Review Lectures 5-10

Introduction to Database Management

CS348 Fall 2022

Announcements (Tue, Oct 25)

- Milestone 1
  - Feedback on Nov 2
- Midterm Exam
  - Fri, Nov 4, 4:30-6:00pm
  - **Cover Lectures 1-6** [instead of Lectures 1-10]
- Assignment 2
  - Due date [Thur, Oct 27, 11:59pm → Mon, Oct 31, 11:59pm]
  - Grade won't be released before midterm exam, but we will cover solutions related to Lectures 1-6 on the midterm review lecture on Thur, Nov 3.
- Final Exam
  - Tue, Dec 13, 7:30pm – 10:00pm

SQL

- Basic SQL (queries, modifications, and constraints)

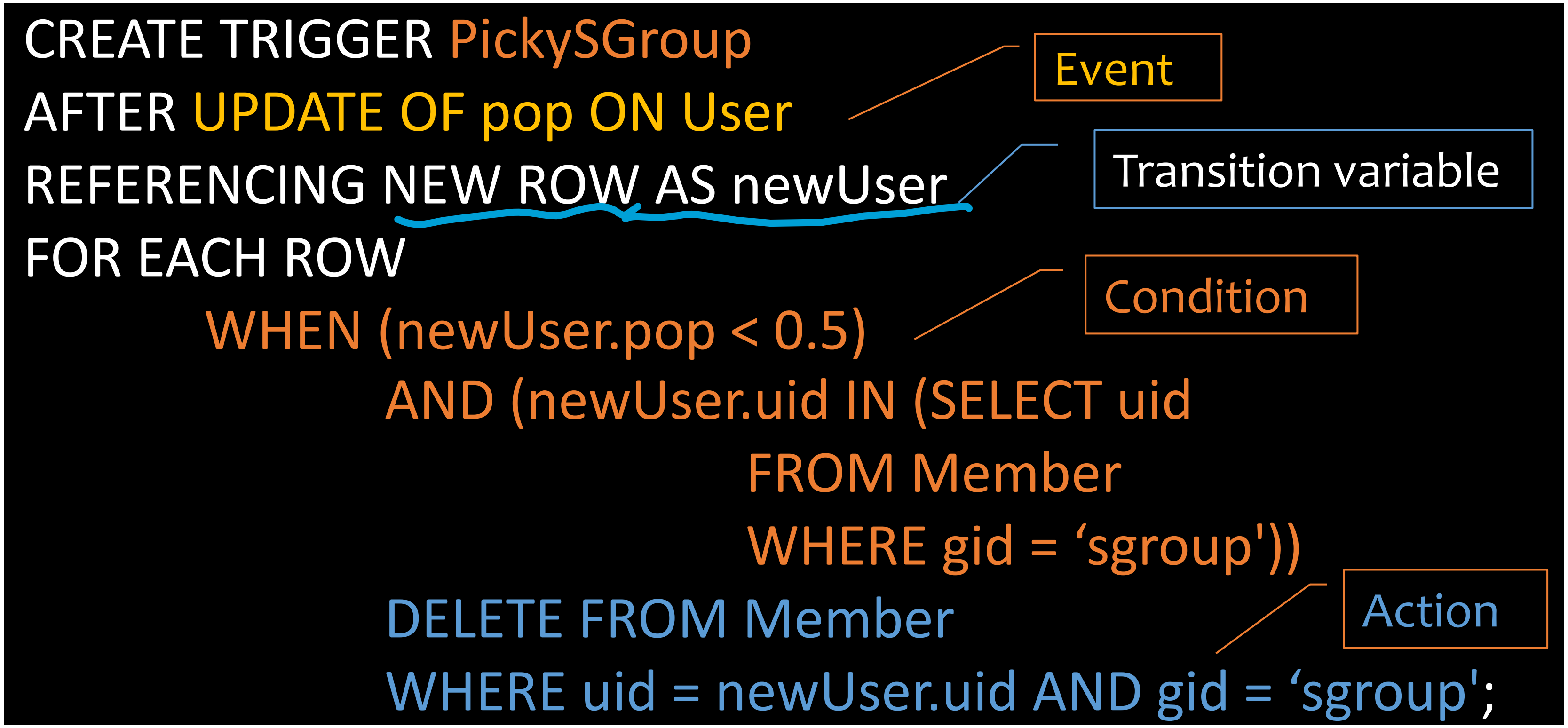
- Intermediate SQL
  - Triggers
  - Views
  - Indexes

- Advanced SQL
  - Programming
  - Recursive queries (Optional)

Lectures 5-6

Triggers

- A trigger is an event-condition-action (ECA) rule
  - When event occurs, test condition; if condition is satisfied, execute action



```
CREATE TRIGGER PickySGroup
AFTER UPDATE OF pop ON User
REFERENCING NEW ROW AS newUser
FOR EACH ROW
        WHEN (newUser.pop < 0.5)
              AND (newUser.uid IN (SELECT uid
                                   FROM Member
                                   WHERE gid = ‘sgroup'))
              DELETE FROM Member
              WHERE uid = newUser.uid AND gid = ‘sgroup';
```

Transition variables/tables

- OLD ROW: the modified row before the triggering event
- NEW ROW: the modified row after the triggering event
- OLD TABLE: a hypothetical read-only table containing all rows to be modified before the triggering event
- NEW TABLE: a hypothetical table containing all modified rows after the triggering event

Event	Row	Statement
Delete	old r; old t	old t
Insert	new r; new t	new t
Update	old/new r; old/new t	old/new t

AFTER Trigger

Event	Row	Statement
Update	old/new r	-
Insert	new r	-
Delete	old r	-

BEFORE Trigger

Statement- vs. row-level triggers

- Simple row-level triggers are easier to implement
  - Statement-level triggers: require significant amount of state to be maintained in OLD TABLE and NEW TABLE

- Exercise 1: However, can you think of a case when a row-level trigger may be less efficient?

