IBM Software Group

Mastering Object-Oriented Analysis and Design with UML

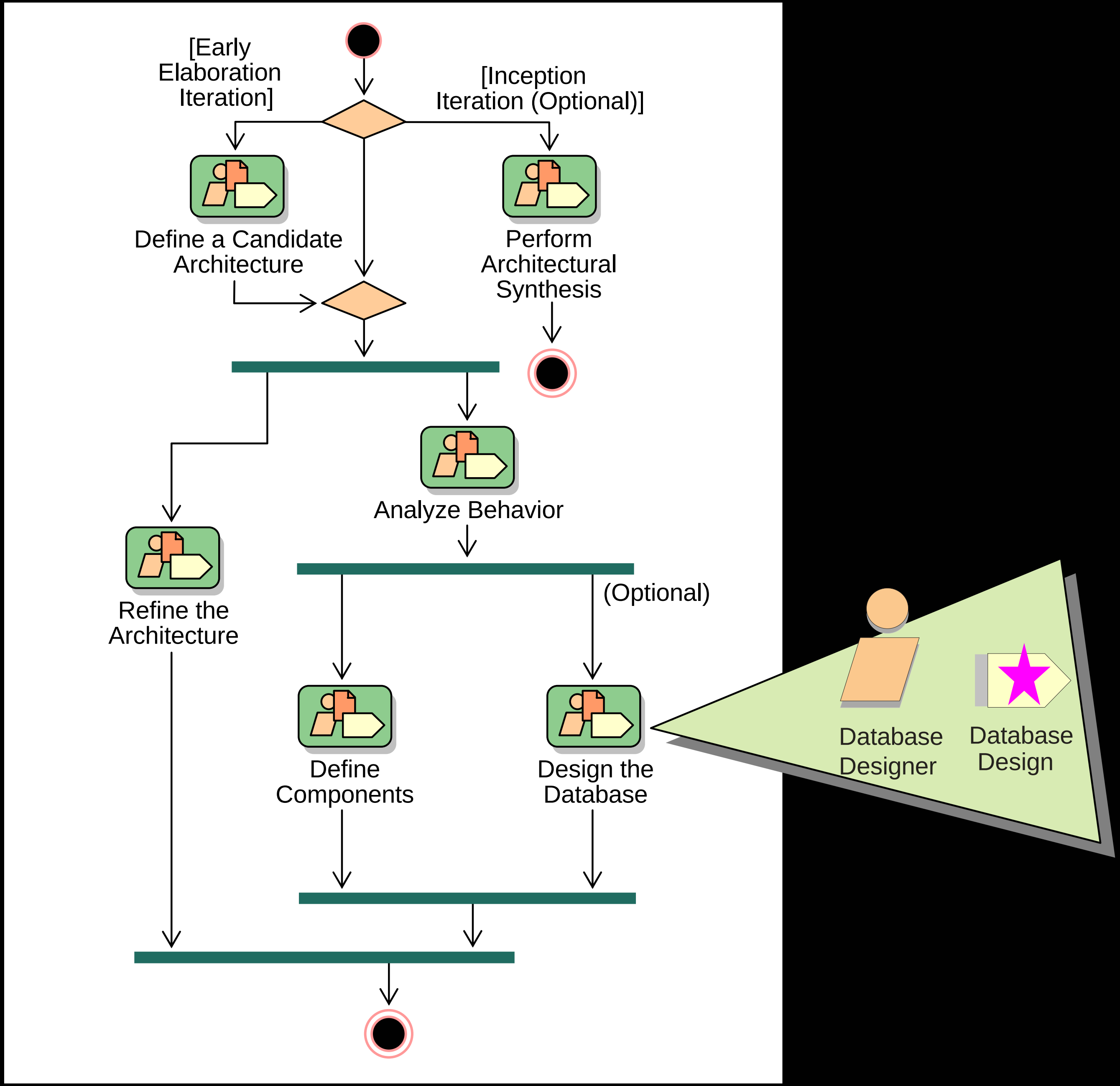
Module 12: Database Design

Rational. software

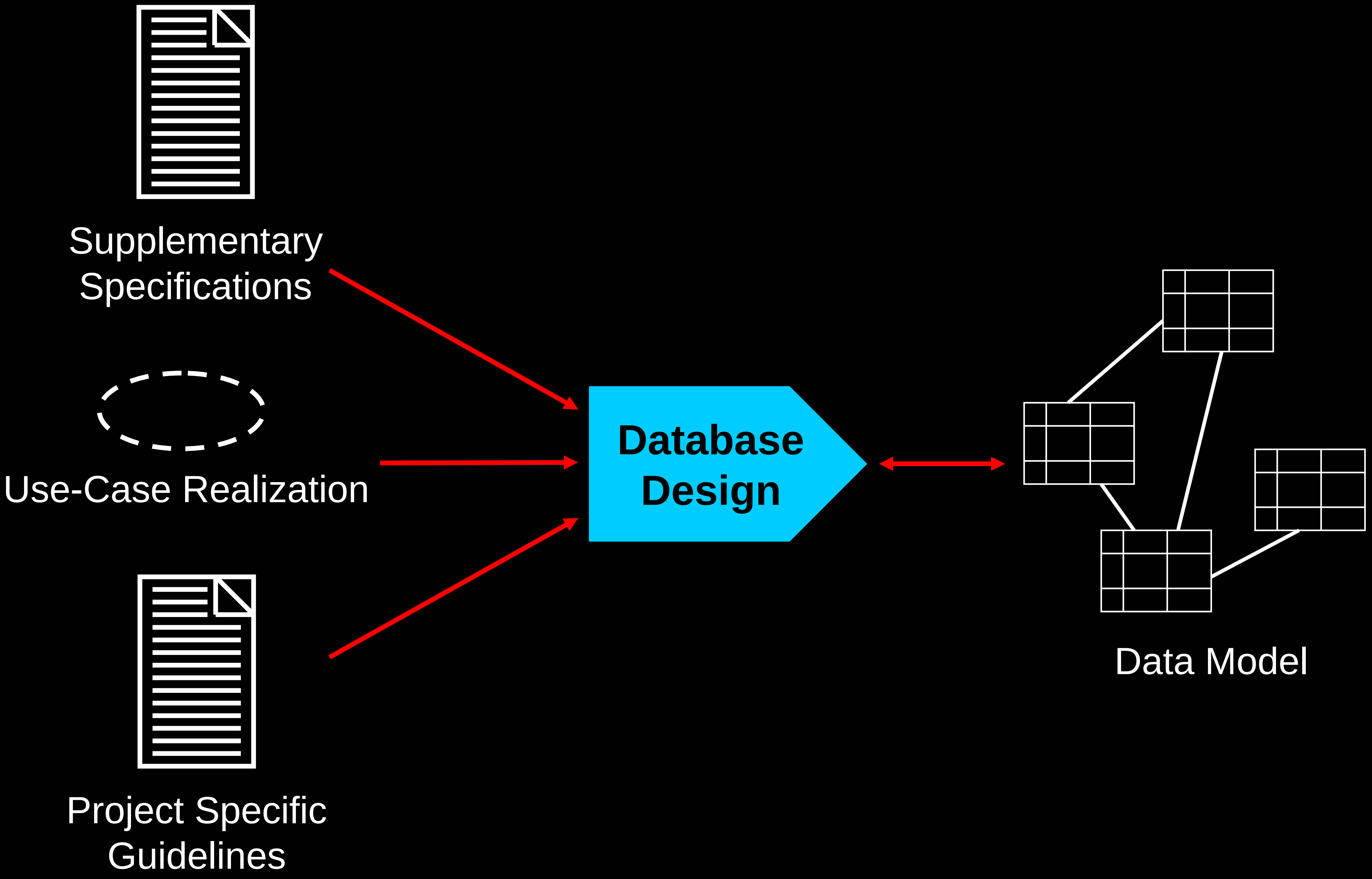
Objectives: Database Design

- Define the purpose of Database Design and where in the lifecycle it is performed
- Explain how persistent classes map to the data model
- Learn how to distribute class behavior to the database

