Database Applications (15-415)

Course Overview and Introduction
Lecture 1, January 10, 2016

Mohammad Hammoud

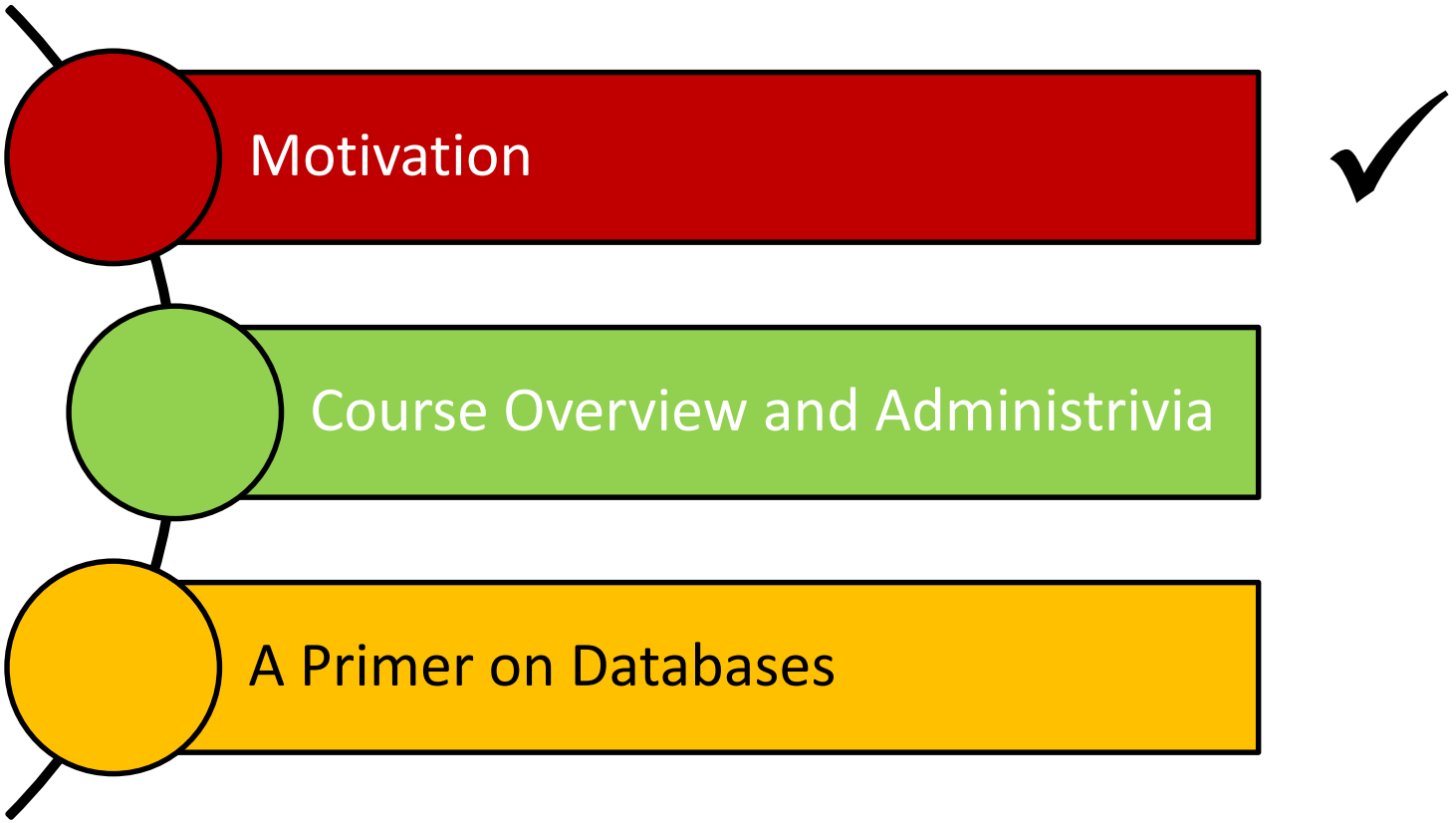
جامعة كارنيجي ميلون في قطر
Carnegie Mellon University Qatar

Today…

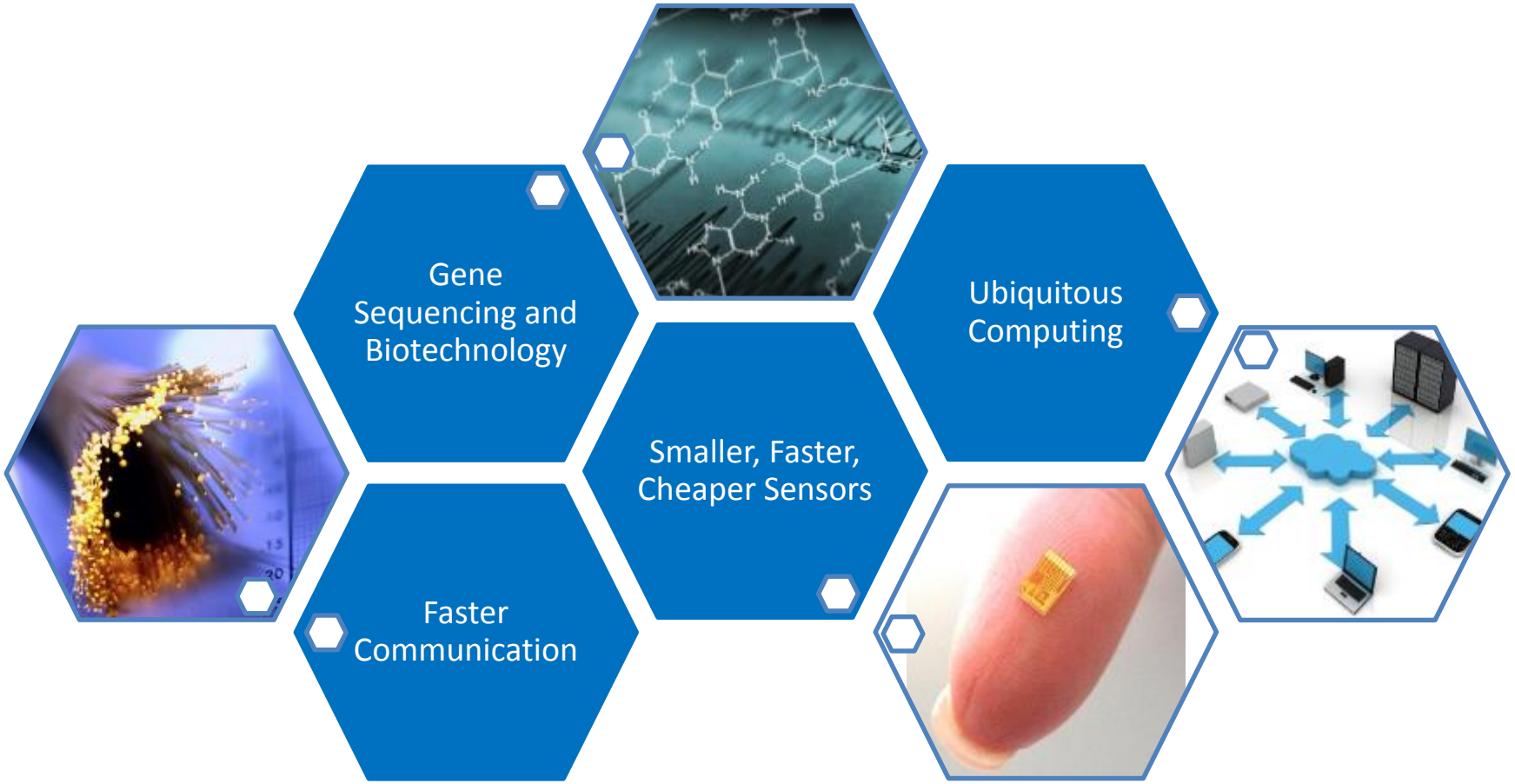
- Why databases and why studying databases?
- Course overview including objectives, topics and learning outcomes
- Administriva
- An introduction to databases and database systems

