1. Entity-Relationship (E-R) Diagram

The following Entity-Relationship diagram illustrates the complete database schema for the University Registration System. It shows all entities, their attributes, relationships, and cardinalities.

Key Components

- **Entities:** Represented as blue rectangles (College, Department, Instructor, Student, Course, CourseSection, CourseSchedule, Classroom, Enrollment, Cart)
- **Relationships:** Shown as green diamonds connecting entities
- **Attributes:** Displayed as ovals, with primary keys highlighted in yellow
- **Cardinality:** Indicated by notations (1, N, M) showing one-to-many or many-to-many relationships

2. Relational Database Schema

This section provides a detailed description of each table in the relational database schema, including design rationale and query optimization considerations.

2.1 College

Purpose: Represents the top-level academic organizational units within the university.

Column	Type	Constraints	Description
college_id	INT	PRIMARY KEY	Unique identifier for each college
name	VARCHAR(100)	NOT NULL	College name

Design Rationale: Simple structure as colleges are top-level entities. The college_id serves as the primary key for efficient joins with the Department table.

2.2 Department

Purpose: Represents academic departments that belong to colleges and organize courses, instructors, and student majors.

Column	Type	Constraints	Description
department_id	INT	PRIMARY KEY	Unique identifier for each department
name	VARCHAR(100)	NOT NULL	Department name

college_id	INT	FOREIGN KEY → College	Parent college reference

Design Rationale: Establishes hierarchical organization (College → Department). A foreign key to College enables efficient queries for all departments within a college.

2.3 Instructor

Purpose: Stores faculty information and their departmental affiliation.

Column	Type	Constraints	Description
instructor_id	INT	PRIMARY KEY	Unique identifier for each instructor
name	VARCHAR(100)	NOT NULL	Instructor name
department_id	INT	FOREIGN KEY → Department	Home department

Query Optimization: Foreign key index on department_id enables fast filtering of instructors by department and supports efficient joins when finding courses taught by department faculty.

2.4 Student

Purpose: Stores student information, including their academic year and major.

Column	Type	Constraints	Description
student_id	INT	PRIMARY KEY, IDENTITY(1000,1)	Auto-generated student ID starting at 1000
name	VARCHAR(100)	NOT NULL	Student name
year	INT	NULL	Academic year (1-4, or NULL for undeclared)
major_department_id	INT	FOREIGN KEY → Department, NULL	Major department (nullable for undeclared)

Design Rationale: The IDENTITY column automatically assigns student IDs starting from 1000. Nullable major_department_id allows for undeclared students. A foreign key relationship enables queries like 'all students in the Computer Science department.'

2.5 Classroom

Purpose: Represents physical locations where course sections meet.

Column	Type	Constraints	Description
classroom_id	INT	PRIMARY KEY, IDENTITY	Auto-generated identifier
building	VARCHAR(1)	NOT NULL	Building code (A-Z)
room_number	VARCHAR(3)	NOT NULL	Room number
capacity	INT	NOT NULL, CHECK > 0	Maximum occupancy
room_type	VARCHAR(20)	CHECK IN ('Lecture', 'Lab'), NULL	Type of classroom
UNIQUE		(building, room_number)	Ensures no duplicate rooms

Design Rationale: Unique constraint on (building, room_number) prevents duplicate room entries. The room_type field helps match course requirements (e.g., labs need lab rooms). A capacity check ensures data validity and supports enrollment capacity validation.

2.6 Course

Purpose: Defines abstract course offerings (e.g., 'CS101: Introduction to Programming').

Column	Type	Constraints	Description
course_id	INT	PRIMARY KEY, IDENTITY	Unique course identifier
course_code	VARCHAR(10)	NOT NULL	Course code (e.g., CS101)
title	VARCHAR(100)	NOT NULL	Course title
department_id	INT	FOREIGN KEY → Department	Offering department

Query Optimization: Index on department_id enables fast retrieval of all courses offered by a department. Queries like 'Show all Computer Science courses' benefit from this indexed foreign key.

2.7 Course Section

Purpose: Represents specific offerings of a course in a particular term/year with a specific instructor.