Database Design in Context

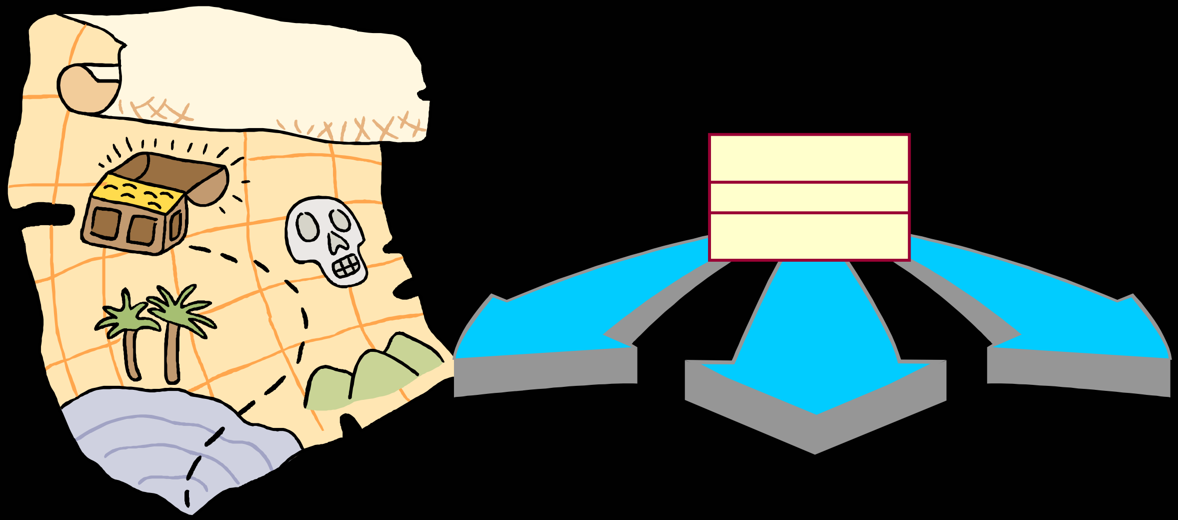

[Early Elaboration Iteration]
[Inception Iteration (Optional)]
Define a Candidate Architecture
Perform Architectural Synthesis
Analyze Behavior
Refine the Architecture
(Optional)
Define Components
Design the Database
Database Designer
Database Design

