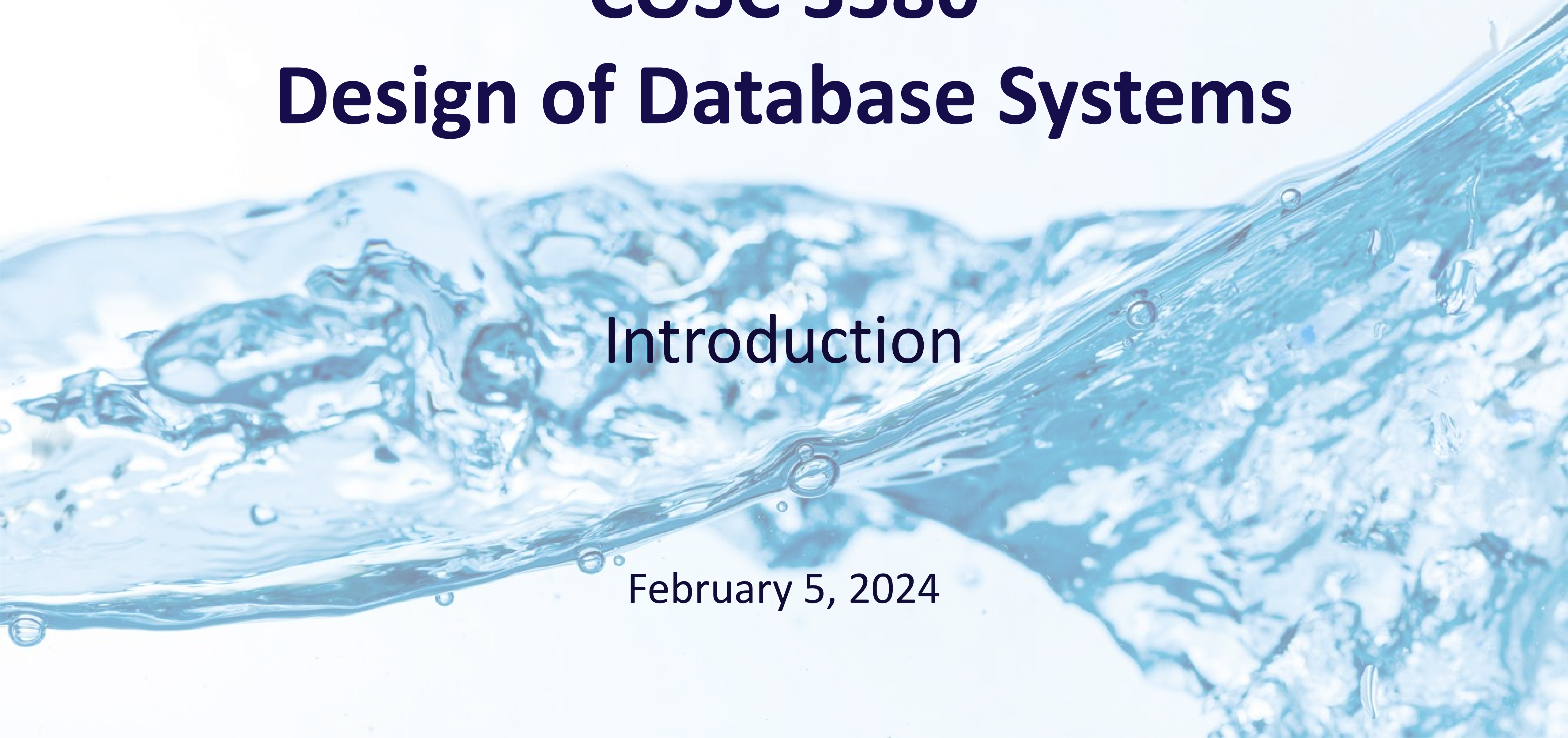
COSC 3380
Design of Database Systems



Introduction

February 5, 2024

Database Users

- Two groups of users
  - Actors on the Scene
  - Workers behind the Scene

Database Users – Actors behind the Scene

- **Tool Developers**
  - Design and implement software systems
    - Modeling and designing databases
    - Prototyping
    - Test data generation
    - User interface creation
    - Simulation
    - Performance monitoring
  - Facilitate building of applications and allow using database effectively