- Exercise 2: Certain triggers are only possible at statement level. Can you think of an example?

INSTEAD OF triggers for views

```
CREATE VIEW AveragePop(pop_avg) AS
        SELECT AVG(pop) FROM User;
```

```
CREATE TRIGGER AdjustAveragePop
INSTEAD OF UPDATE ON AveragePop
REFERENCING OLD ROW AS o,
        NEW ROW AS n
FOR EACH ROW
        UPDATE User
        SET pop = pop + (n.pop_avg-o.pop_avg);
```

0.5

0.4

User

…	pop	…
	0.4 +0.1	
	0.4+0.1	
	0.5 +0.1	
	0.3+0.1	

- What does this trigger do?

```
UPDATE AveragePop SET pop_avg = 0.5;
```

Programming (Lecture 6)

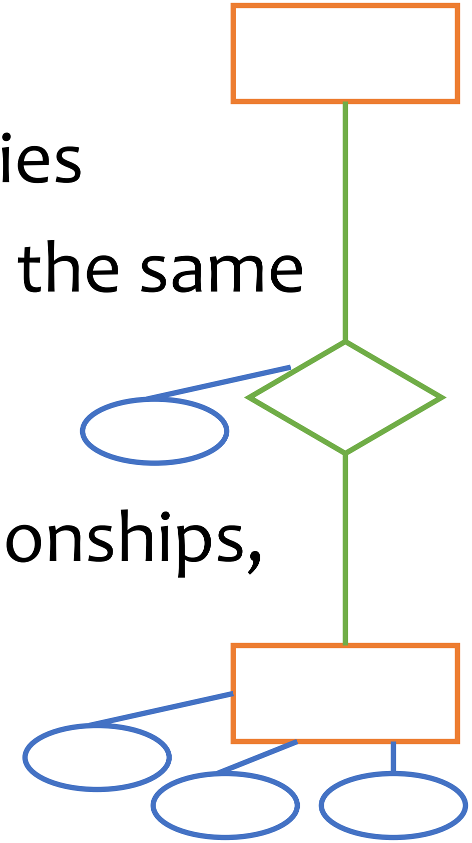
- Pros and cons of SQL
  - Very high-level, possible to optimize
  - Not intended for general-purpose computation
- Solutions
  - Augment SQL with constructs from general-purpose programming languages
    - E.g.: SQL/PSM
  - Use SQL together with general-purpose programming languages: many possibilities
    - Through an API, e.g., Python psycopg2
    - Embedded SQL, e.g., in C
    - Automatic object-relational mapping, e.g.: Python SQLAlchemy
    - Extending programming languages with SQL-like constructs, e.g.: LINQ

Database Design

- Entity-Relationship (E/R) model (Lecture 7)
- Translating E/R to relational schema (Lecture 8)
- Relational design principles (Lectures 9-10)

E/R basics (Lecture 7)

- Entity: a "thing," like an object
- Entity set: a collection of things of the same type, like a relation of tuples or a class of objects
    - Represented as a rectangle
- Relationship: an association among entities
- Relationship set: a set of relationships of the same type (among same entity sets)
    - Represented as a diamond
- Attributes: properties of entities or relationships, like attributes of tuples or objects
    - Represented as ovals



Summary of E/R concepts

- Entity sets
  - Keys
  - Weak entity sets
- Relationship sets
  - Attributes of relationships
  - Multiplicity
  - Roles
  - Supporting relationships (related to weak entity)
  - ISA relationships
- Other extensions:
  - Generalization
  - Structured attributes
  - Aggregation

Case study 3 (Exercise)

- A Registrar's Database:
  - Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
  - Most course sections are taught on-site, but a few are taught at off-site locations.
  - Students have student numbers and names.
  - Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
  - Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
  - A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

