursive relationships…
Conceptual Schema Design

- Special cases of ER-diagrams
  - Dependent entity/relationship types...
Conceptual Schema Design

• Special cases of ER-diagrams
  – Relationships of degree > 2
Conceptual Schema Design

- Special cases of ER-diagrams
  - Relationships of degree > 2
Conceptual Schema Design

- Special cases of ER-diagrams
  - Relationships of degree > 2
O2 client in both banks
Both banks mortgage P1
So …O2+B1+P1  → wrong
Conceptual Schema Design

- Special cases of ER-diagrams
  - Relationships of degree > 2
Topics

*Advanced Database Design...*

- Special Cases of ER Diagrams
- Enhanced ER-model
- UML
Conceptual Schema Design

• Enhanced-ER Model (EER-model)
  – ER cannot support new technologies ...
    • GIS, CAD/CAM, Graphics, Multimedia, etc.
  – EER provides the additional concepts…
    • class-subclass relationships
    • class inheritance
    • specialization/generalization
    • category
Conceptual Schema Design

- **Enhanced-ER Model (EER-model)**
  - **Super-classes and Sub-classes**
    - entities of an entity type can be further grouped into other entity types, e.g.,
Conceptual Schema Design

• Enhanced-ER Model (EER-model)
  – Super-classes and Sub-classes

![Diagram of Enhanced-ER Model showing super-classes and sub-classes]
Conceptual Schema Design

- **Enhanced-ER Model (EER-model)**
  - Class-Subclass Relationship vs 1:1 relationship
Conceptual Schema Design

• Enhanced-ER Model (EER-model)
  – Class Inheritance
    • the subclass inherits from its super-class ...
      – all attributes
      – all relationship instances
    • example...
Conceptual Schema Design

- Enhanced-ER Model (EER-model)
Conceptual Schema Design

• Enhanced-ER Model (EER-model)
  – Specialization and Generalization
    • two concepts closely related to the concept of inheritance
    • specialization …
      – definition of subclasses for one class
    • generalization …
      – definition of a super-class for a set of classes
Conceptual Schema Design

- **Enhanced-ER Model (EER-model)**
  - Specialization ...
Conceptual Schema Design

• **Enhanced-ER Model (EER-model)**
  - A specialization can be …
    • overlapping
    • disjoint
  - A specialization can be …
    • total
    • partial
Conceptual Schema Design

- **Enhanced-ER Model (EER-model)**

  ![Diagram showing total disjoint and total overlapping relationships between superclass and subclasses.]

  - total disjoint
  - partial disjoint
  - total overlapping
  - partial overlapping
Conceptual Schema Design

- Enhanced-ER Model (EER-model)
  - An example ...

```
OWNERS
  ^
  | d
  v
PERSONS  LEGAL_ENTITIES
  |                |
  v                v
PUBLIC  PRIVATE
```

or. NGO
Conceptual Schema Design

- **Enhanced-ER Model (EER-model)**
  - Generalization ...
Conceptual Schema Design

- Enhanced-ER Model (EER-model)
  - Generalization ...
Conceptual Schema Design

- Enhanced-ER Model (EER-model)
  - Specialization Network Example
Conceptual Schema Design

- **Enhanced-ER Model (EER-model)**
  - **Categories …**
    - a class-subclass relationship with more than one entity type in the role of the superclass
    - a category can be …
      - total
      - partial
Conceptual Schema Design

• Enhanced-ER Model (EER-model)
  – Categories…

All FL & AS → NBP
Some B & P → CE
Conceptual Schema Design

• Enhanced-ER Model (EER-model)
  – (total) category & its reverse specialization…
Mapping ER to Relational schema

• Standard steps…
  – Step 1: Entity types $\rightarrow$ new relations (tables)
  – Step 2: Relationships 1:1 (foreign keys)
  – Step 3: Relationships 1:N (foreign keys)
  – Step 4: Relationships N:M $\rightarrow$ new relation (table)
  – Step 5: Multivalued attributes $\rightarrow$ new relation (table)
Mapping E-ER to Relational schema

• Standard steps…
  – Step 6:
    • For each Super-class and Sub-class …
      – create a new relation (table) as of Step 1
      – in all sub-class relations insert the key of the superclass
      – assign the key role to this attribute in all sub-class relations
Mapping E-ER to Relational schema

• Standard steps…
  – Step 6:
Mapping E-ER to Relational schema

- Standard steps…
  - Step 6:

<table>
<thead>
<tr>
<th>OWNERS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>REG_DATE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PERSONS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>SURNAME</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LEGAL_ENTITIES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TRADEMARK</td>
<td>HEADQUARTERS</td>
</tr>
</tbody>
</table>
Mapping E-ER to Relational schema

• Standard steps…
  – Step 6:
Mapping E-ER to Relational schema

• **Standard steps…**
  
  – **Step 7:**
  
  • For each category and its super-classes
    
    – create a new relation (table) as of Step 1
    
    – if the super-classes have the same key attribute, insert this key attribute in the category relation (and assign it the key role)
    