Database Design Overview



Database Design Steps

- Map persistent design classes to the data model
- Distribute class behavior to the database



Database Design Steps

★ ◆ Map persistent design classes to the data model

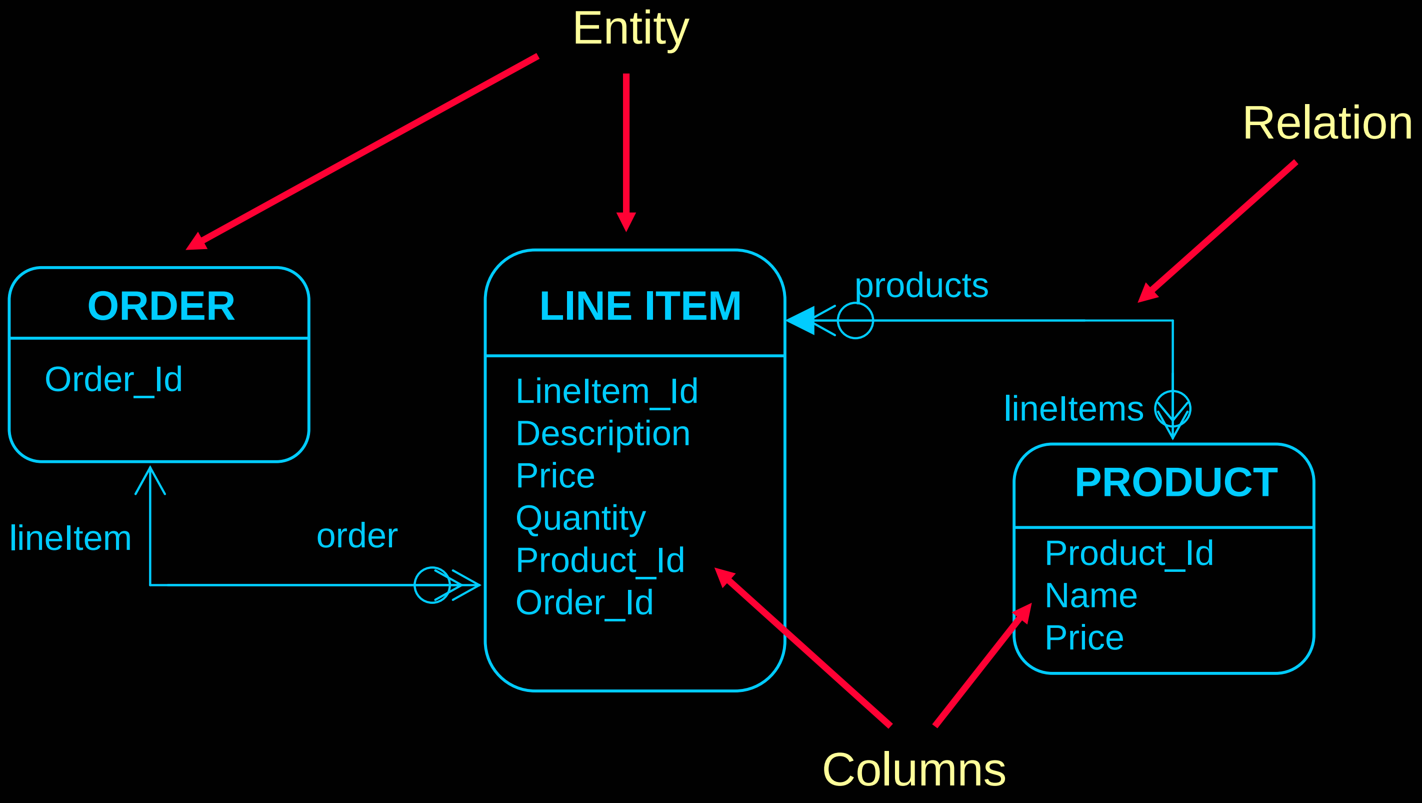
◆ Distribute class behavior to the database