Database Users – Actors behind the Scene

- **System Designers and Implementors**
  - Design and implement DBMS packages in the form of modules and interfaces, test and debug them.
  - The DBMS must interface with applications, programming language compilers, operating system components, etc.
- **Operators and Maintenance Personnel**
  - They manage the actual running and maintenance of the database system hardware and software environment.

When not to use a DBMS

- Main inhibitors (costs) of using a DBMS:
  - High initial investment and possible need for additional hardware
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions

When not to use a DBMS

- When a DBMS may be unnecessary:
  - If the database and applications are simple, well defined, and not expected to change
  - If access to data by multiple users is not required

- When a DBMS may be infeasible:
  - In embedded systems where a general purpose DBMS may not fit in available storage

When not to use a DBMS

- When no DBMS may suffice:
  - If there are stringent real-time requirements that may not be met because of DBMS overhead (e.g., telephone switching systems)
  - If the database system is not able to handle the complexity of data because of modeling limitations (e.g., complex genome and protein databases)
  - If the database users need special operations not supported by the DBMS (e.g., GIS and location based services).

Database System
Concepts and Architecture

Data Models

- **Data Model**
  - A set of concepts to describe
    - the ***structure*** of a database
    - the ***operations*** for manipulating these structures
    - certain ***constraints*** that the database should obey
- **Data Model Structure and Constraints**
  - Constructs are used to define the database structure
  - Constructs typically include ***elements*** (and their ***data types***) as well as groups of elements (e.g. ***entity, record, table***), and ***relationships*** among such groups
  - Constraints specify some restrictions on valid data
    - These constraints must be enforced at all times

Data Models

- **Data Model Operations**
  - Operations used for specifying database *retrievals* and *updates* by referring to the constructs of the data model.
  - Operations on the data model
    - ***Basic model operations***
      - generic insert, delete, update
    - ***User-defined operations***
      - compute_student_gpa
      - update_inventory
      - notify_manager

Categories of Data Models

- **Conceptual (high-level, semantic) data models**
  - Provide concepts that are close to the way many users perceive data
    - ***Entity-based*** or ***object-based*** data models
- **Physical (low-level, internal) data models**
  - Provide concepts that describe details of how data is stored in the computer
  - Usually specified in an ad-hoc manner through DBMS design and administration manuals