- Announcements:
  - Classes: Every Sunday and Tuesday from 4:30PM to 5:50PM in Room 1031
  - Recitations: “Every” Thursday from 4:30PM to 5:20PM in Room 1031
  - Course Webpage: http://www.qatar.cmu.edu/~mhhammou/15415-s16/
  - Materials: Syllabus, schedule, lectures, assignments, projects and announcements can always be found/checked on the course webpage

Outline

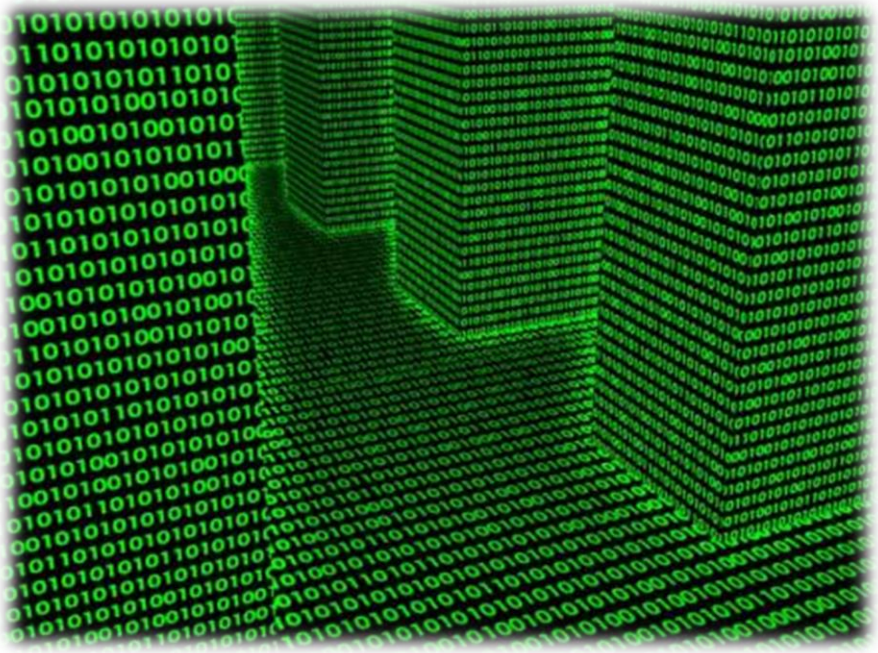


On the Verge of A Disruptive Century: Breakthroughs



A Common Theme is Data

The amount of data is only growing…
1.2 Zettabytes (1ZB = 10^{21} B or 1 Billion TB) in 2010



We Live in a World of Data

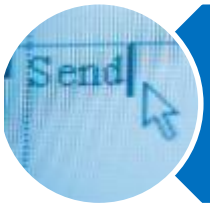
- Nearly 500 Exabytes per day are generated by the Large Hadron Collider experiments (not all recorded!)
- 2.9 million emails are sent every second
- 20 hours of video are uploaded to YouTube every minute
- 24 PBs of data are processed by Google every day
- 50 million tweets are generated per day
- 700 billion total minutes are spent on Facebook each month
- 72.9 items are ordered on Amazon every second

Data and *Big* Data

- The value of data as an organizational asset is widely recognized
- Data is literally exploding and is occurring along three main dimensions
  - "Volume" or the amount of data
  - "Velocity" or the speed of data
  - "Variety" or the range of data types and sources
- What is Big Data?
  - It is the proliferation of data that floods organizations on a daily basis
  - It is *high* volume, *high* velocity, and/or *high* variety information assets
  - It requires new forms of processing to enable *fast* mining, enhanced decision-making, insight discovery and process optimization

What Do We Do With Data and Big Data?

Store



Share

Query

Mine

Encrypt

…. and more!

We want to do these *seamlessly* and *fast*...