Relational Databases and Object Orientation

- RDBMS and Object Orientation are not entirely compatible
  - RDBMS
    - Focus is on data
    - Better suited for ad-hoc relationships and reporting application
    - Expose data (column values)
  - Object Oriented system
    - Focus is on behavior
    - Better suited to handle state-specific behavior where data is secondary
    - Hide data (encapsulation)

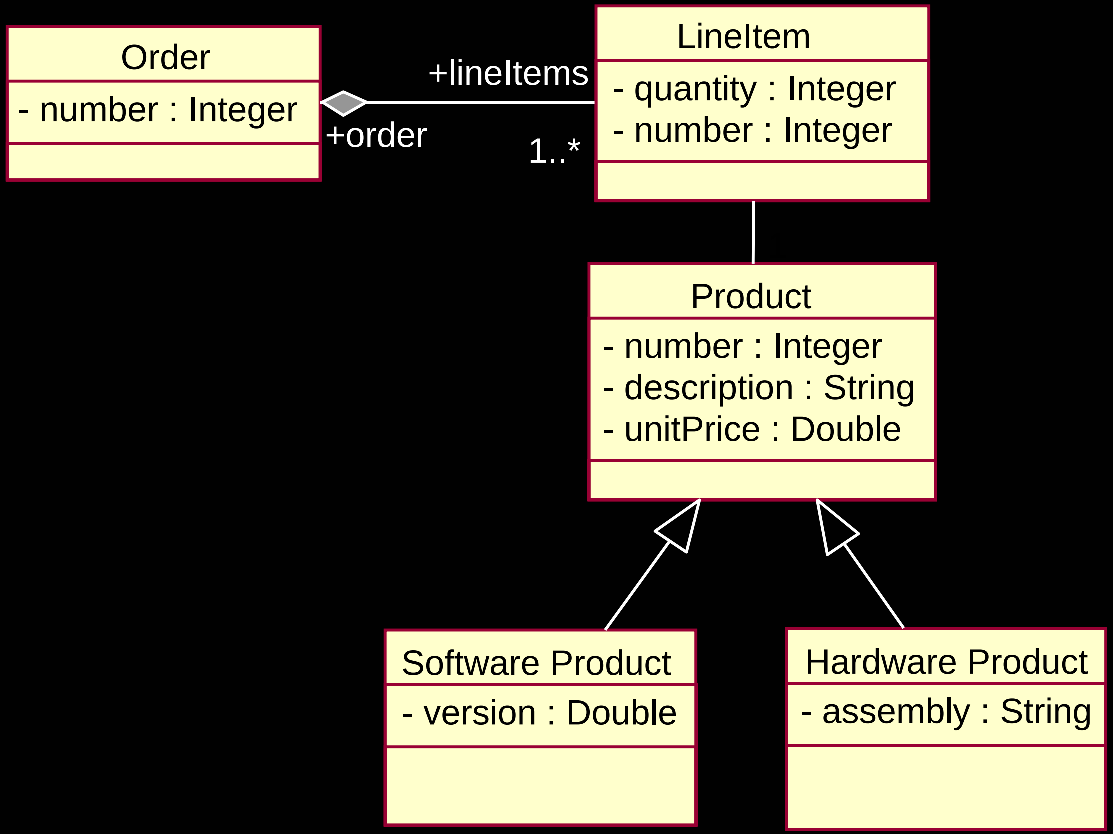
The Relational Data Model

- Relational model is composed of
  - Entities
  - Relations



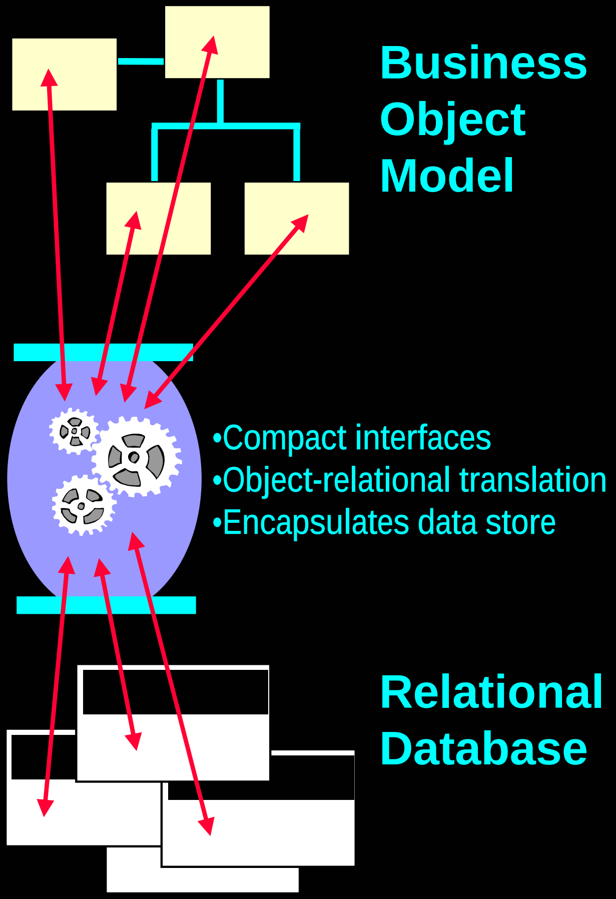
The Object Model

- The Object Model is composed of
  - Classes (attributes)
  - Associations



Persistence Frameworks

- The challenge:
  - Changes should not break the model
- The solution: An object-relational framework that
  - Encapsulates the physical data store
  - Provides object translation services
- The importance of the framework
  - 30% of development time is spent in accessing an RDBMS
  - Maintenance can be 60% of total cost



Object-Relational Framework: Characteristics

- Performance
  - Decomposing objects to data
  - Composing objects from data
- Minimize design compromises
  - Limit changes to object and relational models
- Extensibility
  - 15%-35% of the framework needs to be designed as an extensible framework

Object-Relational Frameworks: Characteristics (cont.)

- Documentation of the API
- Support for common object-relational mappings
- Persistence interfaces
  - Examples are save, delete, and find

