1. What tables should you build? (3 PTS)

Courses

course_id (Primary Key)
mnemonic (e.g., course code or abbreviation)
course_name
course_description
active (Flag indicating if the course is currently active)

Learning Outcomes

outcome_id (Primary Key)
course_id (Foreign Key referencing Courses)
learning_outcomes (Description of the learning outcome)
active (Flag indicating if the learning outcome is currently active)

Instructor

instructor_id (Primary Key)
instructor_name
active (Flag indicating if the instructor is currently employed)

Course_Assignment

assignment_id (Primary Key)
course_id (Foreign Key referencing Courses)
instructor_id (Foreign Key referencing Instructor)
term (e.g., Spring, Summer, Fall)

2. For each table, what field(s) will you use for the primary key? (2 PTS)

Courses: course_id
Learning Outcomes: outcome_id
Instructor: instructor_id
Course_Assignment: assignment_id

3. For each table, what foreign keys will you use? (2 PTS)

Learning Outcomes:
course_id (Foreign Key referencing Courses)
Course_Assignment:
course_id (Foreign Key referencing Courses)
instructor_id (Foreign Key referencing Instructor)
The other two tables do not have a foreign key

4. Learning outcomes, courses, and instructors need a flag to indicate if they are currently active or not. How will your database support this feature? (2 PTS)

To track if a course, learning outcome, or instructor is currently active, each relevant table will include a boolean field named active. This can look like using active = TRUE to see learning outcomes, courses, and instructors that are active.

5.. Is there anything to normalize in the database, and if so, how will you normalize it? (1 PT)

Yes, normalization is essential to eliminate redundancy.

I separated information regarding courses and instructors so that information regarding learning outcomes, descriptions, and active status are not duplicated. Next I created primary keys for each table and columns that are not primary or foreign keys are dependent on the primary key.

6. Are there indexes that you should build? Explain your reasoning. (1 PT)

Yes, indexes will improve join operations and query performance, especially as the dataset grows, Here are some that I made.

Courses: Index on course_id

Learning Outcomes: Index on course_id

Instructor: Index on instructor_id

Course_Assignment: Index on course_id, instructor_id, and term

7. Are there constraints to enforce? Explain your answer and strategy. (2 PTS)

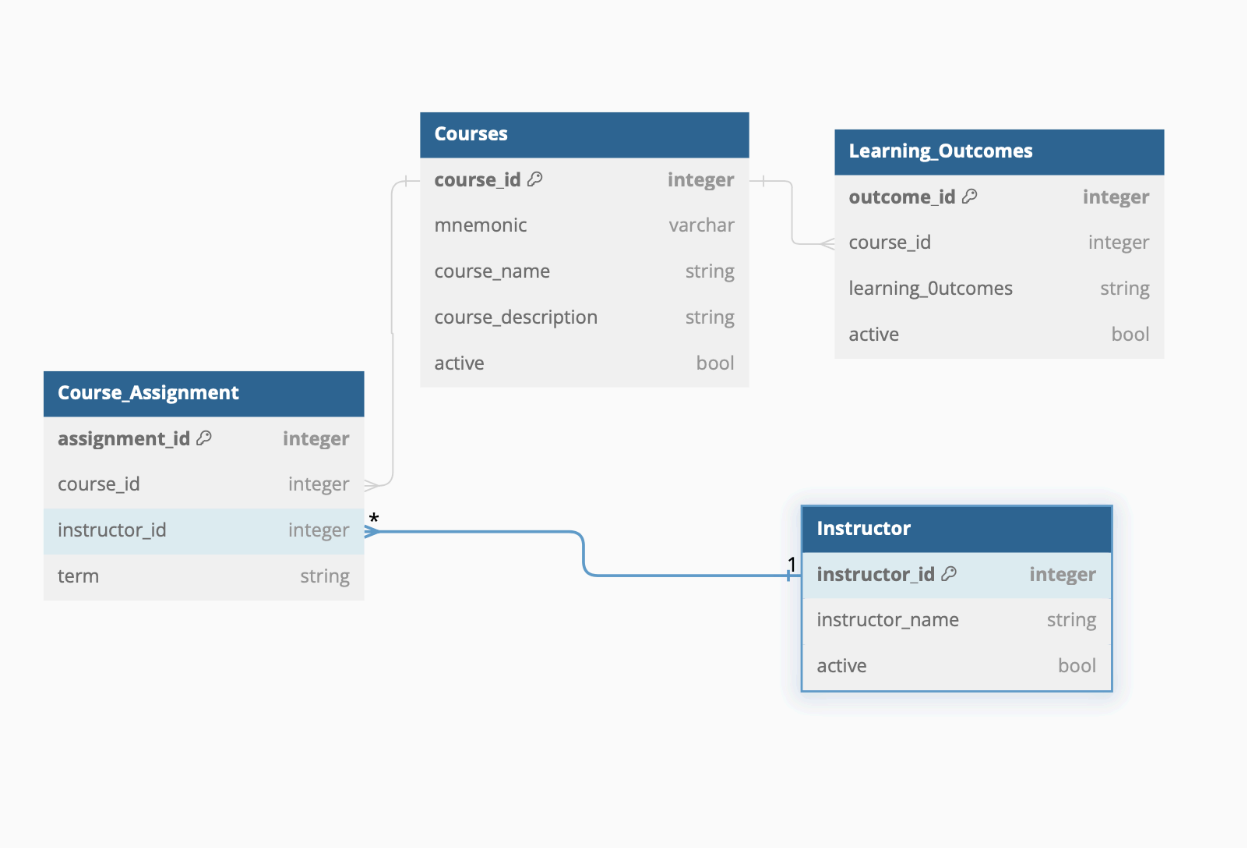
Yes! Here are several

Valid data constraints:

- **Instructor**
- **Courses**

All entries are not null and should have a single value

8. Draw and submit a Relational Model for your project. (5 PTS)



Courses	
course_id	integer
mnemonic	varchar
course_name	string
course_description	string
active	bool

Learning_Outcomes	
outcome_id	integer
course_id	integer
learning_Outcomes	string
active	bool

Course_Assignment	
assignment_id	integer
course_id	integer
instructor_id	integer
term	string

Instructor	
instructor_id	integer
instructor_name	string
active	bool

9. Suppose you were asked if your database could also support the UVA SDS Residential MSDS Program. Explain any issues that might arise, changes to the database structure (schema), and new data that might be needed. (2 PTS)

The Residential MSDS Program has a different set of courses and different terms. In order to accommodate this we would need new data about the courses, offering times, professors, and LOs for the course, and probably some way to differentiate between courses provided online, in person, or both.