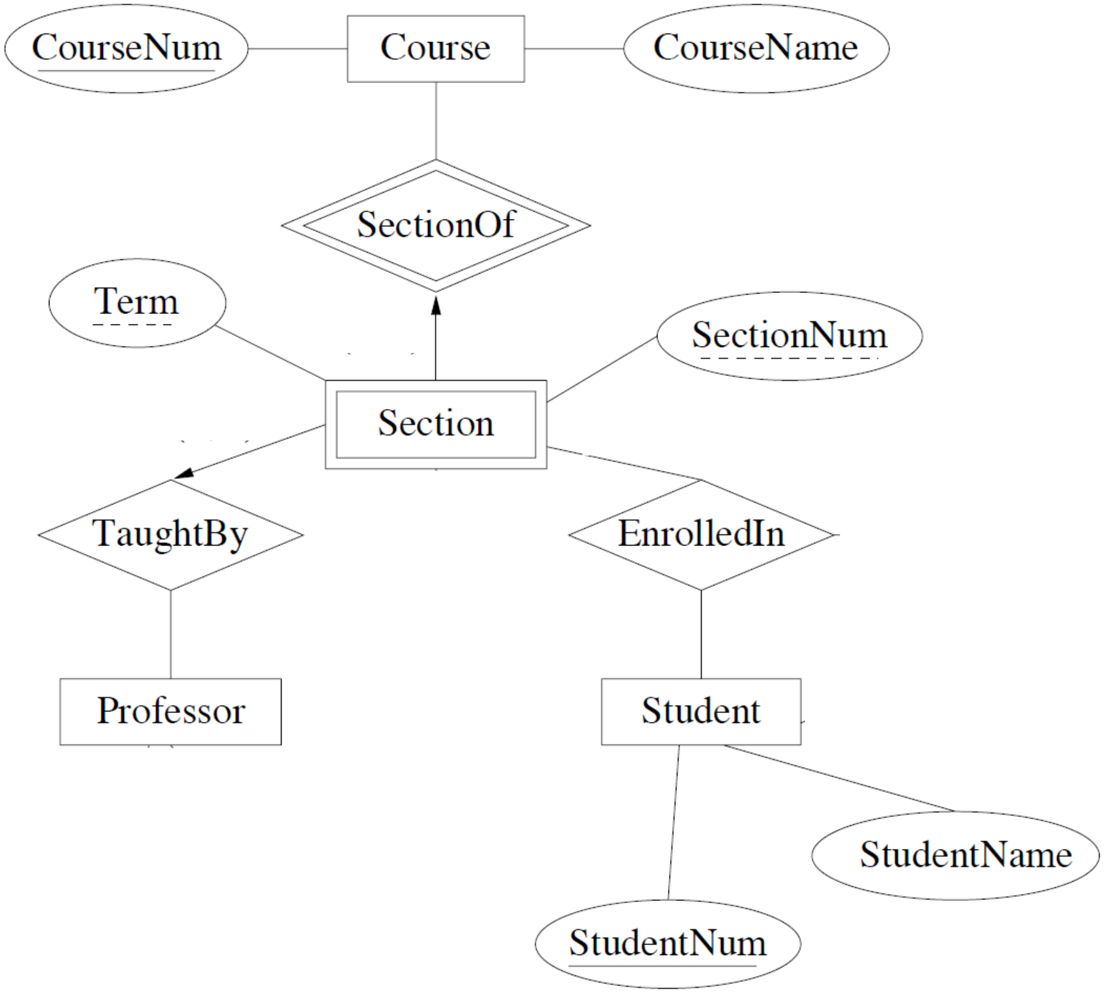
Case study 3 (Exercise)

- A Registrar’s Database:
  - Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
  - Most course sections are taught on-site, but a few are taught at off-site locations
  - Students have student numbers and names.
  - Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
  - Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
  - A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

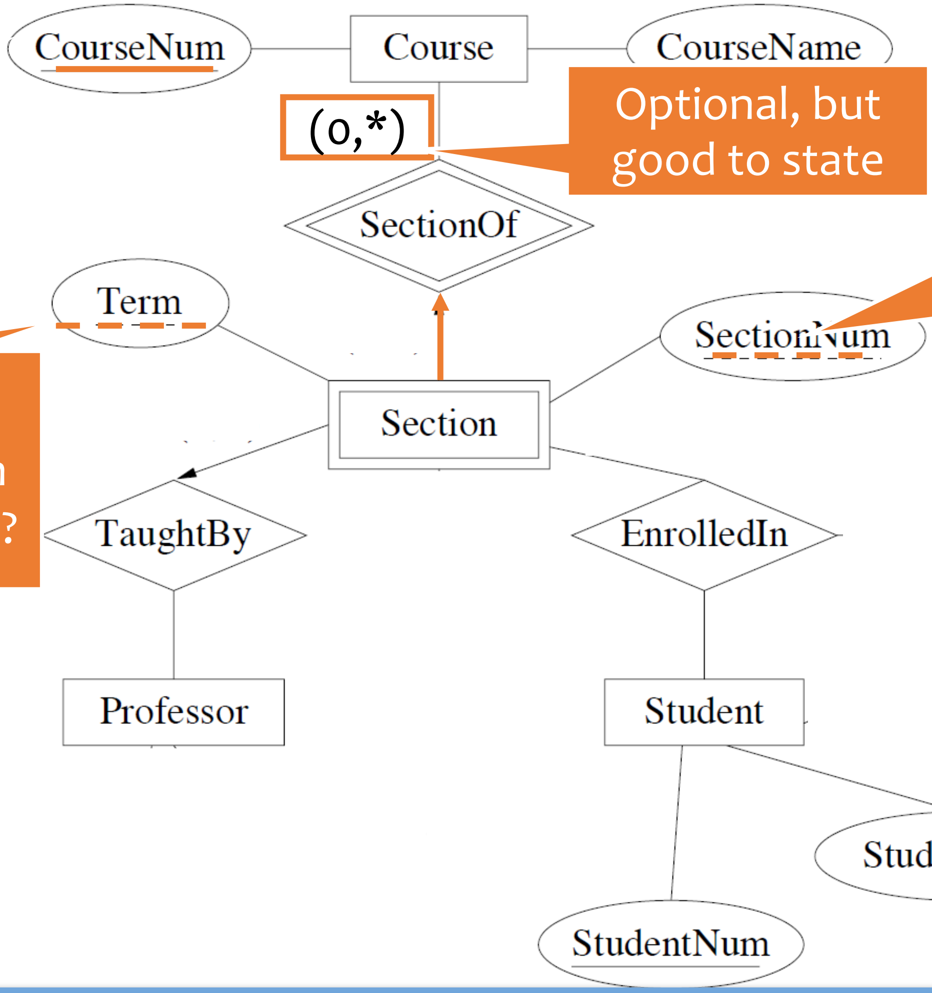
Case study 3 (Exercise)

- A Registrar's Database:
  - Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
  - Most course sections are taught on-site, but a few are taught at off-site locations.
  - Students have student numbers and names.
  - Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
  - Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
  - A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

Case study 3 (Exercise) cont.



Case study 3 (Exercise) cont.