Using Diverse Interfaces & Devices

Computers

Mobile Devices

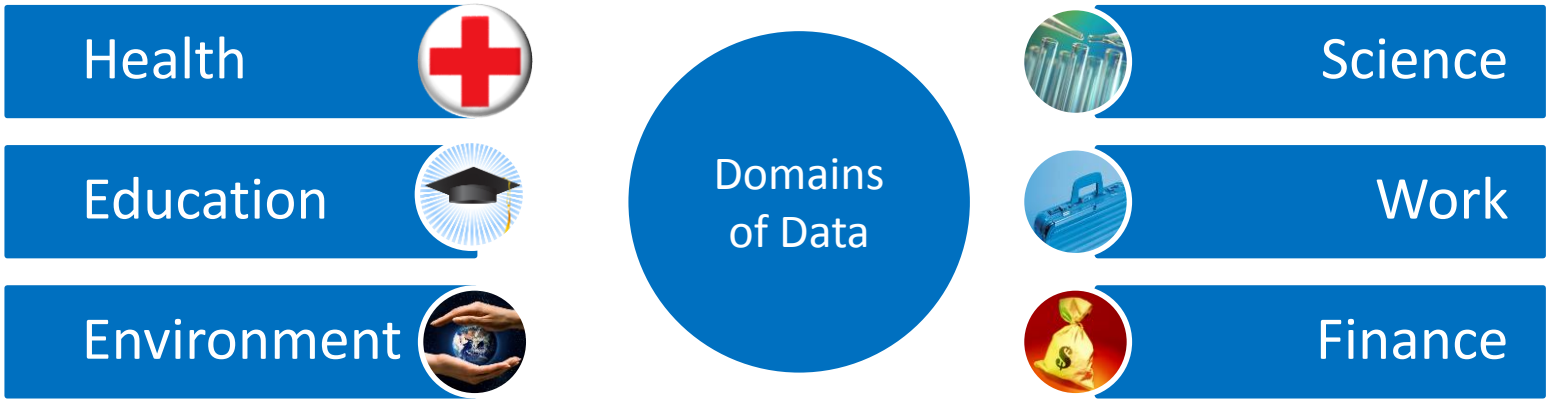
Consumer Electronics

Personal Monitors and Sensors

…and even appliances

We also want to access, share and process our data from all of our devices, **anytime, anywhere**!

Data is Becoming Critical to Our Lives

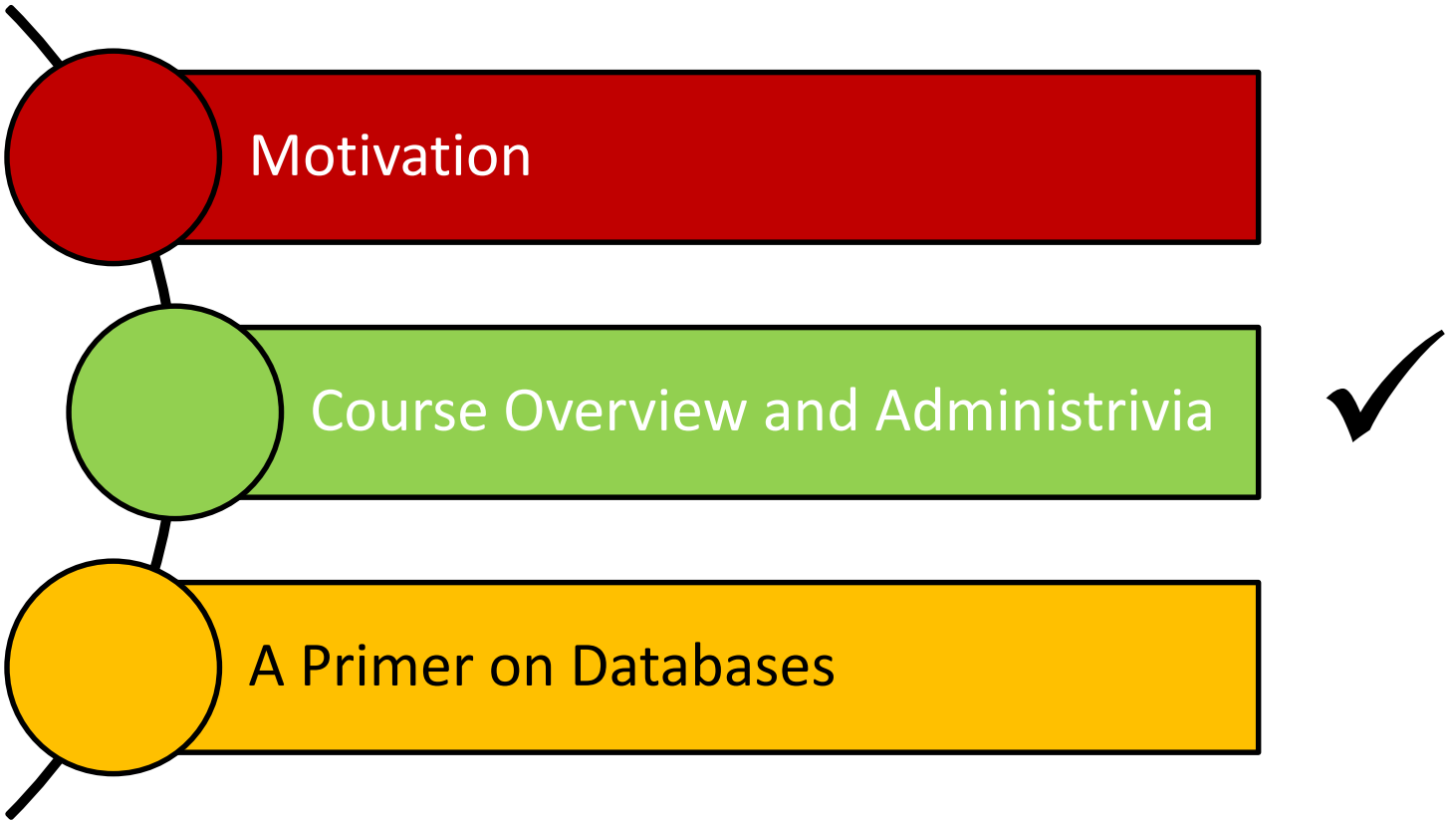


Why Studying Databases?

- Data is *everywhere* and is *critical* to our lives
- Data need to be recorded, maintained, accessed and manipulated *correctly*, *securely*, *efficiently* and *effectively*
  - At the "low end": scramble to web-scale (a mess!)
  - At the "high end": scientific applications
- Database management systems (DBMSs) are indispensable software for achieving such goals
- The principles and practices of DBMSs are now an integral part of computer science curricula
  - They encompass OS, languages, theory, AI, multimedia, and logic, among others

