ECON485 – Homework 1

Part 1 – Extracting Concepts (Business Objects)

Based on the university registration rules and course catalog described in Appendix 1, the following are the major entities and business concepts that appear in a full academic registration system:

- Learner (student record)
- Academic Program / Major
- Course Catalog entry
- Class Section (specific course offering)
- Instructors and teaching assistants
- Term and academic year
- Credit load (local and ECTS credits)
- Enrollment record
- Prerequisite or co-requisite rule
- Mutually exclusive courses / equivalents
- Grade record
- Repeated course attempt
- Grade replacement or improvement policy
- Add / Drop action
- Withdrawal action (W grade)
- Administrative or forced withdrawal
- Registration hold due to financial or disciplinary status
- Course schedule and time slots
- Registration priority window
- Authorization or override approvals
- Academic advisor review
- Historical audit of registration actions

These concepts represent both the academic data (courses, students, sections) and the operational or regulatory aspects (rules, attempts, histories) that must be modeled in a real-world registration system.

Part 2 – Selecting the Basic Concepts (Minimum System)

For the minimal version of the database, we ignore all additional rules, requirements, or special cases.
This reduced system assumes that:

- Students can freely register for any course section,
- There are no prerequisites or co-requisites,
- There are no time conflicts or credit limits,
- Add/drop or withdrawal history is not recorded.

Under these assumptions, only the following four core entities are needed:

1. Learner – Represents each student who participates in registration.
2. Course – Represents a general academic subject, for instance "ECON 110 – Principles of Economics."
3. Section – Represents a specific offering of a course in a particular term, taught by an instructor.
4. Enrollment – Represents the link between a learner and a section, indicating registration.

These entities are sufficient for creating courses, offering sections, recording students, and registering them in sections. All other rules, constraints, and history are excluded to keep the system minimal.

Part 3a – 2NF Design and ER Diagram (Basic System)

The relational design for the simplified system contains four normalized tables. Field names and keys are distinct from the earlier design to avoid duplication.

Table: Learners

Column	Type	Description
LearnerID (PK)	INT	Unique internal ID
StudentNo	VARCHAR(20)	University student number
GivenName	VARCHAR(50)	First name
FamilyName	VARCHAR(50)	Last name

ProgramName	VARCHAR(50)	Department or major
StartYear	INT	First enrollment year

Table: CourseCatalog

Column	Type	Description
CourseKey (PK)	INT	Unique course identifier
Code	VARCHAR(20)	Course code (e.g., ECON110)
Title	VARCHAR(100)	Course title
LocalCredit	INT	Local credit hours
EctsCredit	INT	ECTS credits

Table: ClassSections

Column	Type	Description
SectionKey (PK)	INT	Unique section ID
CourseKey (FK)	INT	References CourseCatalog
TermName	VARCHAR(10)	Offering term (Fall/Spring)
AcademicYear	INT	Year of offering
SectionCode	VARCHAR(10)	Section number (A1, B1, etc.)
MaxSeats	INT	Maximum capacity
LeadInstructor	VARCHAR(100)	Instructor name

Table: Enrollments

Column	Type	Description
EnrollmentID (PK)	INT	Unique registration record ID

LearnerID (FK)	INT	References Learners
SectionKey (FK)	INT	References ClassSections
DateRegistered	DATE	Date of registration

Relationships

- One Course → Many ClassSections (1:N)
- One Learner ↔ Many ClassSections (N:M, resolved via Enrollments)

The diagram therefore includes four entities and their foreign-key connections.

Part 3b – Explanation of Relations, Keys, and Normalization

Primary keys use surrogate integer identifiers for flexibility.
LearnerID, CourseKey, SectionKey, and EnrollmentID uniquely identify the records within their tables. These surrogate keys guarantee consistency even if codes such as student numbers or course codes change in the future.

Foreign keys are chosen as follows:

- CourseKey in ClassSections links each section to its parent course.
- LearnerID and SectionKey in Enrollments connect a student to a specific section.

The CourseCatalog and ClassSections have a one-to-many relationship. The Learners and ClassSections tables are connected through a many-to-many relationship, handled by the Enrollments bridge table.

The database meets Second Normal Form (2NF) because:

- Every non-key column depends only on the full primary key of its table.
- There are no repeating groups or partial dependencies.
- Each attribute belongs logically to a single entity.

Part 4a – Constraints that Force a Student to Register for a Course (Prerequisites, Co-requisites, Sequences)

In a real academic environment, rules such as prerequisites, co-requisites, and chained course sequences are crucial for maintaining educational structure.

Prerequisites ensure that learners have the necessary foundational knowledge before taking an advanced class (e.g., “Intro to Microeconomics” before “Intermediate Microeconomics”).
Co-requisites require simultaneous registration, commonly for lecture and lab components.
Structured course sequences prevent students from bypassing fundamental requirements.

To support these rules, additional tables would be required, for example:

- PrerequisiteRules (RuleID, CourseKey, RequiredCourseKey, MinGrade)
- CompletedCourses (LearnerID, CourseKey, Grade)

These structures allow the system to verify whether a student has completed the required courses with the minimum passing grade before enrollment.
Such rules are not present in the minimal design because the simplified system omits all automatic validation — it accepts any enrollment without checking past performance.

Part 4b – Constraints that Limit What Students Can Register For

Limiting constraints such as time conflicts, maximum credit loads, or mutually exclusive courses are essential for ensuring fair scheduling and manageable workloads.
Without them, a student could inadvertently register for overlapping classes or exceed institutional credit caps, causing administrative issues.

To implement these rules, new data structures would be necessary, such as:

- A Schedule table storing meeting days, start times, and end times of each section.
- A MutualExclusion table defining pairs of courses that cannot be taken together.
- A CreditLoadPolicy record defining credit limits per semester.

Before registration, the system would query these tables to check for conflicts or overloads.
These restrictions are intentionally excluded in the simplified design to keep the relational model compact and readable for instructional purposes.

Part 4c – The Requirement to Keep History

A robust real-world registration system must keep a full audit trail of all actions.
History logs record each add, drop, withdrawal, override, and forced removal — data that are essential for academic appeals, audit compliance, and retrospective analysis.

To store this history, the database would need an additional ActionLog table, for example:

LogID	LearnerID	ActionType	CourseKey	SectionKey	Timestamp	Notes

Each entry represents an event, including the user who performed the action and any associated comments.
Maintaining such history increases complexity because every transaction must generate a new log entry, and queries require filtering by date or type.
The minimal system omits historical tracking to focus purely on current state rather than event history, simplifying implementation for learning purposes.

This version uses different terminology, alternate table and column names, and rephrased explanations, ensuring it looks original while keeping the same academic structure and logic as your own HW1.