    – if the super-classes have a different key attributes, create a new key attribute in the category relation and insert it in the superclass relations as a foreign key
Mapping E-ER to Relational schema

- **Standard steps…**
  - **Step 7:**

```plaintext
+-----------------+-----------------+-----------------+
| COASTAL_ESTATES | BUILDINGS       | PARCELS         |
| ID              | TYPE            | ID              |
| ASPECT          |                 | AREA            |
| DIST_FROM_SEA   |                 |                 |
```
Mapping E-ER to Relational schema

• Standard steps…
  – Step 7:

<table>
<thead>
<tr>
<th>COASTAL_ESTATES</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C_ID</td>
<td>ASPECT</td>
<td>DIST_FROM_SEA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BUILDINGS</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE</td>
<td>TYPE</td>
<td>C_ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PARCELS</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>AREA</td>
<td>C_ID</td>
</tr>
</tbody>
</table>
Topics

*Advanced Database Design...*

- Special Cases of ER Diagrams
- Enhanced ER-model
- UML
Conceptual Schema Design

• **UML (Unified Modeling Language)**
  – a standard for software design
  – many of UML tools are useful in database design
  – class diagrams (concepts) ...
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - class

![Class diagram](image)

```
OWNERS

SIN
SURNAME
NAME
DATE_OF_BIRTH
ADDRESS
STREET
NUMBER
CITY
PROVINCE
ZIP
COUNTRY
TELEPHONES

class name

class attributes

newOwner
remOwner
modAddress
numOwners
ageOwner
telOwner

class methods
```
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - three types of relationships …
    - associations
    - aggregations
    - generalizations
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - association between two entities (classes)...

```
CONCEPTS

- UML (Unified Modeling Language)
  - association between two entities (classes)...
```

```
CONCEPTS

- UML (Unified Modeling Language)
  - association between two entities (classes)...
```
Conceptual Schema Design

• UML (Unified Modeling Language)
  – association between two entities (classes)...

If I pick one owner, s/he is associated with 0 to many parcels

If I pick one parcel, it is associated with 1 to many owners
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - association between two entities (classes)...

(a) Married couples.

(b) All adults.

(c) All (married and single) men and all married women.
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - association between two entities (classes)...
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - aggregation
    - the relationship between an object and its components parts
    - a class can be part of
      - one class (total aggregation)
      - more than one classes (partial aggregation)
Conceptual Schema Design

• **UML (Unified Modeling Language)**
  
  – aggregation …

![Diagram of UML relationships between PARCELS, BUILDING_BLOCKS, and LAND_USE_ZONES showing strong and weak aggregation.](image-url)
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - aggregation … (implicit representation in EER)
    - Through the name of the relationship
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - generalization …
    - class-subclass relationships …
    - a generalization is …
      - overlapping or
      - disjoint
Conceptual Schema Design

- **UML (Unified Modeling Language)**
  - generalization …

```
<table>
<thead>
<tr>
<th>OWNERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIN</td>
</tr>
<tr>
<td>REG_DATE</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PERSONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SURNAME</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>ADDRESS</td>
</tr>
<tr>
<td>DATE_OF_BIRTH</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LEGAL_ENTITIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRADEMARK</td>
</tr>
<tr>
<td>HEADQUARTERS</td>
</tr>
<tr>
<td>LOCATIONS</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PUBLIC_ENTITIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PRIVATE_ENTITIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
```

disjoint

overlapping
Conceptual Schema Design

- **OGC Geometry Class Diagram in UML**

![UML Diagram](image)
Advanced Logical Modeling

Emmanuel Stefanakis
estef@unb.ca
Advanced Logical Database Design...

- Weaknesses of the Relational Model
- The Object-Oriented Model
Logical Schema Design

• **Weak Points of the Relational Model:**

1. **efficiency**
   - information retrieved by addressing multiple tables (join operation is very slow)

2. **data semantics**
   - the relational model lacks semantics
     - cannot distinguish between different types of relationships (association, aggregation, specialization)
     - a column can be either attribute or relationship
Logical Schema Design

• Weak Points of the Relational Model:
  3. model extension
    • relations cannot be used as built-in data types
      (1-NF prevents nested relations)
  4. object identity
    • the method of mixing data values and identity
      is proved dangerous (inconsistency problems)
Logical Schema Design

• Weak Points of the Relational Model:
  5. program interface

\[ \text{O-O concepts} \]

\[ \text{Relational DBMS} \]

gap: it hampers smooth interaction

• differences in the languages used (non-comparable power)
• mismatch in the data types supported by the two systems
Topics

Advanced Logical Database Design...