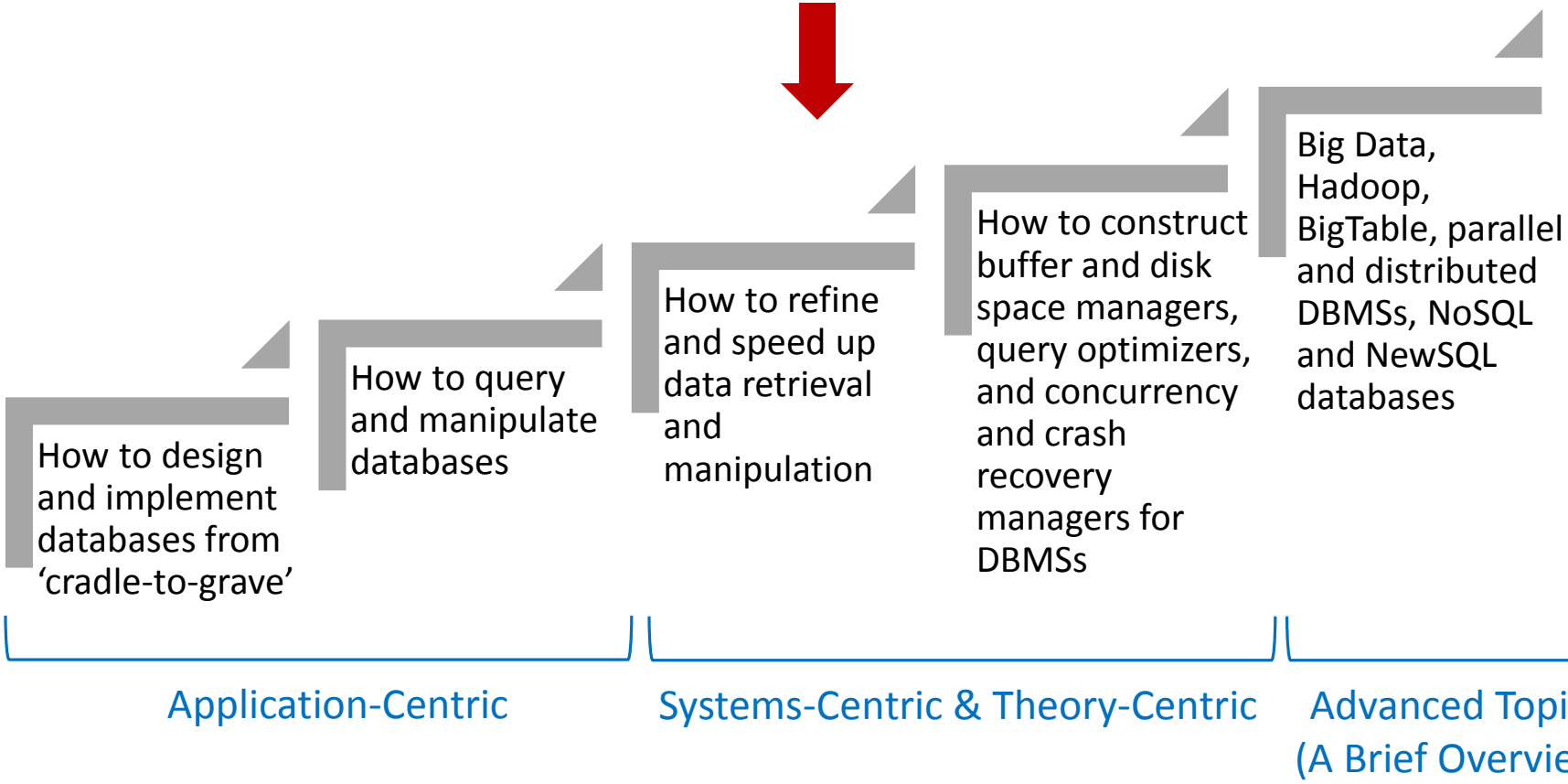
As such, the study of database systems can prove to be richly rewarding in more ways than one!

Outline



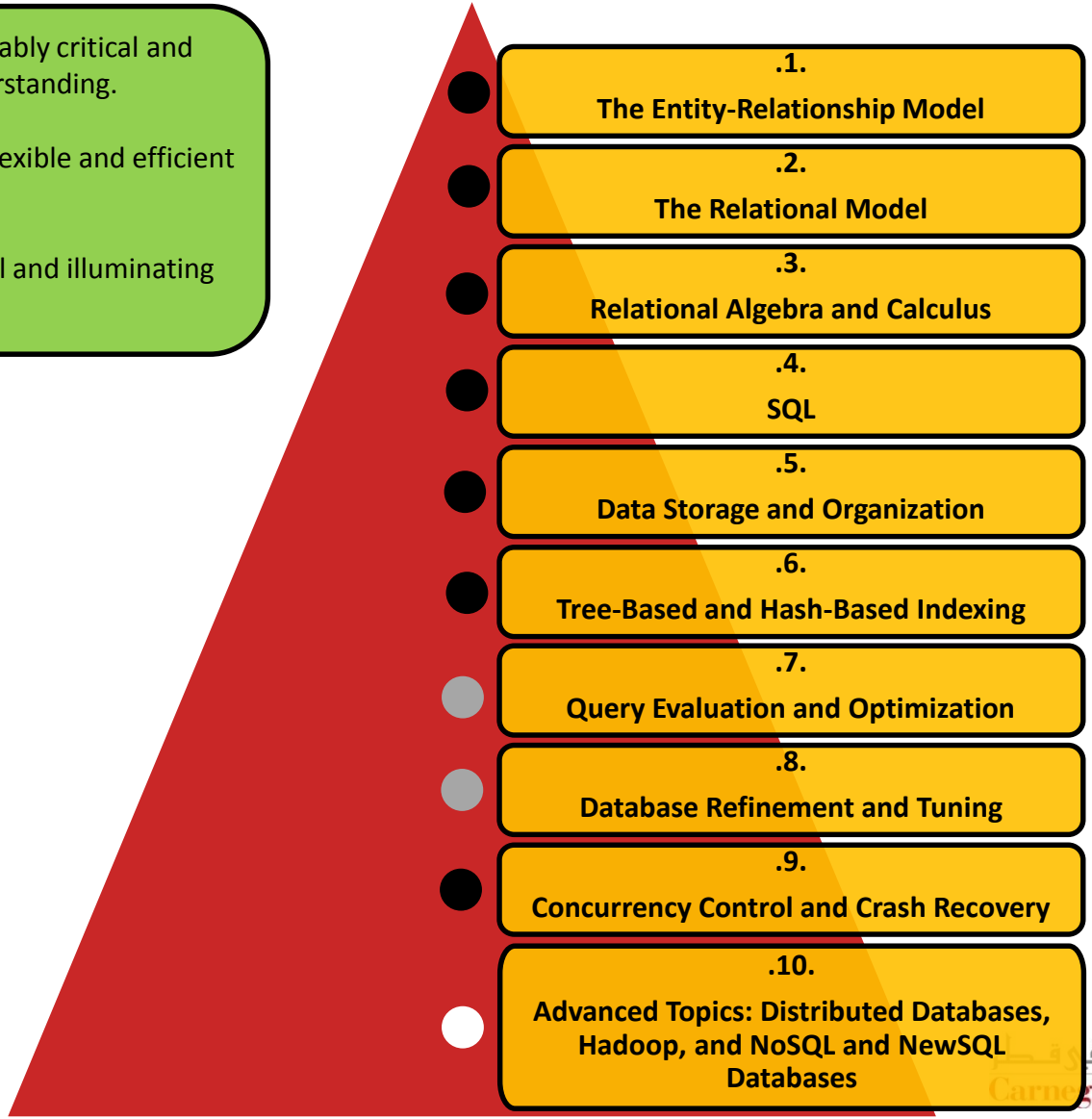
Course Objectives

In this course we aim at studying:



List of Topics

- **Considered:** a reasonably critical and comprehensive understanding.
- **Thoughtful:** Fluent, flexible and efficient understanding.
- **Masterful:** a powerful and illuminating understanding.