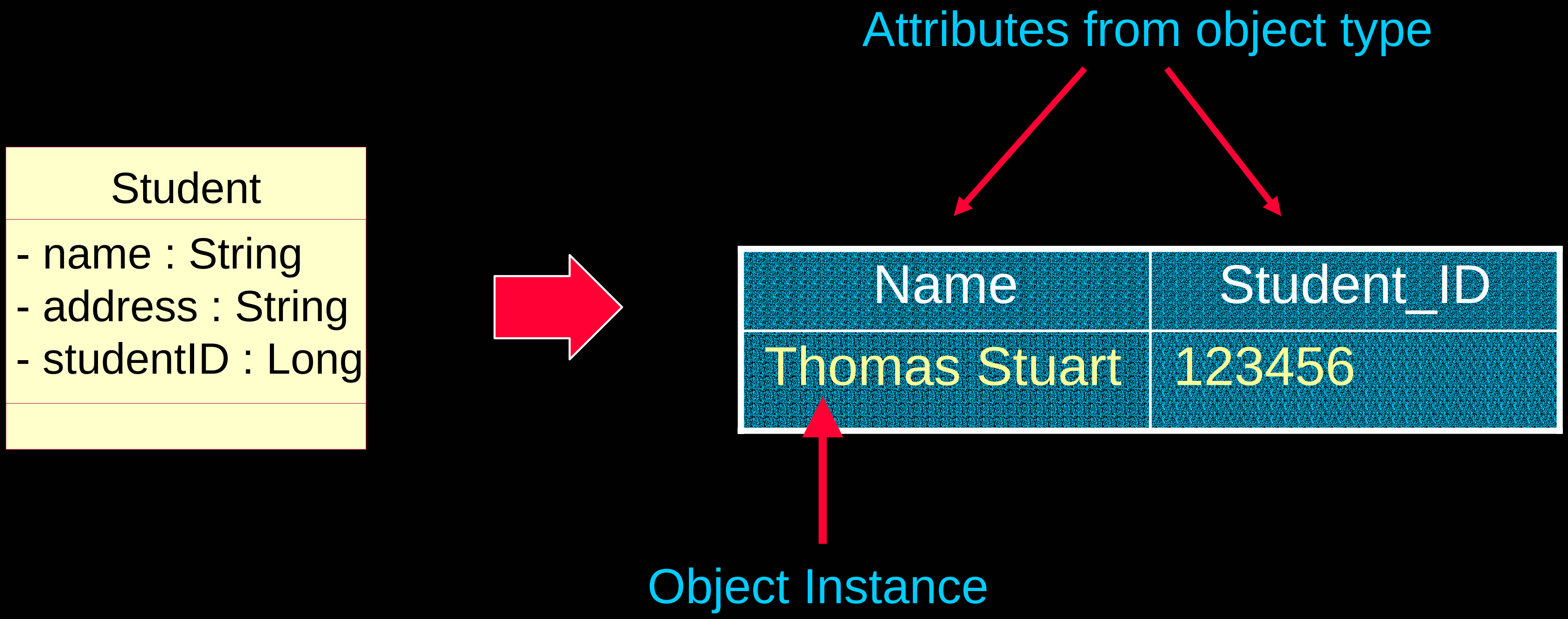
Common Object-Relational Services

- Patterns are beginning to emerge for object-relational applications
  - CORBA Services specification
    - Persistence
    - Query
    - Transactions
    - Concurrency
    - Relationships

Refer to the appropriate CORBA specifications for further details.

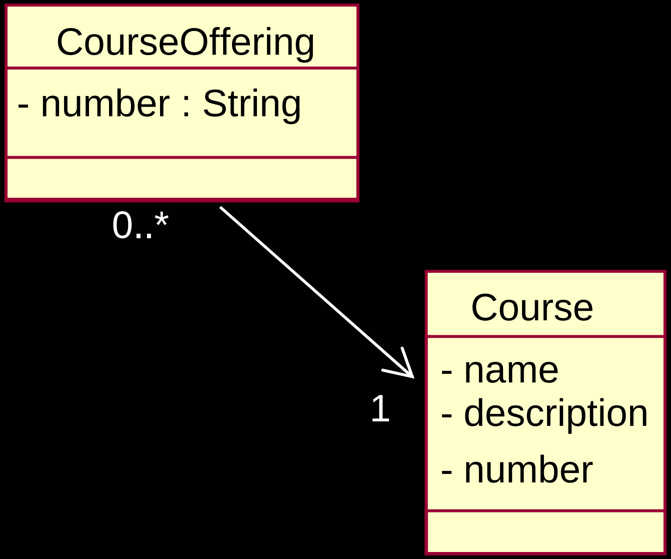
Mapping Persistent Classes to Tables

- In a relational database
  - Every row is regarded as an object
  - A column in a table is equivalent to a persistent attribute of a class



Mapping Associations Between Persistent Objects

- Associations between two persistent objects are realized as foreign keys to the associated objects.
  - A foreign key is a column in one table that contains the primary key value of associated object



