DATABASE DESIGN & DATA MODELING

CS660 Week 2: From Concepts to Implementation

Dr. John Conklin

UNIT 2 OVERVIEW

1. **Database Design & Problem Solving**
   Systematic approach to designing effective databases
2. **Entity Relationship (ER) Modeling**
   Visual representation of data structures and relationships
3. **Normalization**
   Organizing data to eliminate redundancy and anomalies
4. **Referential Integrity**
   Maintaining consistency across related tables
5. **Business Rules**
   Translating business requirements into database constraints

DATABASE DESIGN & PROBLEM SOLVING

Design Philosophy

Database design is a systematic process of defining the structure, storage, and retrieval methods for data. Good design requires understanding the problem domain, identifying entities and relationships, and creating a schema that is efficient, maintainable, and aligned with business needs.

Understand Requirements

Gather comprehensive business needs

Identify Entities

Recognize real-world objects to model

Define Relationships

Determine how entities connect

Apply Standards

Follow normalization and best practices

THE DESIGN PROCESS

1	**Requirements Gathering**	Interview stakeholders and document needs
2	**Conceptual Design**	Create high-level ER diagram
3	**Logical Design**	Convert to relational schema with normalization
4	**Physical Design**	Optimize for specific DBMS
5	**Implementation**	Create actual database structures
6	**Testing & Refinement**	Validate and optimize performance

ENTITY RELATIONSHIP MODELING

What is ER Modeling?

Entity-Relationship modeling is a visual technique for database design that represents entities (objects), their attributes (properties), and the relationships between entities. ER diagrams provide a clear, graphical view of the database structure before implementation.

	**Entity**	Real-world object or concept	*Example: Customer, Product, Order*
	**Attribute**	Property of an entity	*Example: Name, Price, Date*
	**Relationship**	Connection between entities	*Example: Customer places Order*
	Cardinality	Number of instances	*Example: One-to-Many, Many-to-Many*

ER DIAGRAM NOTATION

Common Notation Styles

Chen Notation (Classic)

- Entities: Rectangles
- Relationships: Diamonds
- Attributes: Ovals
- Lines: Connect elements

Crow's Foot (Modern)

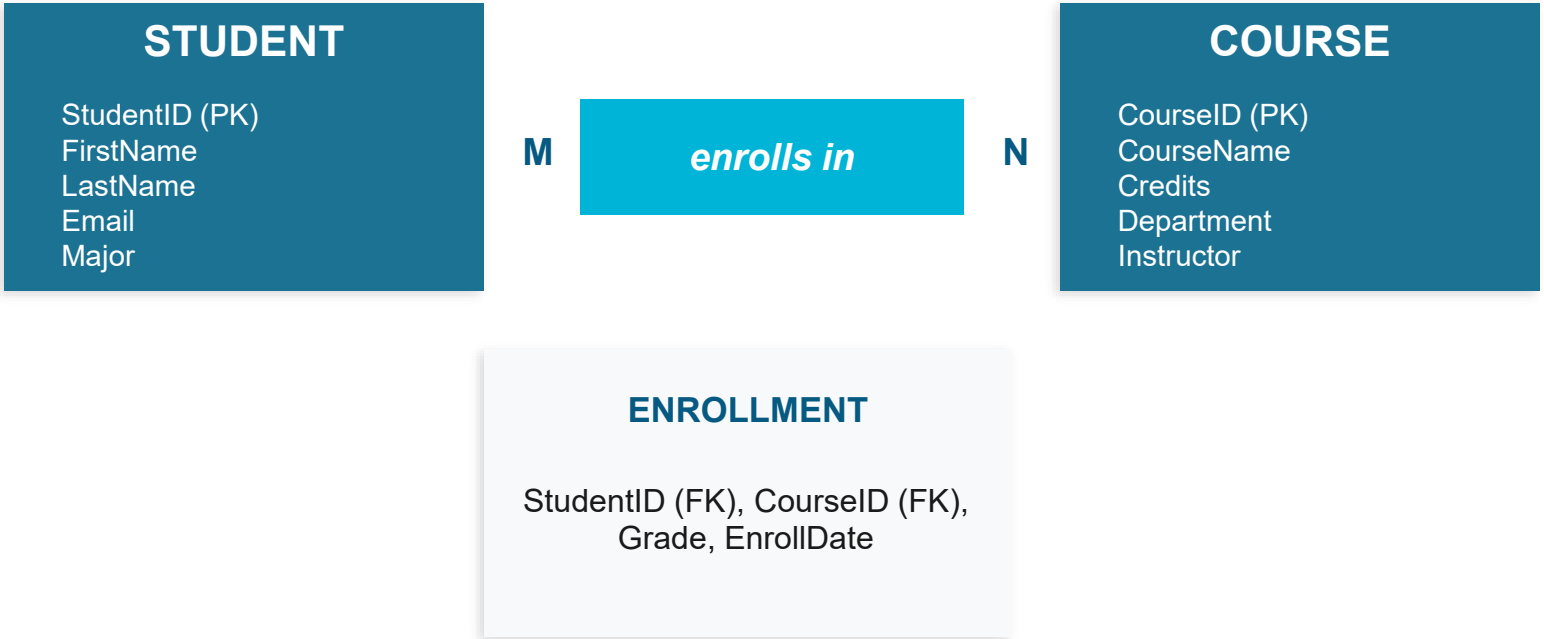
- Entities: Rectangles
- Relationships: Lines with symbols
- Attributes: Inside entity boxes
- Cardinality: Crow's foot, dash, circle