Learning Outcomes

After finishing this course you will be able to:

1. Describe a wide range of data involved in real-world organizations using the entity-relationship (ER) data model
2. Explain how to translate an ER diagram into a relational database
3. Analyze and apply two formal query languages, relational calculus and algebra
4. Indicate how SQL builds upon relational calculus and algebra and effectively apply SQL to create, query and manipulate relational databases
5. Design and develop multi-tiered, full-fledged standalone and web-based applications with back-end databases
6. Appreciate how DBMSs create, manipulate and manage files of fixed-length and variable-length records on disks
7. Create and operate various static and dynamic tree-based (e.g., ISAM and B+ trees) and hash-based (e.g., extendable and linear hashing) indexing schemes

Learning Outcomes

After finishing this course you will be able to:

8. Explain and evaluate various algorithms for relational operations (e.g., join) using techniques such as iteration, indexing and partitioning

9. Analyze and apply different query evaluation plans and describe the various tasks of a typical relational query optimizer

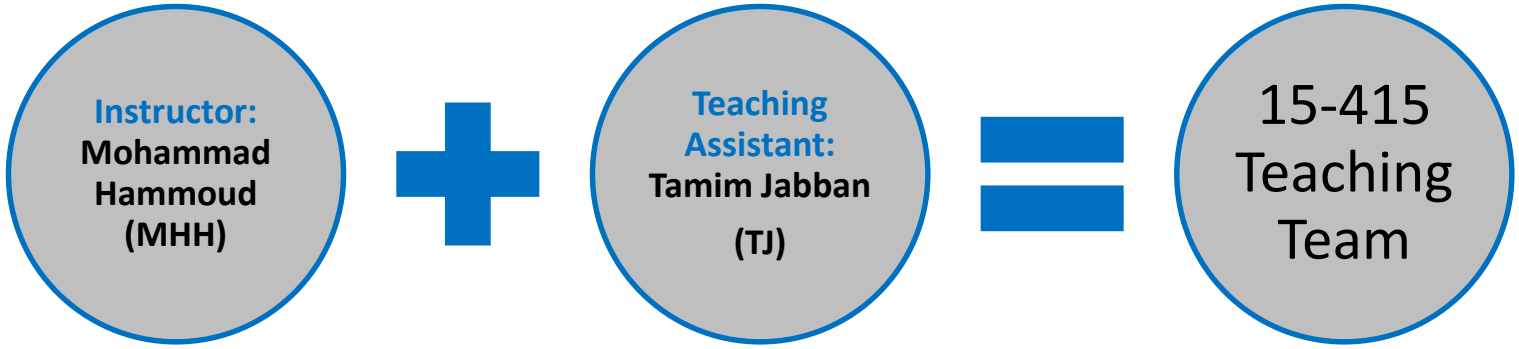
10. Explain how conceptual schemas can be refined using the theory of functional dependencies and techniques like decomposition and synthesis

11. Describe how transactions can be interleaved correctly, and indicate how a DBMS can ensure atomicity and durability when systems fail or entirely crash

12. Identify alternative architectures for distributed databases, and describe how data can be partitioned and distributed across networked nodes of a DBMS

13. Appreciate the scale of Big Data, discuss some popular analytics engines for Big Data processing and denote the applicability of NoSQL databases for Big Data storage

Teaching Team



MHH Office Hours

- Wednesday, 4:30- 5:30PM
- Welcome when my office door is open
- By appointment

TJ Office Hours

- Tuesday, 9:30- 11:59AM & Thursday, 10:30- 11:59AM
- Welcome when his office door is open
- By appointment

Teaching Methods, Assignments and Projects

26 Lectures

- Motivate learning
- Provide a framework or roadmap to organize the information of the course
- Explain subjects and reinforce the critical big ideas

14 Recitations

- Get you to reveal what you do not understand, so we can help you
- Allow you to practice skills you will need to become an expert

5 Assignments

- We will have 5 assignments which involve problem solving and span most of the topics that we discuss in the class

3 Projects

- We will have 3 projects which involve using Postgres, SQL, C, and Java

Some Rules on the Projects

- For all the projects (*except the final one*), the following rules apply:
  - If you submit one day late, 25% will be deducted from your project score
  - If you are two days late, 50% will be deducted
  - The project will not be graded (and you will receive a zero score) if you submit more than two days late
  - There will be a 3-grace-day quota