Course Offering table

Number	Course_ID
678	456789

Foreign Key

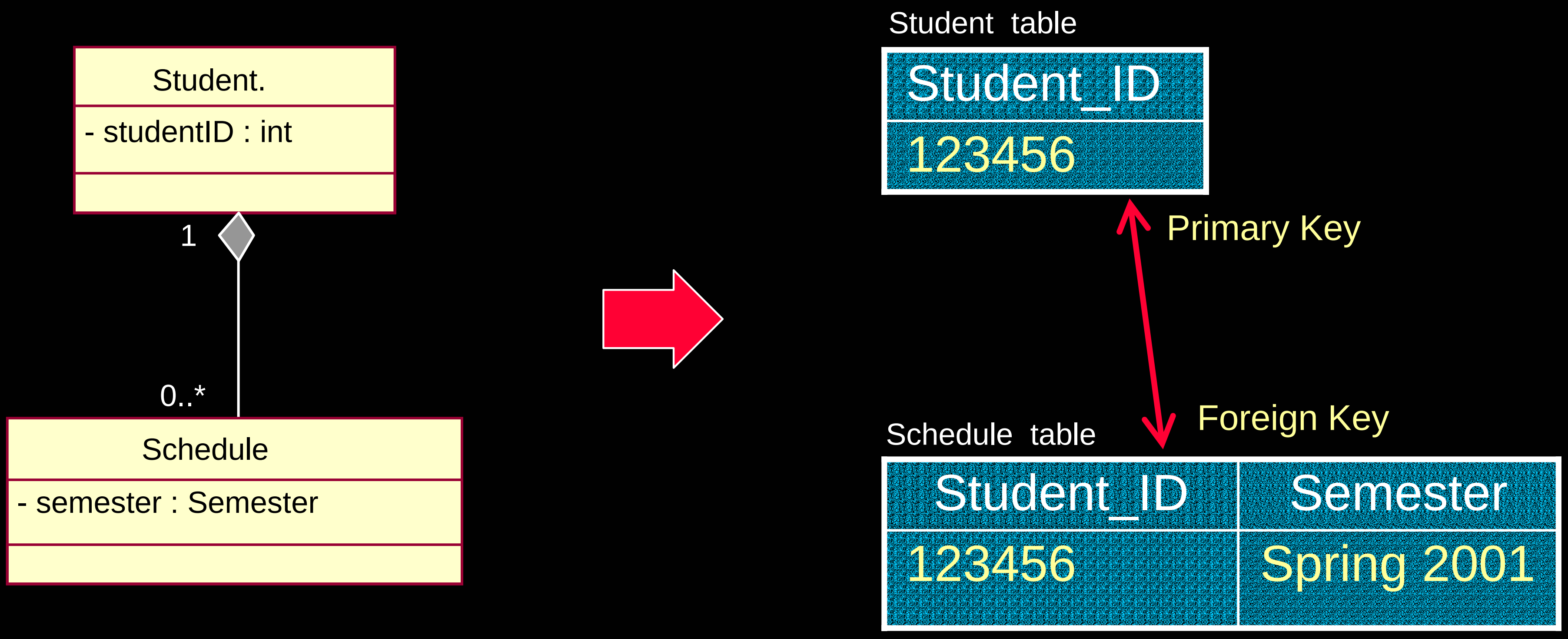
Primary Key

Course table

Name	Description	Number
Math 101	Algebra	456789

Mapping Aggregation to the Data Model

- Aggregation is also modeled using foreign key relationships
  - The use of composition implements a cascading delete constraint

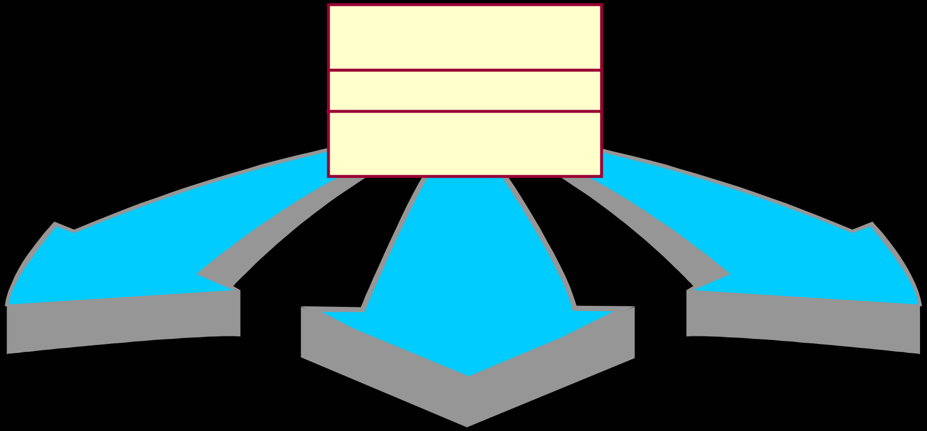


Modeling Inheritance in the Data Model

- A Data Model does not support modeling inheritance in a direct way
- Two options:
  - Use separate tables (normalized data)
  - Duplicate all inherited associations and attributes (de-normalized data)

Database Design Steps

- Map persistent design classes to the data model
- ★ Distribute class behavior to the database