Assume it is unique

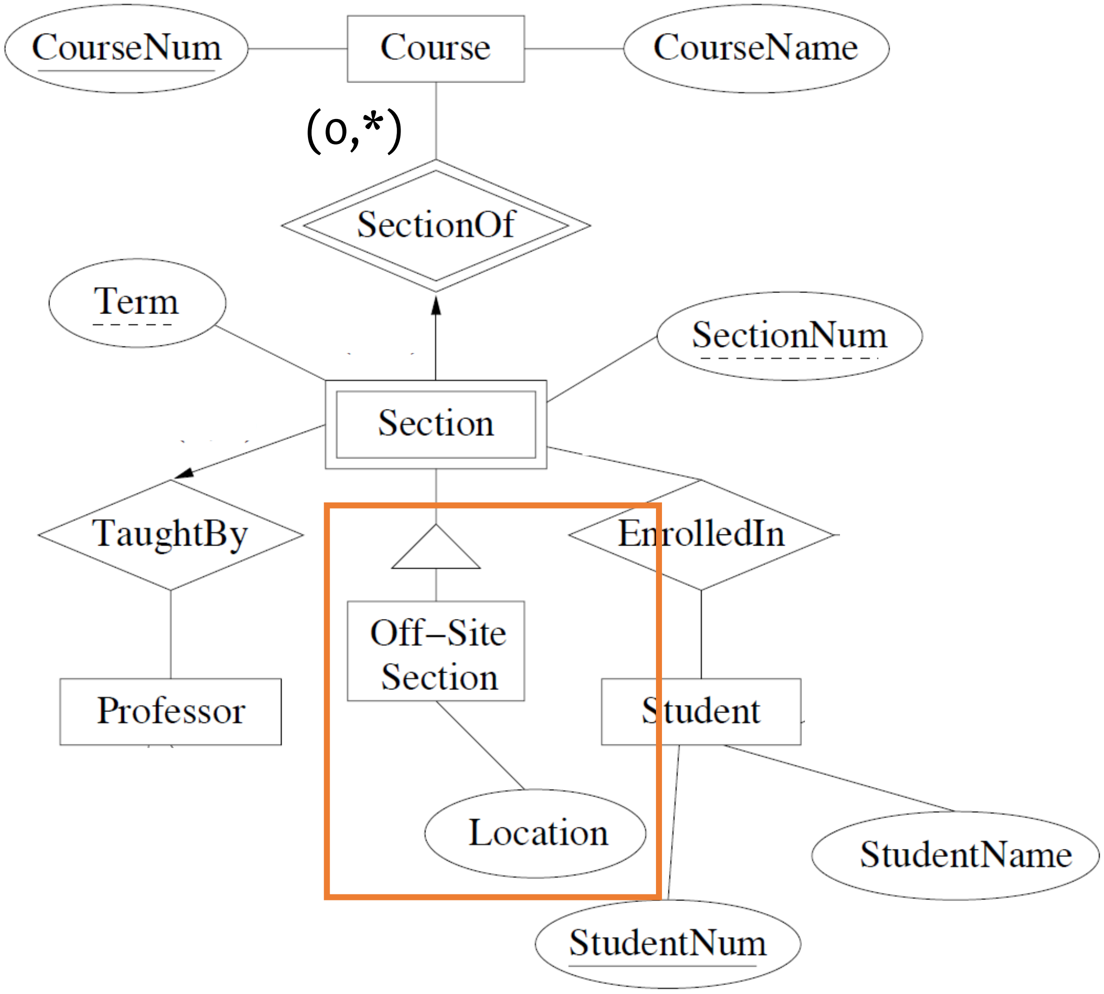
Optional, but good to state

Assume (term,sectionNum) is unique given the courseNum

Question: can we place "term" as an attribute of Course?

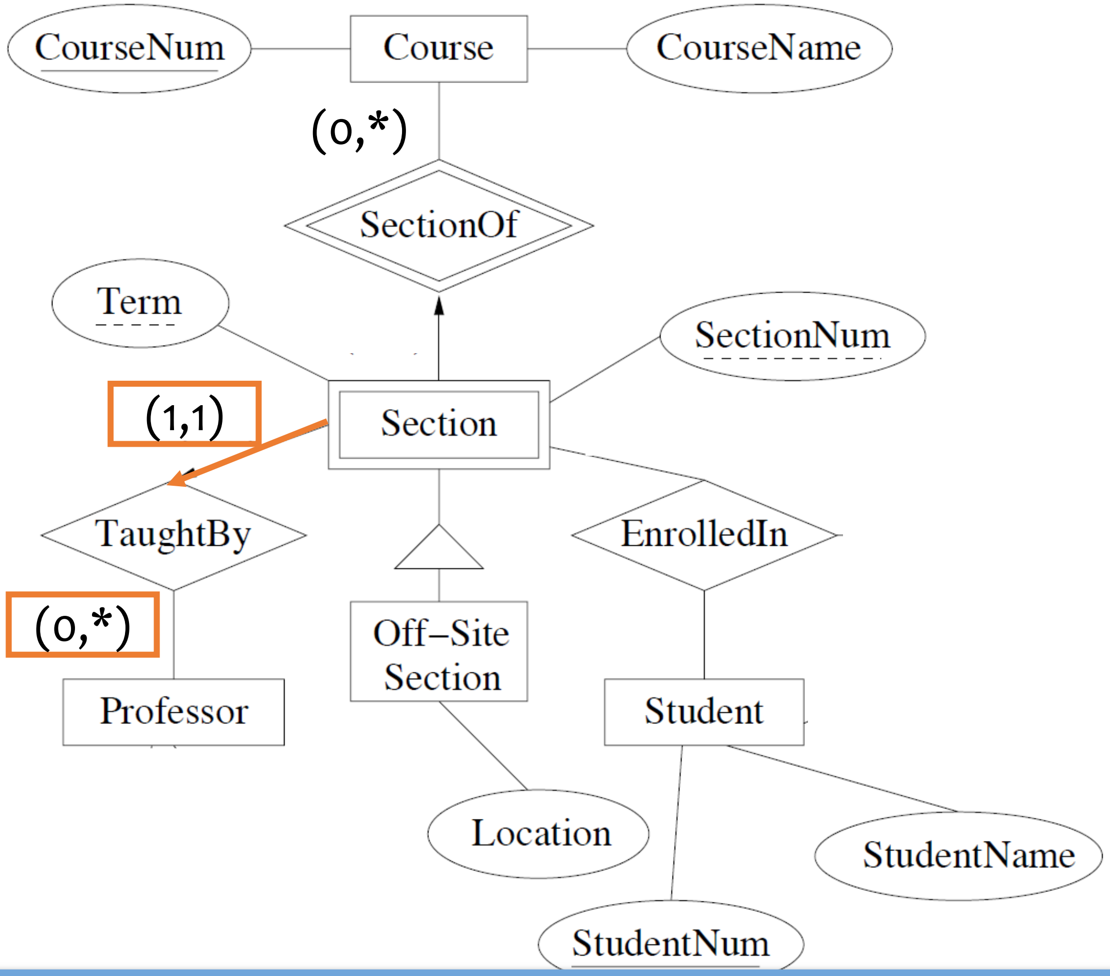
Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.