Assessment Methods

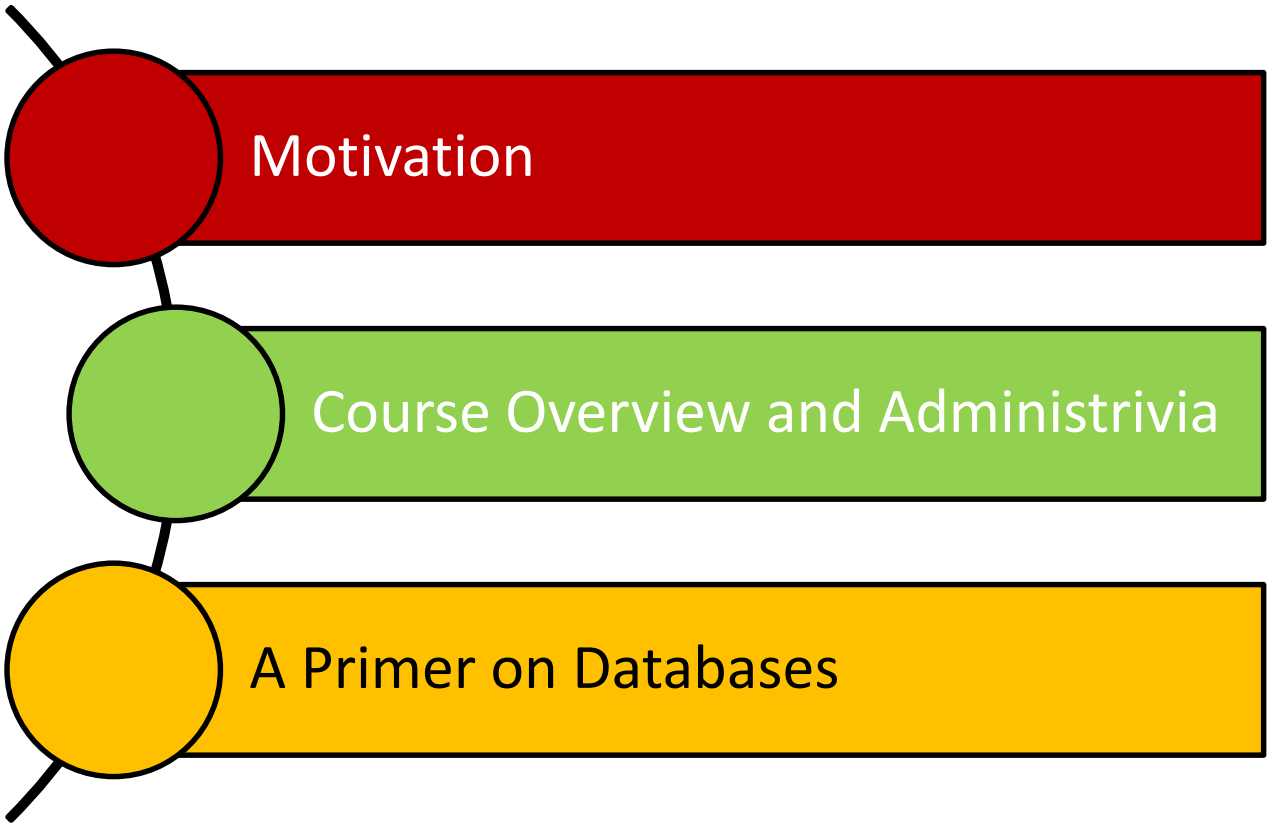
- How do we measure learning?

Type	#	Weight
Projects	3	35%
Exams	2	30%
Problem Solving Assignments	5	20%
Quizzes	2	10%
Class/Recitation Participation and Attendance	40	5%

Target Audience, Prerequisites and Textbook

- Target Audience:
  - Juniors and Seniors
- Prerequisites:
  - 15-121 and 15-213
  - Students should have a basic knowledge of data structures, algorithms, computer systems and programming languages like C, C++ and Java
- Textbook:
  - Raghu Ramakrishnan and Johannes Gehrke, *"Database Management Systems"*, Third Edition, McGraw-Hill, 2002

Outline



A Motivating Scenario

- Qatar Foundation (QF) has a large collection of data (say 500GB) on employees, students, universities, research centers, etc.,
- This data is accessed by multiple users and applications
- Queries on data must be executed quickly
- Changes made to the data by different users must be applied *consistently*
- Access to certain parts of data (e.g., salaries) must be *restricted*
- This data should survive system crashes and failures

Performance (Concurrency Control)

Performance (Response Time)

Correctness (Consistency)

Correctness (Security)

Correctness (Durability and Atomicity)

Managing Data using File Systems

- What about managing QF data using local file systems?
  - Files of fixed-length and variable-length records as well as formats
  - Main memory vs. disk
  - Computer systems with 32-bit addressing vs. 64-bit addressing schemes
  - Special programs (e.g., C++ and Java programs) for answering user questions
  - Special measures to maintain atomicity
  - Special measures to maintain consistency of data
  - Special measures to maintain data isolation
  - Special measures to offer software and hardware fault-tolerance
  - Special measures to enforce security policies in which different users are granted different permissions to access diverse subsets of data

This becomes tedious and inconvenient, especially at large-scale, with evolving/new user queries and higher probability of failures!

Data Base Management Systems

- A special software is accordingly needed to make the preceding tasks easier
- This software is known as Data Base Management System (DBMS)
- DBMSs provide automatic:
  - Data independence
  - Efficient data access
  - Data integrity and security
  - Data administration
  - Concurrent access and crash recovery
  - Reduced application development and tuning time

Some Definitions

- A database is a collection of data which describes one or many real-world enterprises
  - E.g., a university database might contain information about entities like students and courses, and relationships like a student enrollment in a course
- A DBMS is a software package designed to store and manage databases
  - E.g., DB2, Oracle, MS SQL Server, MySQL and Postgres
- A database system = (Big) Data + DBMS + Application Programs

Data Models

- The user of a DBMS is ultimately concerned with some real-world enterprises (e.g., a University)
- The data to be stored and managed by a DBMS *describes* various aspects of the enterprises
  - E.g., The data in a university database describes students, faculty and courses entities and the relationships among them
- A data model is a collection of high-level data description constructs that hide many low-level storage details
- A widely used data model called the entity-relationship (ER) model allows users to pictorially denote entities and the relationships among them

The Relational Model

- The relational model of data is one of the most widely used models today
- The central data description construct in the relational model is the relation
- A relation is basically a table (or a set) with rows (or records or tuples) and columns (or fields or attributes)
- Every relation has a schema, which describes the columns of a relation
- Conditions that records in a relation must satisfy can be specified
  - These are referred to as integrity constraints

The Relational Model: An Example

- Let us consider the student entity in a university database

Students Schema

Students(*sid*: string, *name*: string, *login*: string, *dob*: string, *gpa*: real)