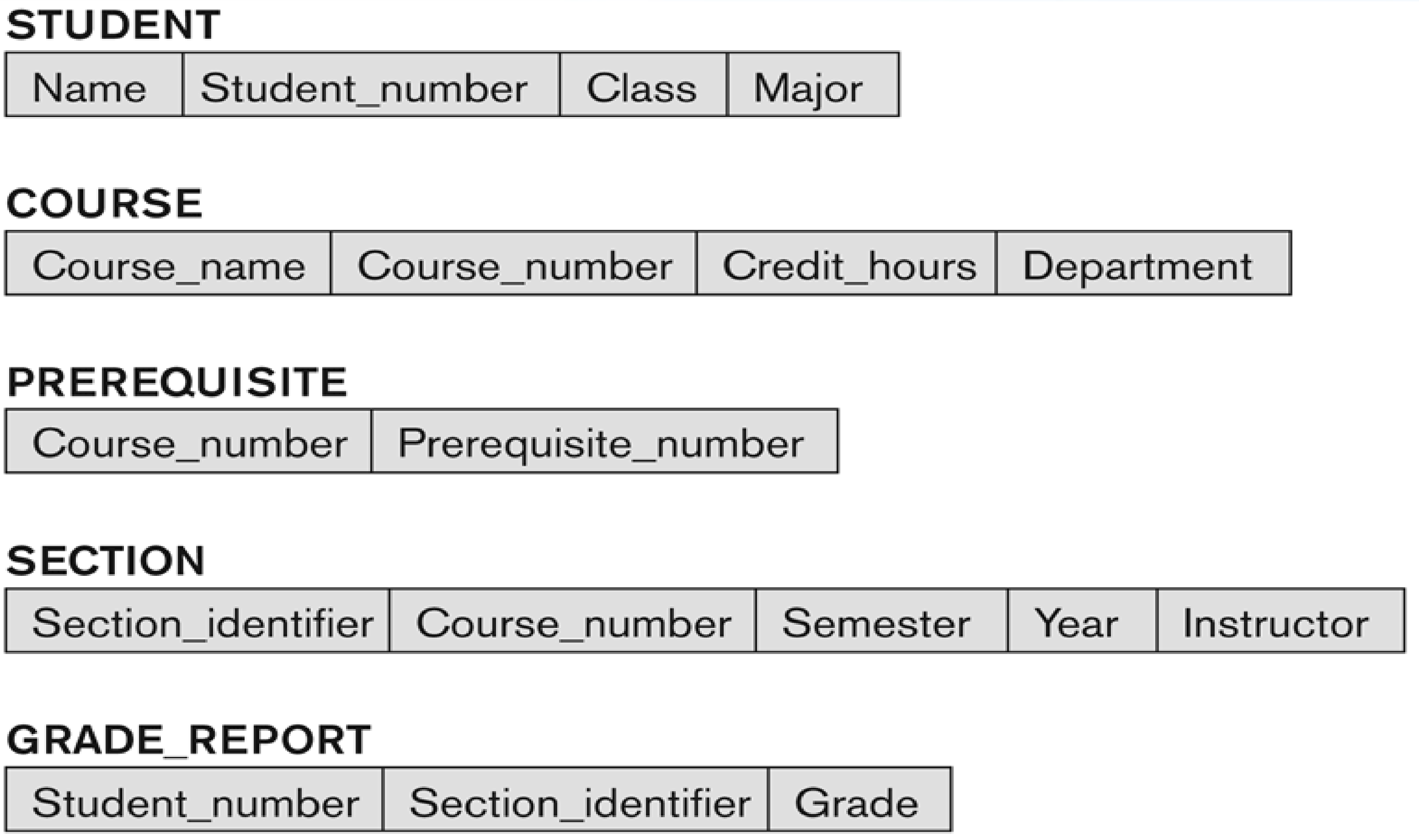
Categories of Data Models

- **Implementation (representational) data models**
  - Provide concepts that fall between conceptual and physical models
  - Used by many commercial DBMS implementations
  - Example: relational data models used in many commercial systems
- **Self-describing data models**
  - Combine the description of data with the data values
  - Examples include XML, key-value stores and NOSQL systems

Schema

- Database Schema
  - ***Description*** of a database
  - Includes descriptions of the database structure, data types, and the constraints on the database
- Schema Diagram
  - An ***illustrative*** display of (most aspects of) a database schema
- Schema Construct
  - A ***component*** of the schema or an object within the schema, e.g., STUDENT, COURSE