Case study 3 (Exercise) cont.



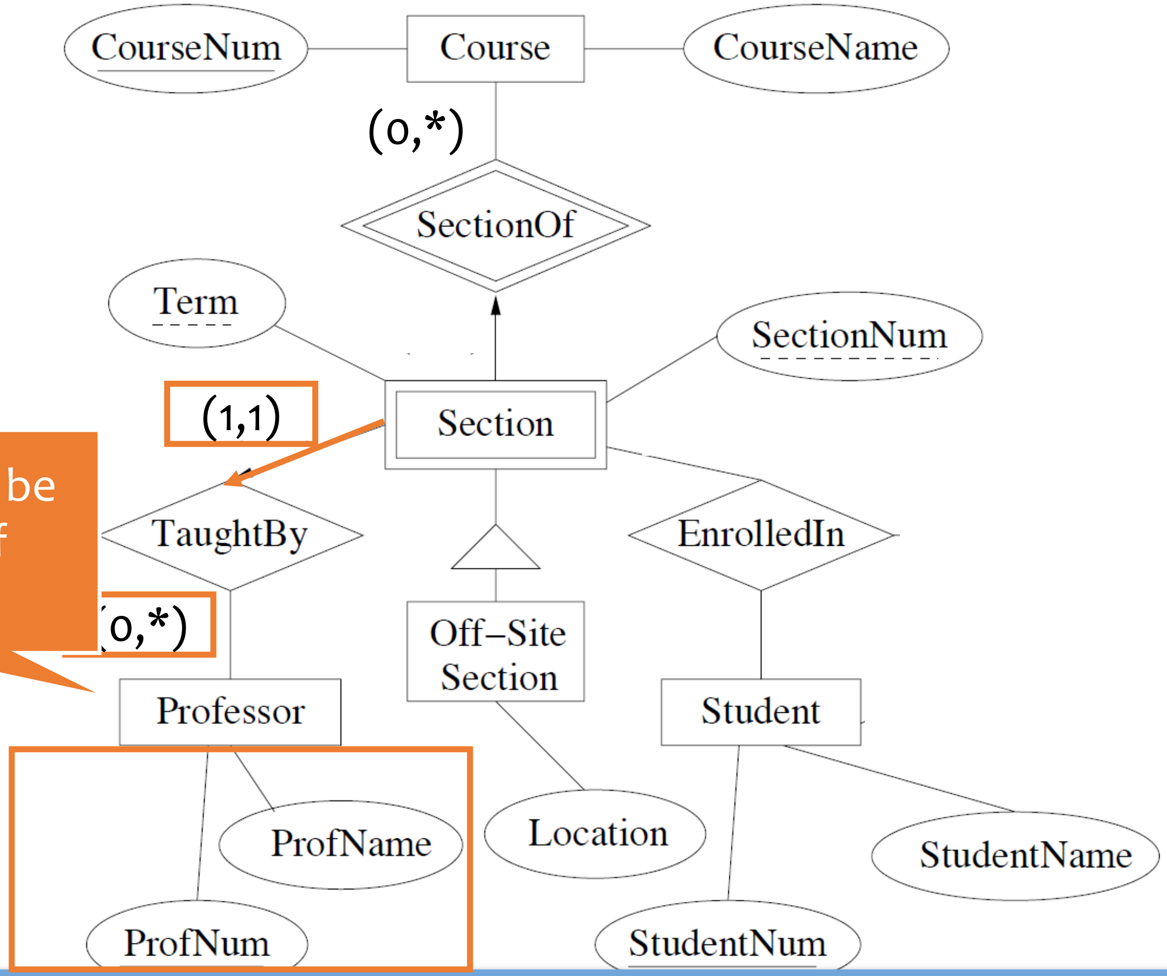
Most course sections are taught on-site, but a few are taught at off-site locations.

Case study 3 (Exercise) cont.



Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

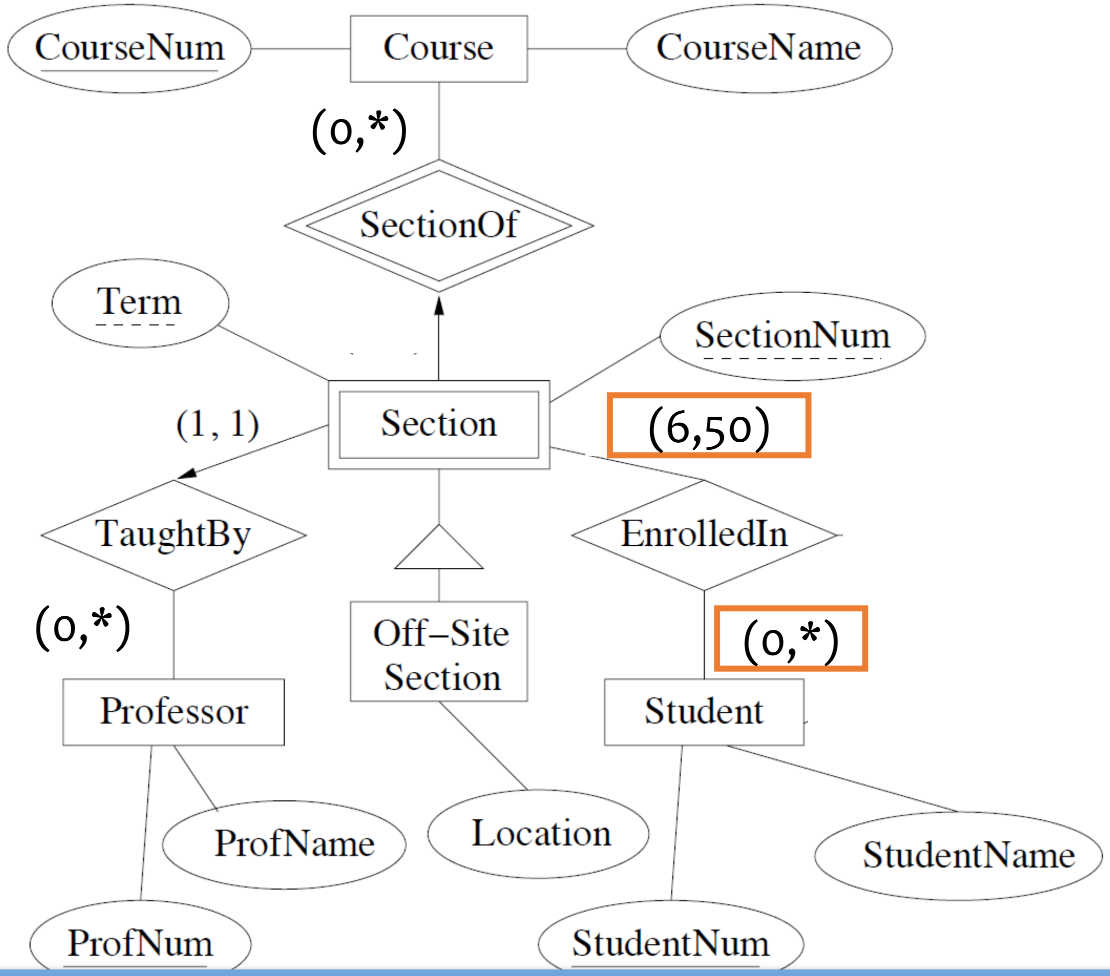
Case study 3 (Exercise) cont.



Can “Professor” be an attribute of Section?

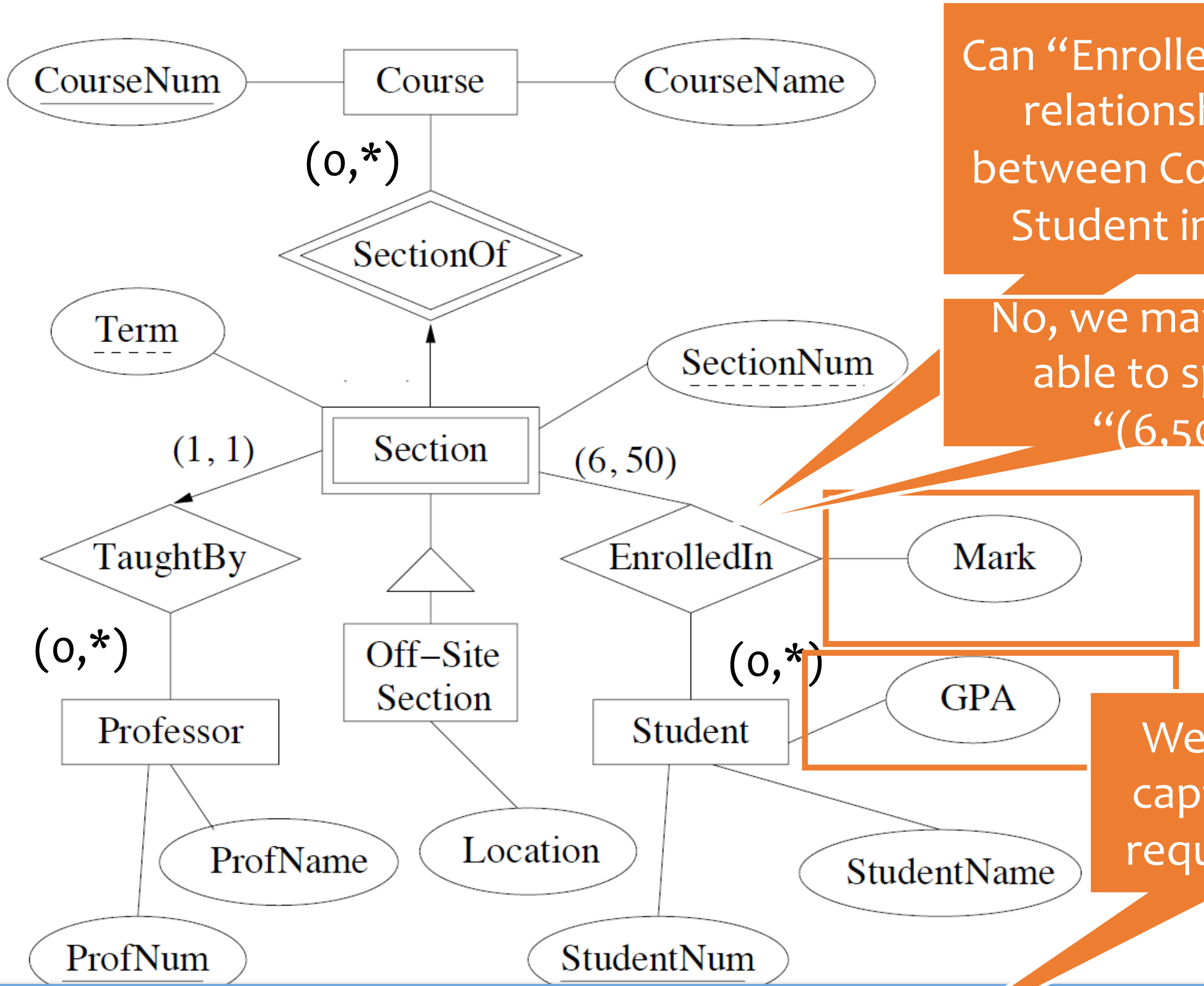
Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