An attribute, field or column

Integrity Constraint: Every student has a unique *sid* value

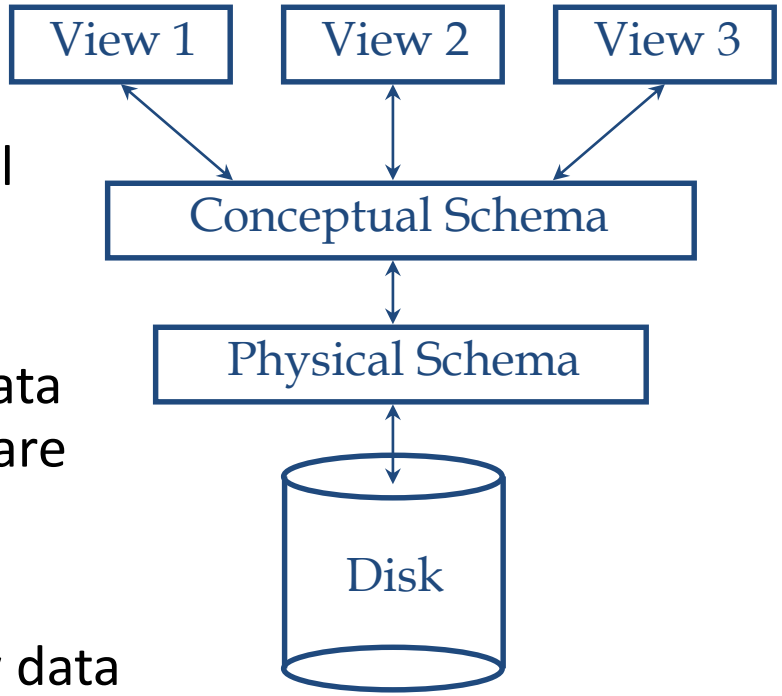
A record, tuple or row

sid	*name*	*login*	*dob*	*gpa*
512412	Khaled	khaled@qatar.cmu.edu	18-9-1995	3.5
512311	Jones	jones@qatar.cmu.edu	1-12-1994	3.2
512111	Maria	maria@qatar.cmu.edu	3-8-1995	3.85

An *instance* of a Students *relation*

Levels of Abstraction

- The data in a DBMS is described at three levels of abstraction, the conceptual (or logical), physical and external schemas



- The conceptual schema describes data in terms of a specific data model (e.g., the relational model of data)
- The physical schema specifies how data described in the conceptual schema are stored on secondary storage devices
- The external schema (or views) allow data access to be customized at the level of individual users or group of users (views can be 1 or many)

Views

- A view is conceptually a relation
- Records in a view are computed as needed and usually not stored in a DBMS
- Example: University Database

Conceptual Schema	Physical Schema	External Schema (View)
• Students(sid: string, name: string, login: string, dob: string, gpa:real) • Courses(cid: string, cname:string, credits:integer) • Enrolled(sid:string, cid:string, grade:string)	• Relations stored as heap files • Index on first column of Students	Students can be allowed to find out course enrollments: • Course_info(cid: string, enrollment: integer)

Can be computed from the relations in the conceptual schema (so as to avoid data redundancy and inconsistency).

Iterating: Data Independence

- One of the most important benefits of using a DBMS is data independence

- With data independence, application programs are insulated from how data are structured and stored

- Data independence entails two properties:
  - Logical data independence: users are shielded from changes in the conceptual schema (e.g., add/drop a column in a table)
  - Physical data independence: users are shielded from changes in the physical schema (e.g., add index or change record order)

Queries in a DBMS

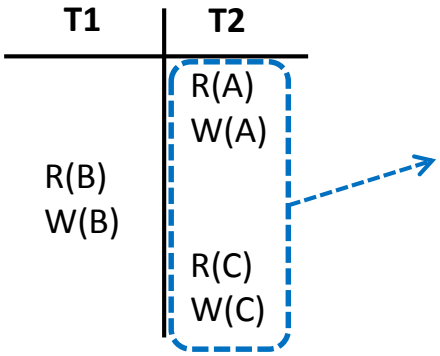
- The ease with which information can be queried from a database determines its value to users

- A DBMS provides a specialized language, called the query language, in which queries can be posed

- The relational model supports powerful query languages
  - Relational calculus: a formal language based on mathematical logic
  - Relational algebra: a formal language based on a collection of operators (e.g., selection and projection) for manipulating relations
  - Structured Query Language (SQL):
    - Builds upon relational calculus and algebra
    - Allows creating, manipulating and querying relational databases
    - Can be embedded within a host language (e.g., Java)

Concurrent Execution and Transactions

- An important task of a DBMS is to *schedule* concurrent accesses to data so as to improve performance



An *atomic* sequence of database actions (read/writes) is referred to as "*transaction*"

- When several users access a database *concurrently*, the DBMS must order their requests carefully to avoid conflicts
  - E.g., A check might be cleared while account balance is being computed!

- DBMS ensures that conflicts do not arise via using a locking protocol
  - Shared vs. Exclusive locks