Example of a Database Schema



STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

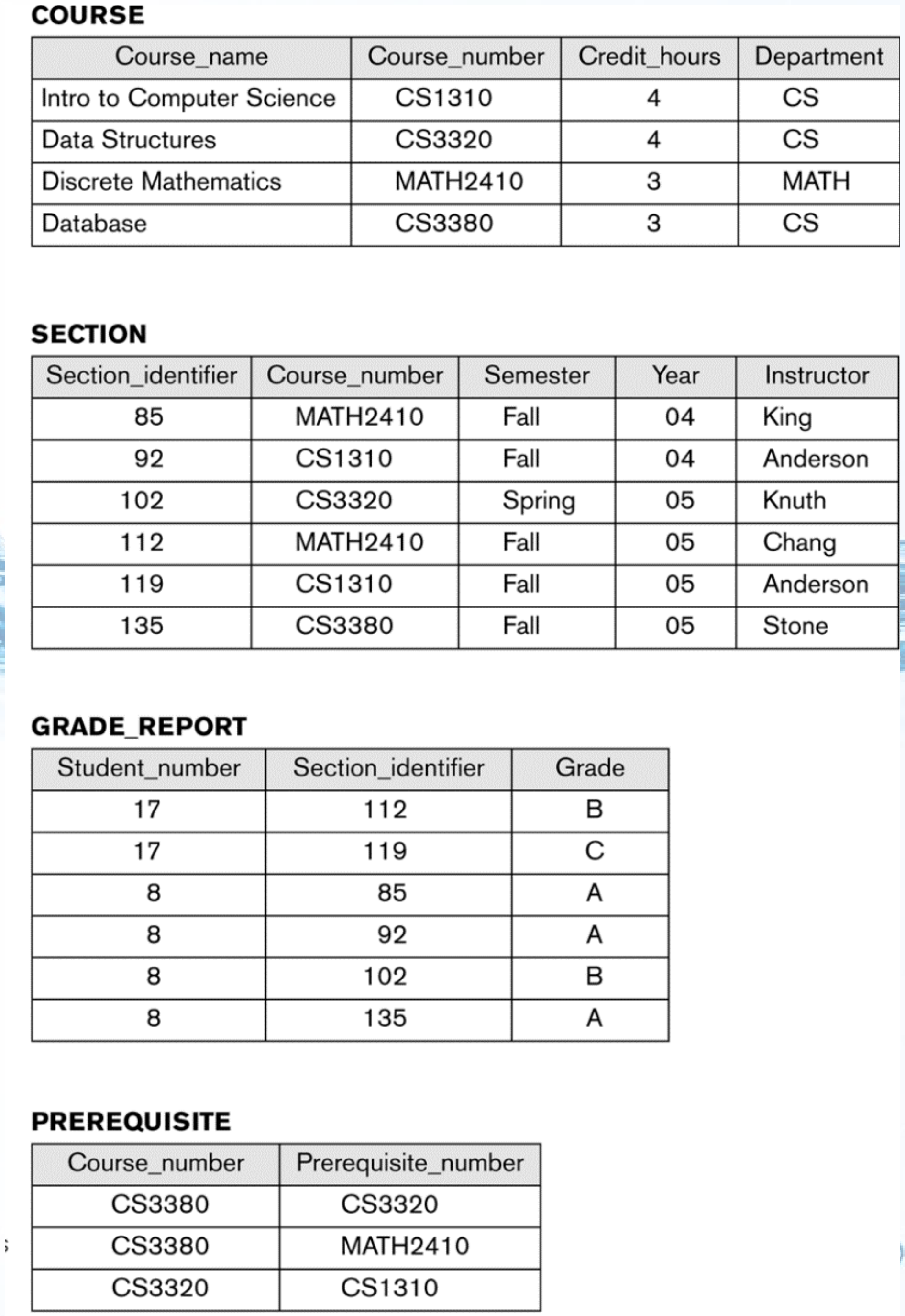
Instances

- Database State
  - The actual data stored in a database at a ***particular moment in time***
  - Includes the collection of all the data in the database
  - Also called database instance (or occurrence or snapshot)
    - The term ***instance*** is also applied to individual database components, e.g. *record instance, table instance, entity instance*

Database State

- **Database State**
  - Refers to the ***content*** of a database at a moment in time
- **Initial Database State**
  - Refers to the database state when it is initially loaded into the system
- **Valid State**
  - A state that satisfies the structure and constraints of the database

Example of a database state



COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Database Schema vs. Database State

- Distinction
  - The ***database schema*** changes very infrequently
  - The ***database state*** changes every time the data in the database are updated

- **Schema → intension**
- **State → extension**