Case study 3 (Exercise) cont.



Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.

Case study 3 (Exercise) cont.



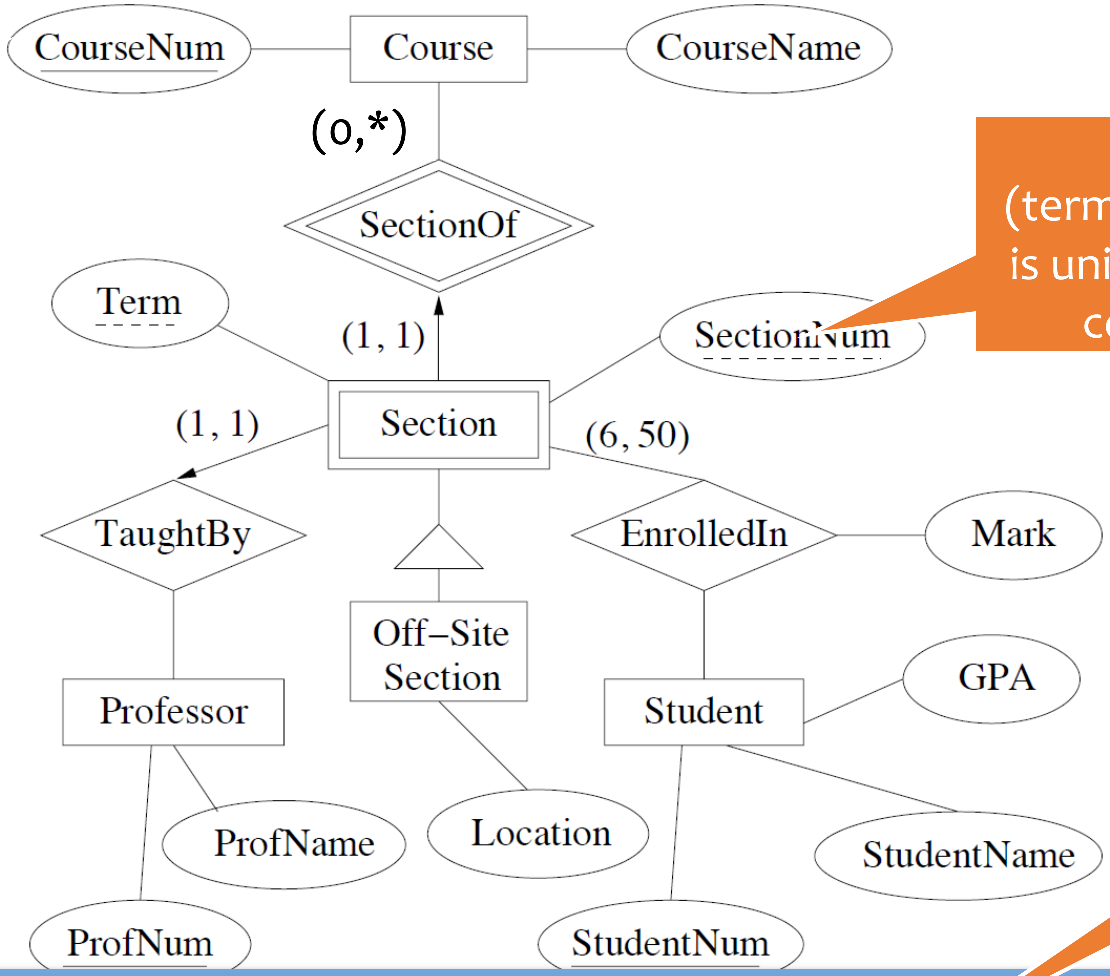
Can "EnrolledIn" be a relationship set between Course and Student instead?