Column	Type	Constraints	Description
section_id	INT	PRIMARY KEY, IDENTITY	Unique section identifier
course_id	INT	FOREIGN KEY → Course	Parent course
instructor_id	INT	FOREIGN KEY → Instructor	Assigned instructor
term	VARCHAR(10)	CHECK IN ('Fall', 'Winter', 'Spring', 'Summer')	Academic term
year	INT	CHECK >= 2024 AND <= 2027	Academic year
section_number	VARCHAR(5)	NOT NULL	Section identifier (e.g., '001', 'A')
max_capacity	INT	CHECK > 0	Maximum enrollment
UNIQUE		(course_id, term, year, section_number)	Prevents duplicate sections

Design Rationale: Separates course definition from course offerings, allowing the same course (e.g., CS101) to be offered multiple times. Unique constraint ensures no duplicate sections for the same course/term/year. Term and year constraints validate data integrity.

2.8 Course Schedule

Purpose: Stores meeting times and locations for course sections (supports multiple meeting times per section).

Column	Type	Constraints	Description
schedule_id	INT	PRIMARY KEY, IDENTITY	Unique schedule entry
section_id	INT	FOREIGN KEY → CourseSection	Associated section
day_of_week	VARCHAR(10)	CHECK IN (Monday-Friday)	Meeting day
start_time	TIME	NOT NULL	Class start time

end_time	TIME	NOT NULL	Class end time
classroom_id	INT	FOREIGN KEY → Classroom, NULL	Assigned room (nullable)
CHECK		start_time < end_time	Validates time order
CHECK		DATEDIFF(MINUTE, ...) <= 180	Max 3-hour duration

Design Rationale: One-to-many relationship with CourseSection allows courses to meet multiple times per week (e.g., MWF). Nullable classroom_id supports online courses. Time validation prevents logical errors.

Query Optimization: Foreign key indexes support efficient queries like 'find all courses meeting on Monday' or 'check room availability on Tuesday at 2pm.'

2.9 Prerequisite

Purpose: Defines prerequisite relationships between courses.

Column	Type	Constraints	Description
prerequisite_id	INT	PRIMARY KEY, IDENTITY	Unique identifier
course_id	INT	FOREIGN KEY → Course	Course requiring a prerequisite
prerequisite_course_id	INT	FOREIGN KEY → Course	Required prerequisite course
minimum_grade	VARCHAR(2)	CHECK IN (A+ to F), NULL	Minimum passing grade
CHECK		course_id <> prerequisite_course_id	Prevents self-prerequisites
UNIQUE		(course_id, prerequisite_course_id)	One prerequisite per pair

Design Rationale: The self-referential table allows flexible prerequisite chains (e.g., CS101 → CS201 → CS301). Check constraint prevents courses from being their own prerequisite. Supports many-to-many relationships (a course can have multiple prerequisites, and be a prerequisite for multiple courses).

2.10 Enrollment

Purpose: Records student registrations in course sections, tracking status and final grades.

Column	Type	Constraints	Description
enrollment_id	INT	PRIMARY KEY, IDENTITY	Unique enrollment record
student_id	INT	FOREIGN KEY → Student	Enrolled student
course_id	INT	FOREIGN KEY → Course	Course enrolled in
section_id	INT	FOREIGN KEY → CourseSection	Specific section
status	VARCHAR(20)	CHECK IN ('Enrolled', 'Completed', 'Withdrawn'), DEFAULT 'Enrolled'	Current status
grade	VARCHAR(2)	CHECK IN (A+ to F, I, W), NULL	Final grade
UNIQUE		(student_id, section_id)	Prevents duplicate enrollments

Design Rationale: Denormalized design includes both course_id and section_id for query performance. The unique constraint prevents double-enrollment. Status field tracks lifecycle (Enrolled → Completed/Withdrawn).

Query Optimization: Indexes on student_id and section_id support fast queries for 'student's current schedule' and 'section enrollment count.' The course_id redundancy avoids joins when querying completed courses for prerequisite validation.

2.11 Cart

Purpose: Temporary storage for courses students plan to register for before finalizing enrollment.

Column	Type	Constraints	Description
cart_id	INT	PRIMARY KEY, IDENTITY	Unique cart item
student_id	INT	FOREIGN KEY → Student	Student's cart
course_id	INT	FOREIGN KEY → Course	Course in cart
section_id	INT	FOREIGN KEY → CourseSection	Specific section selected
UNIQUE		(student_id, section_id)	One instance per section

Design Rationale: Separate from Enrollment to distinguish planning from actual registration. Similar structure to Enrollment for easy migration when the student finalizes registration. Unique constraint prevents duplicate cart items.

