**University Course Management – Producing BCNF-Compliant ERD with Git Version Control**

**Background:**

A mid-sized university is moving from manual course enrollment to a digital platform. The IT department must design a robust and normalized database model that supports:

* Multi-course enrollment
* Instructor assignments
* Optional course reviews
* Compliance with normalization (BCNF)
* Collaboration via Git for versioned ERD evolution

**Phase 1: Requirements Gathering via Interview Drill**

The team conducted stakeholder interviews with:

* Registrar (on student-course lifecycle)
* Faculty coordinator (on instructor allocations)
* IT admin (on integration and review needs)

**Key Findings:**

* A student may enroll in multiple courses.
* A course is taught by exactly one instructor.
* Students optionally review courses post-completion.

**Phase 2: Entity Identification and Relationship Design**

Identified Entities:

* Student(student\_id, name, email, department)
* Instructor(instructor\_id, name, specialization)
* Course(course\_id, title, credits)
* Enrollment(enrollment\_id, student\_id, course\_id, enrollment\_date)
* Review(student\_id, course\_id, rating, comment)

Cardinalities:

* **Student—Course** → M:M (via Enrollment)
* **Course—Instructor** → M:1
* **Student—Review—Course** → 1:1 per course per student (optional)

**Phase 3: Atomicity & Attribute Cleanup**

Initially:

* name, address, and contact\_info were non-atomic.

Revised:

* name → first\_name, last\_name
* address → street, city, state, zip
* contact\_info split into email, phone

**Phase 4: Normalization to BCNF**

**Raw Table:**

| **student\_name** | **course\_name** | **instructor\_name** | **rating** |
| --- | --- | --- | --- |

**1NF: Atomicity enforced**

No repeating or nested values.

**2NF: Eliminate partial dependency**

* Moved student and instructor data to separate tables.

**3NF: Remove transitive dependency**

* Instructor info moved out of course.

**BCNF: Every determinant is a candidate key**

* Relationship tables (Enrollment, Review) have composite primary keys or surrogate keys with all FDs resolved.

**Final Tables:**

* Student(student\_id, first\_name, last\_name, email, dept)
* Instructor(instructor\_id, name)
* Course(course\_id, title, credits, instructor\_id)
* Enrollment(enrollment\_id, student\_id, course\_id)
* Review(student\_id, course\_id, rating, comment)

**Phase 5: Crow-Foot ERD Drafting**

Created using [dbdiagram.io](https://dbdiagram.io) with DBML syntax:

dbml

Table course {

course\_id int [pk]

title varchar

credits int

instructor\_id int [ref: > instructor.instructor\_id]

}

Exported diagram in PNG format for documentation.

**Phase 6: Naming Standards Enforcement**

Standards used:

* Tables: singular (student, course)
* Columns: snake\_case
* FKs: end with \_id
* All lowercase, no special characters

**Phase 7: Git-Based ERD Version Control**

1. **Initialize Git repository**

bash

git init

1. **Track files**

bash

git add erd.dbml erd.png README.md

git commit -m "Initial normalized ERD (BCNF)"

1. **Push to GitHub**

bash

git remote add origin https://github.com/university/db-erd.git

git push -u origin main

1. **Subsequent Update Example:**  
   New requirement – Add semester field to Enrollment table:

bash

git checkout -b add-semester-field

# Edit DBML

git commit -am "Added semester to enrollment"

git push origin add-semester-field