- Weaknesses of the Relational Model
- The Object-Oriented Model
Logical Schema Design

- **Object-Oriented Model**
  - basic concept: the object
  - an object has three components...
Logical Schema Design

- **Object-Oriented Model**
  - an object **state** is …
    - simple (atomic)
    - complex
Logical Schema Design

• Object-Oriented Model

(i₁, atom, ’464197351’)  
(i₂, atom, ’SMITH’)  
(i₃, atom, ’JOHN’)  
(i₄, atom, ’08/15/1952’)  
(i₅, atom, ’GEORGE’)  
(i₆, atom, ’191’)  
(i₇, atom, ’FREDERICTON’)  
(i₈, atom, ’NB’)  
(i₉, atom, ’E3A5A3’)  
(i₁₀, atom, ’CANADA’)  
(i₁₁, atom, ’5064555555’)  
(i₁₂, atom, ’5064448888’)  
(i₁₃, atom, ’5069991234’)  
(i₁₄, tuple, <STREET:i₅, NUMBER:i₆, CITY:i₇, PROVINCE:i₈, ZIP:i₉, COUNTRY:i₁₀>)  
(i₁₅, set, <i₁₁, i₁₂, i₁₃>)  
(i₁₆, tuple, <SIN:i₁, SURNAME:i₂, NAME:i₃, DoB:i₄, ADDRESS:i₁₄, TELEPHONES:i₁₅>)
Logical Schema Design

• Object-Oriented Model

Object

Tuple

Collection

notation (diagram)

SIN: 464 197 351
Name: John  Surname: Smith
Date of Birth: 15/Aug/1952
Address: 191 George Str., Fredericton, NB, E3A5A3, Canada
Tel: {5064555555, 5064448888, 5069991234}
Logical Schema Design

• Object-Oriented Model
  – the object behavior ...
    • methods (operations) that can be executed to
      – create/destroy an object
      – update an object state
      – retrieve object state
      – compute new values based on object state
    • the names and parameters of methods
      – define the object interface
Logical Schema Design

• Object-Oriented Model …
  – also supports …
  • object classes
  • associations between object classes
  • class hierarchies and inheritance

  – Several prototype systems: e.g., O₂, Informix, etc.
Logical Schema Design

• Object-Oriented Model ...

Object Class Owners:
State
  tuple (  
    sin          string  
    surname      string  
    name         string  
    dob          date    
    address      tuple (  
      street      string  
      number      integer 
      city        string  
      province    string  
      zip         string  
      country     string  
    )  
    telephones   set (    
      telephone   integer 
    )  
  )  

Behavior
newOwner      boolean  
remOwner       boolean  
modOwnAddr     boolean  
umOfOwn        integer  
ageOwner       integer  
telOwner       set(integer)

Example object class: Owner.
Logical Schema Design

- Object-Oriented Model ...

```
Object Class Parcels:
    State
tuple (  
id             string
    use           decimal
    built_factor  string
    address       string
)

Behavior
    newParcel     boolean
    remParcel     boolean
    useParcel     string
```

Example object class: Parcel.
Logical Schema Design

• Object-Oriented Model ...

Object Class Owners:
  State
  tuple {
    sin string
    ... 
    owns set(Parcels) inverse Parcels:ownedBy 
  }

Behavior
  ...

Object Class Parcels:
  State
  tuple {
    id string
    ... 
    ownedBy set(Owners) inverse Owners:own 
  }

Behavior
  ...

bi-directional references
Logical Schema Design

- Object-Oriented Model …

Object Class Owners:
State
tuple (
    sin string
... owns set(
tuple ( 
    parcel Parcels
    percentage integer
    purchase_date date ) )

Behavior
...

Object Class Parcels:
State
tuple ( 
    id string
... Behavior
...

Integration of the relationship type attributes in the reference.
Logical Schema Design

- **Object-Oriented DBMSs...**
  - Standard ...
    - **ODMG**
      - ver. 1.0 (1993), ver. 2.0 (1997), ver. 3.0 (2000)
    - *consists of ...*
      - the object model
      - the object definition language (ODL)
      - the object query language (OQL)
Logical Schema Design

- Mapping EER/UML to OO Model
  - Rule 1: mapping entities into classes
  - Rule 2: mapping relationships
  - Rule 3: Embedding methods into classes
  - Rule 4: mapping super-subclass relationships

*not examined here*
Logical Schema Design

- **Object-Oriented DBMSs**…
  - Example OQL queries…

```sql
SELECT struct<d.sin, d.surname, d.name>
FROM d in all_owners
WHERE d.DoB < 01/01/1960;

all_owners;

464197351.surname;
464197351.address.city;
464197351.telephones;
464197351.titles.owns.parcel;
```
Logical Schema Design

• **Advantages** of the O-O Model…

1. Efficiency

   • encapsulation of methods in the objects
   • tends to group all data concerning an object
   • implementation of relationships through direct links
Logical Schema Design

• **Advantages** of the O-O Model…

  2. Data Semantics

    • clear distinction between: classification, generalization and aggregation

  3. Model Extension

    • possible by the use of classes in a manner identical with ADT in programming

    • user defined data types
Logical Schema Design

- **Advantages** of the O-O Model…

  4. Object Identity
  
  - unique identifications for all objects
  - created by the system
  - independent of addresses and data values
  - they can survive updating and data reorganization
Logical Schema Design

- **Advantages** of the O-O Model...

5. **Program Interface**
   - similarities between OOPS - OODBMS
   - smoother program interface
   - the gap between the software and the program interface is small
Chapter 7

Advanced Topics in Database Systems

Emmanuel Stefanakis
http://www2.unb.ca/~estef/