2.12 Materialized View: mv_StudentCompletedCourses

Purpose: Optimized view for prerequisite validation, providing fast access to completed courses with grades.

Definition: Indexes enrollments where status='Completed' and grade is valid (not I or W).

Included Columns: student_id, course_id, course_code, title, grade, section_id, term, year

Indexes:

- **Clustered:** enrollment_id (for uniqueness)
- **Non-clustered:** (student_id, course_id) for prerequisite checks
- **Non-clustered:** (grade, student_id) for GPA calculations

Query Optimization: Eliminates the need to filter the Enrollment table for completed courses during registration. Prerequisite validation query 'Has student X completed course Y with grade >= B?' becomes an index seek instead of a table scan. Dramatically improves performance for enrollment validation.

2.13 Overall Query Optimization Strategy

- **Foreign Key Indexes:** All foreign keys are automatically indexed, supporting efficient joins and parent-child queries
- **Unique Constraints:** Prevent duplicate data and create implicit indexes for fast lookups
- **Materialized Views:** Pre-computed results for expensive queries (completed courses)
- **Denormalization:** Strategic redundancy (e.g., course_id in Enrollment) reduces joins
- **Check Constraints:** Validate data at insertion, preventing invalid queries later
- **IDENTITY Columns:** Auto-generated keys reduce locking and improve insert performance

3. Business Rules Summary

The following business rules govern the operation of the University Registration System and ensure data integrity.

3.1 Academic Organization

- The university is organized into colleges, which contain multiple departments
- Each department must belong to exactly one college
- Instructors are affiliated with a specific department
- Courses are offered by departments and may be taught by any instructor in that department

3.2 Student Rules

- Students are automatically assigned unique IDs starting from 1000
- Students may have a declared major in a department, or remain undeclared (NULL major)
- Students must complete prerequisites before enrolling in dependent courses
- Students cannot enroll in the same section multiple times
- Students can add courses to a shopping cart before finalizing registration

3.3 Course and Section Rules

- Courses are abstract offerings; sections are specific instances with instructors and schedules
- Each section must have a unique combination of course, term, year, and section number
- Sections have maximum capacity limits that cannot be exceeded
- Sections can meet multiple times per week with different schedules
- Each section is taught by exactly one instructor

- Terms are restricted to: Fall, Winter, Spring, Summer
- Academic years must be between 2024 and 2027

3.4 Schedule Rules

- Course meetings are scheduled Monday through Friday only
- Start time must be before end time
- Maximum class duration is 3 hours (180 minutes)
- Classes may be assigned to physical classrooms or be online (NULL classroom)
- Classrooms are identified by unique building and room number combinations

3.5 Prerequisite Rules

- Courses may require one or more prerequisite courses
- Each prerequisite may specify a minimum grade requirement (A+ through F)
- Courses cannot be their own prerequisite
- Prerequisites must be completed with a passing grade before enrollment
- Only one prerequisite relationship can exist between any two courses

3.6 Enrollment Rules

- Enrollment status can be: Enrolled, Completed, or Withdrawn
- Default enrollment status is 'Enrolled'
- Grades are assigned only for completed courses
- Valid grades: A+, A, A-, B+, B, B-, C+, C, D, F (letter grades), I (Incomplete), W (Withdrawn)
- Incomplete (I) and Withdrawn (W) grades do not count toward prerequisite completion
- Students cannot be enrolled in a section if they have already completed the parent course

3.7 Classroom Rules

- Classrooms have a maximum capacity that must be greater than zero
- Classrooms may be designated as 'Lecture' or 'Lab' type
- Room capacity should be sufficient for the section's max_capacity
- No two classrooms can have the same building and room number combination

3.8 Data Integrity Rules

- All foreign key relationships must reference existing records (referential integrity)
- Deleting a college requires first reassigning or deleting all departments in that college
- Deleting a course requires first removing all sections, prerequisites, and enrollments
- Student IDs, once assigned, are never reused
- All required fields (NOT NULL constraints) must have values