Cardinality Indicators

`1`	`0..1`	`0..*`	`1..*`	`M:N`
One (Mandatory)	Zero or One (Optional)	Zero or Many	One or Many	Many-to-Many

ER MODELING EXAMPLE

University Student Enrollment System



NORMALIZATION

Organizing data to eliminate redundancy and anomalies

Problems Without Normalization

- Update Anomalies

Change data in multiple places

- Insert Anomalies

Cannot add data without unrelated info

- Delete Anomalies

Lose data when deleting records

Benefits of Normalization

- Data Consistency

Single source of truth for each fact

- Reduced Redundancy

Store each piece of data once

- Easier Maintenance

Fewer places to update

Normalization Process

Start with unnormalized data → Apply normal forms progressively → Achieve optimal structure

1NF	Atomic values, no repeating groups
2NF	Remove partial dependencies
3NF	Remove transitive dependencies
BCNF	Every determinant is candidate key

FIRST NORMAL FORM (1NF)

Rules for 1NF

- Each column contains atomic (indivisible) values
- No repeating groups or arrays within columns
- Each row must be unique (has a primary key)

❌ Violates 1NF

CUSTOMERS: ID | Name | PhoneNumbers

1 | John Smith | 555-1234, 555-5678, 555-9012

✓ Meets 1NF

CUSTOMERS: ID | Name

1 | John Smith

PHONES: CustomerID | PhoneNumber

1 | 555-1234
1 | 555-5678
1 | 555-9012

SECOND NORMAL FORM (2NF)

Rules for 2NF

- Must be in 1NF
- Remove partial dependencies on composite keys
- Every non-key column must depend on the ENTIRE primary key

❌ Violates 2NF

`ORDER_DETAILS: (OrderID, ProductID) PK | ProductName | Quantity | Price`

Problem: ProductName depends only on ProductID, not on (OrderID, ProductID)

✓ Meets 2NF

`PRODUCTS: ProductID PK | ProductName | Price`

`ORDER_DETAILS: (OrderID, ProductID) PK | Quantity`

THIRD NORMAL FORM (3NF)

Rules for 3NF

- Must be in 2NF
- Remove transitive dependencies
- Non-key columns must depend ONLY on primary key (not on other non-key columns)

❌ Violates 3NF

`EMPLOYEES: EmployeeID PK | Name | DepartmentID | DepartmentName | DeptLocation`

Problem: DepartmentName depends on DepartmentID (transitive dependency)

✓ Meets 3NF

`DEPARTMENTS: DepartmentID PK | DepartmentName | Location`

`EMPLOYEES: EmployeeID PK | Name | DepartmentID FK`

BOYCE-CODD NORMAL FORM

Rules for BCNF

- Stricter version of 3NF
- Every determinant must be a candidate key
- Eliminates anomalies not caught by 3NF

When to Use BCNF

- Complex schemas with overlapping candidate keys
- When 3NF still has anomalies
- Theoretical completeness required

Practical Considerations

- 3NF sufficient for most applications
- BCNF may require more tables
- More joins can impact performance

Normalization Summary

1NF → 2NF → 3NF → BCNF : Each level eliminates specific types of anomalies

REFERENTIAL INTEGRITY

What is Referential Integrity?

Referential integrity ensures that relationships between tables remain consistent. Foreign key values must match existing primary key values in the referenced table, or be NULL if allowed. This prevents orphaned records and maintains data consistency.