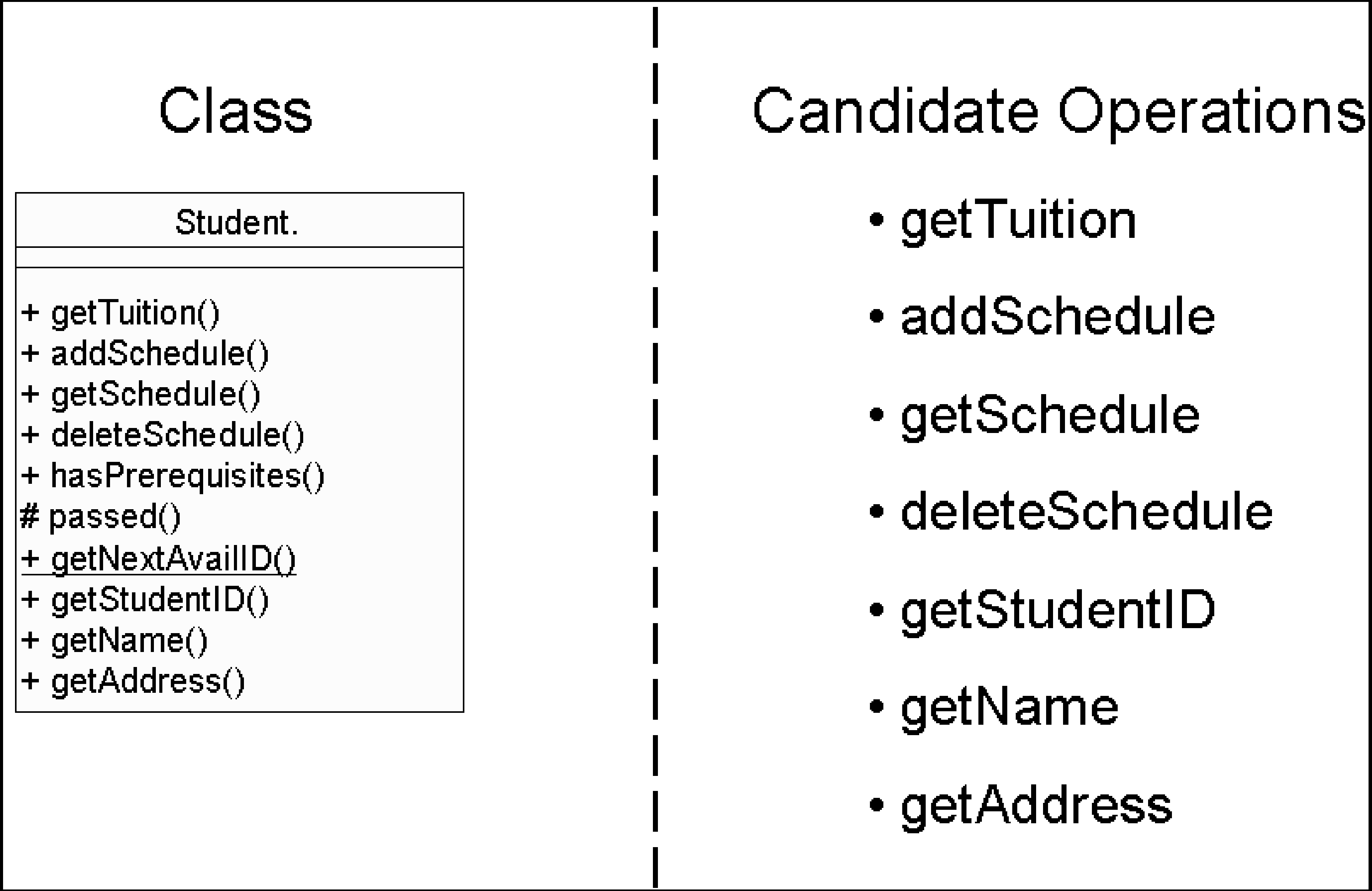
What Are Stored Procedures?

- A stored procedure is executable code that runs under the RDBMS
- Two types of stored procedures:
  - Procedures: Executed explicitly by an application
  - Triggers: Invoked implicitly when some database event occurs

Map Class Behavior to Stored Procedures

- Determine if any operations can be implemented as a stored procedure
- Candidates:
  - Operations that deal with persistent data
  - Operations in which a query is involved in a computation
  - Operations that need to access the database to validate data

Example: Map Class Behavior to Stored Procedures



Checkpoints: Database Design

- Have all persistent classes been mapped to database structures?
- Have stored procedures and triggers been defined?
- Does the persistence mechanism use stored procedures and database triggers consistently?

Review: Database Design

- What is the purpose of the Database Design?
- What comprises a relational data model?
- What are the components of an object model?
- When mapping persistent classes to tables, what is every row in a table regarded as? What is every column equivalent to?
- What are stored procedures?