No, we may not be able to specify "(6,50)"

We did not capture this requirement

A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

Case study 3: possible solution



Assume it is unique

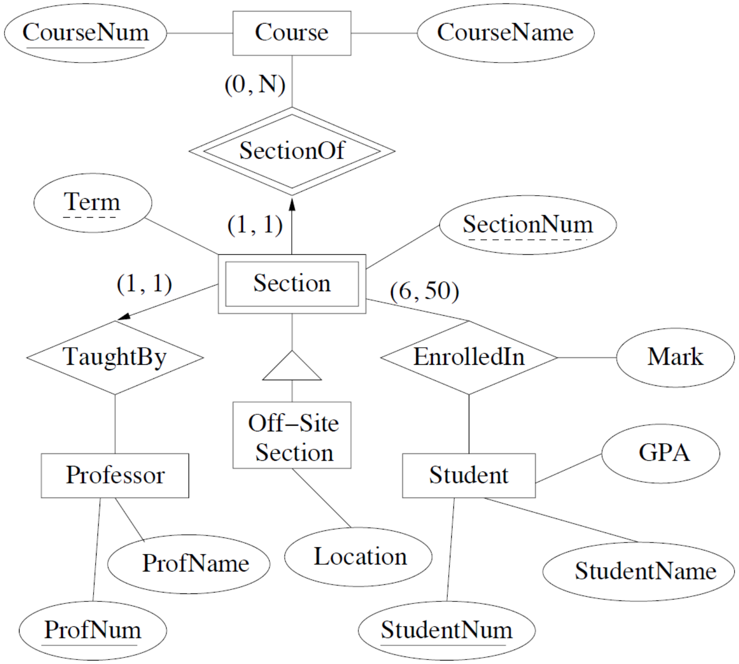
Assume (term,sectionNum) is unique given the courseNum

We did not capture this requirement

A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

More examples (Exercise) (Lecture 8)

- ER Diagram

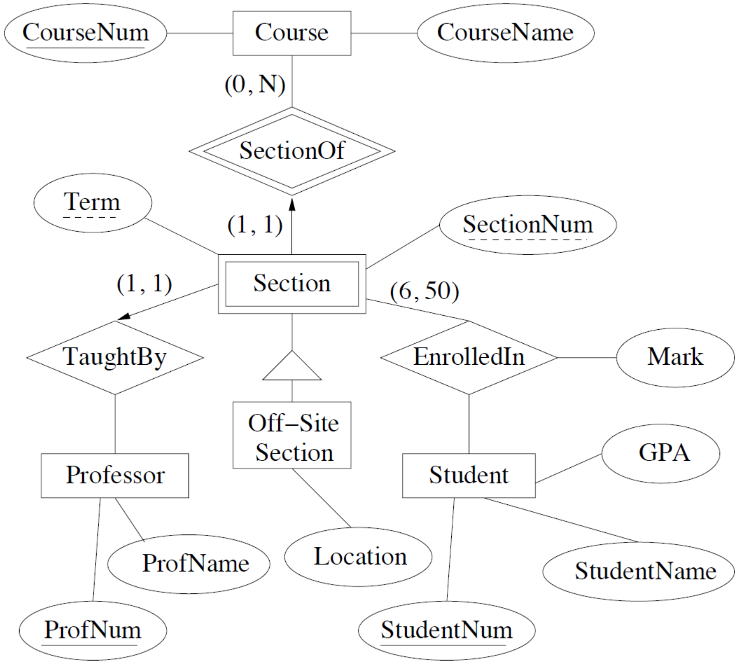


Relational Schema

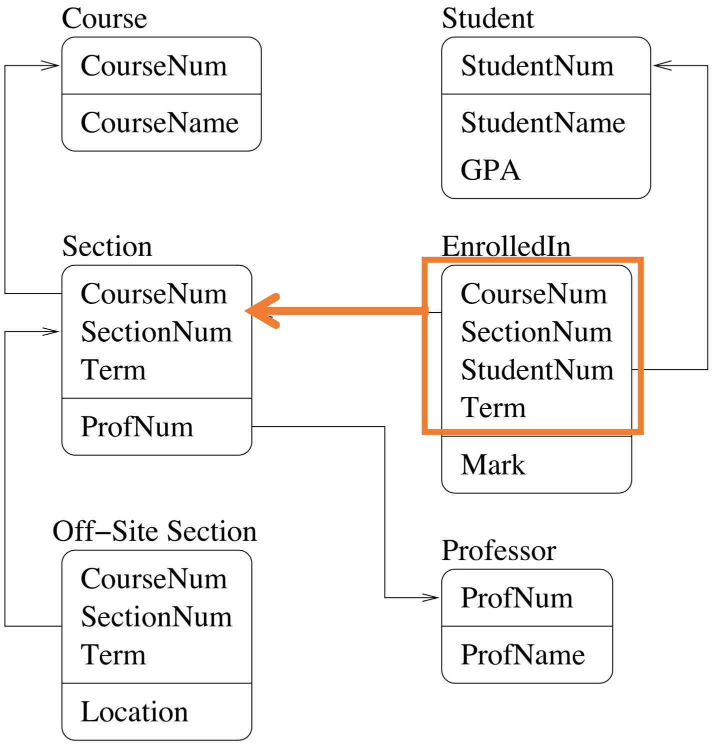
?

More examples

- ER Diagram



Relational Diagram


Course
CourseNum
CourseName

Student
StudentNum
StudentName
GPA

Section
CourseNum
SectionNum
Term
ProfNum

EnrolledIn
CourseNum
SectionNum
StudentNum
Term
Mark

Off−Site Section
CourseNum
SectionNum
Term
Location

Professor
ProfNum
ProfName

Design Theory (Lectures 9-10)

- Functional dependencies: provide clues towards elimination of (some) redundancies in a schema.
  - Closure of FDs (rules, e.g. Armstrong's axioms)
  - Compute attribute closure (1 algorithm + 2 uses)

- Schema decomposition
  - 2 properties for good schema decomposition
    - Property 1: Lossless join decompositions
    - Property 2: Dependency preserving decompositions
  - Normal forms based on FDs
    - BCNF → lossless join decompositions (1 algorithm)
    - 3rd NF → lossless join and dependency-preserving decompositions with more redundancy (2 algorithms)