Foreign Keys

References to primary keys in other tables

Consistency

No orphaned or invalid references

Constraints

Database enforces rules automatically

Cascade Rules

Define behavior on delete/update

CASCADE RULES

What happens when parent records are deleted or updated?

CASCADE	ON DELETE: Delete child records automatically ON UPDATE: Update foreign key values automatically	*Use: Parent-child with strong dependency*
RESTRICT	ON DELETE: Prevent deletion if children exist ON UPDATE: Prevent update if children exist	*Use: Preserve data integrity*
SET NULL	ON DELETE: Set foreign key to NULL ON UPDATE: Set foreign key to NULL	*Use: Retain child, remove relationship*
NO ACTION	ON DELETE: Raise error (similar to RESTRICT) ON UPDATE: Raise error (similar to RESTRICT)	*Use: Default behavior, explicit control*

BUSINESS RULES

Translating Requirements into Constraints

Business rules are constraints that enforce business logic and policies at the database level. They ensure data validity, maintain business processes, and prevent invalid states. Examples include age restrictions, credit limits, and inventory levels.

Types of Business Rules

Entity Constraints

Rules about single entities

Example: Age >= 18

Domain Constraints

Valid values for attributes

Example: Grade IN ('A','B','C','D','F')

Relationship Constraints

Rules between entities

Example: Manager must be Employee

Business Logic

Complex calculations

Example: OrderTotal = SUM(LineItems)

IMPLEMENTING BUSINESS RULES

CHECK Constraints

```
ALTER TABLE Students
ADD CONSTRAINT chk_age
CHECK (Age >= 18);
```

✓ Simple, declarative, enforced by DBMS

⚠ Limited to single-row logic

Triggers

```
CREATE TRIGGER validate_order
BEFORE INSERT ON Orders
FOR EACH ROW
BEGIN
  -- Complex validation
END;
```

✓ Complex logic, multiple tables, auditing

⚠ Performance overhead, harder to debug

Stored Procedures

```
CREATE PROCEDURE PlaceOrder
  @CustomerID INT,
  @ProductID INT
AS
BEGIN
  -- Business logic
END;
```

✓ Centralized logic, reusable, secure

⚠ Application must call procedures

KEY TAKEAWAYS

- Database design follows a systematic process from requirements to implementation
- ER modeling visually captures entities, attributes, and relationships
- Normalization eliminates redundancy through 1NF, 2NF, 3NF, and BCNF
- Referential integrity maintains consistency with foreign keys and cascade rules
- Business rules enforce logic through constraints, triggers, and procedures

Master these skills to design robust, efficient databases! 🎯

INDIVIDUAL PROJECT

Individual Project

The retail store has provided you with a list of the business rules they use to conduct daily operations. The company has also provided you with the subjects of interest to them, namely, those things that the company wants to track in the process of taking orders from their customers.

What can be done to effectively document the relational database system solution that you are proposing to the retail store?

The project deliverables are as follows:

- Subjects of Interest (proposed entities)
  - Customers
  - Orders
  - Products
  - Add a minimum of 3 of your own subjects of interest based on your retail store.
- Business Rules
  - Each product is assigned to a maximum of 1 category.
  - A person who has placed at least 1 order is a customer.
  - All products have a minimum reorder level.
  - Add a minimum of 3 of your own business rules based on your retail store.
- Entity–Relationship Model (4–5 pages)
  - Include a list of the business rules that will be enforced by the proposed database system.
  - Include a list or table of the entities, attributes (including data types), relationships, and cardinality constraints.
  - Include an entity–relationship (E–R) diagram that uses crow's foot notation and graphically depicts the entities, attributes (including data types), and relationships (including degree and cardinality constraints).
  - Verify that the data design that is depicted in the E–R diagram adheres to a minimum of third normal form (3NF), and if necessary, provide documentation and justification for the use of a higher level of normal form.
- Provide your analysis as to how this part of the project fulfills the mission and 1 or more goals of the case study organization.
- **Note:** Use Microsoft Visio or equivalent to create the E–R diagram. You will embed the diagram in the Word document and also provide it as an attachment.
- All sources should be cited both in-text and in References using APA format.
- Name the document "yourname_CS660_IP2.doc."

Contact Information

Email:	Jconklin@coloradotech.edu
Phone:	602.796.5972
Website:	http://drjconklin.com
Office Hours:	Wednesdays: 6:00 PM – 7:00 PM (CST)
	Saturdays: 11:00 AM – 12:00 PM (CST)
Live Chats:	Wednesdays: 6:00 PM – 7:00 PM (CST)