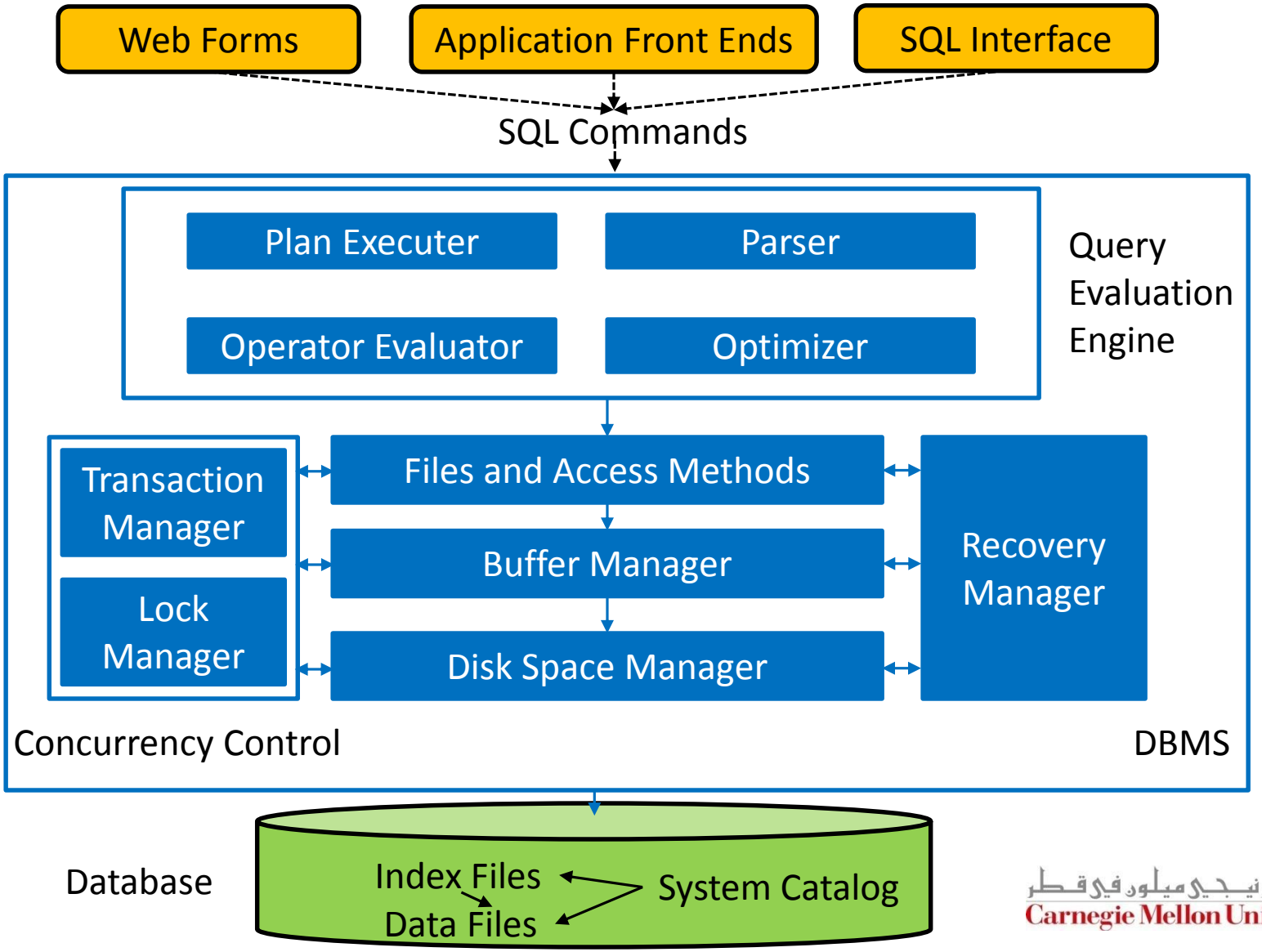
Ensuring Atomicity

- Transactions can be interrupted before running to completion for a variety of reasons (e.g., due to a system crash)

- DBMS ensures atomicity (all-or-nothing property) even if a crash occurs in the middle of a transaction

- This is achieved via maintaining a log (i.e., history) of all writes to the database
  - *Before* a change is made to the database, the corresponding log entry is forced to a safe location (this protocol is called Write-Ahead Log or WAL)
  - After a crash, the effects of partially executed transactions are *undone* using the log

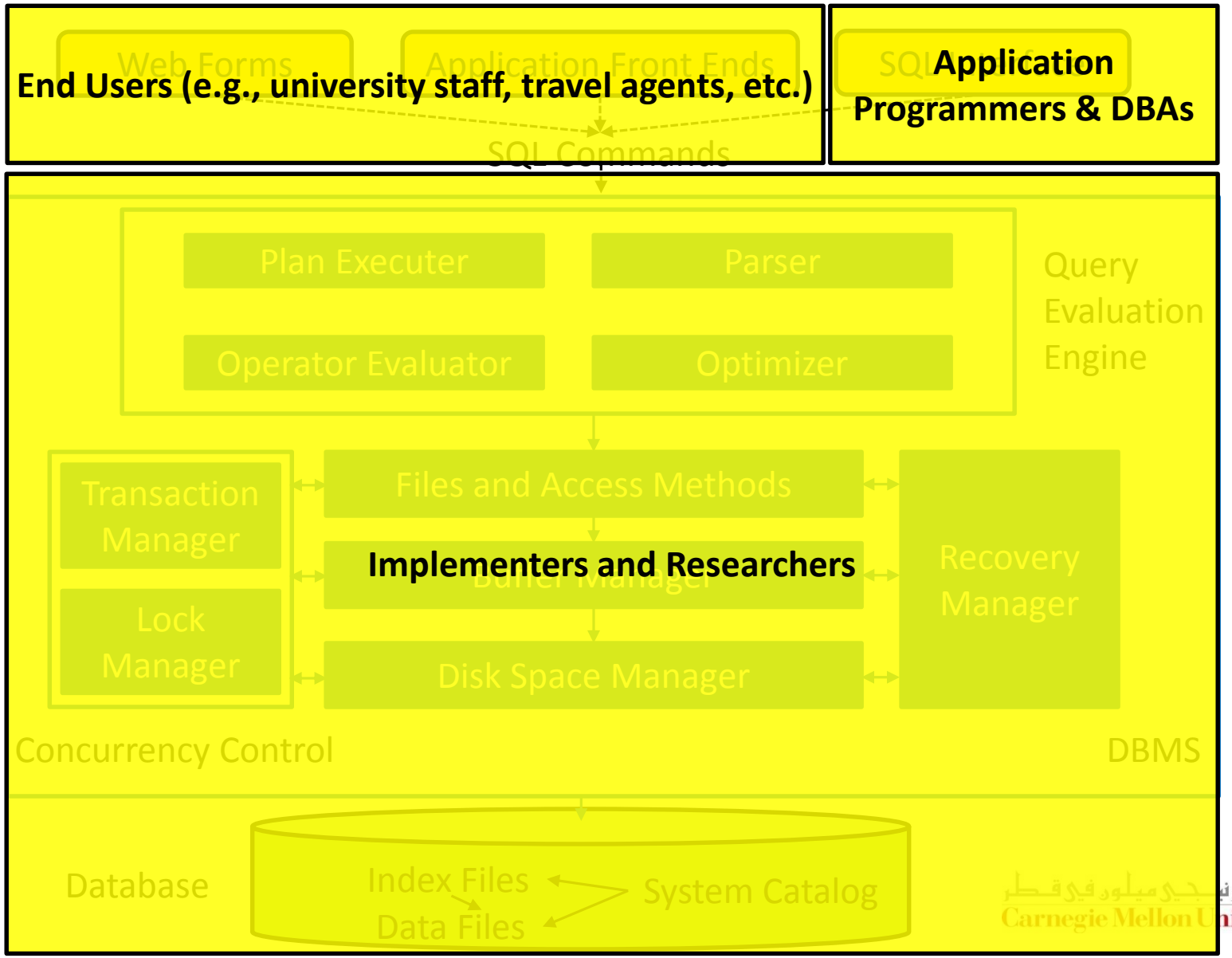
The Architecture of a Relational DBMS



People Who Work With Databases

- There are five classes of people associated with databases:
  1. End users
     - Store and use data in DBMSs
     - Usually not computer professionals
  2. Application programmers
     - Develop applications that facilitate the usage of DBMSs for end-users
     - Computer professionals who know how to leverage host languages, query languages and DBMSs altogether
  3. Database Administrators (DBAs)
     - Design the conceptual and physical schemas
     - Ensure security and authorization
     - Ensure data availability and recovery from failures
     - Perform database tuning
  4. Implementers
     - Build DBMS software for vendors like IBM and Oracle
     - Computer professionals who know how to build DBMS internals
  5. Researchers
     - Innovate new ideas which address evolving and new challenges/problems

The Architecture of a Relational DBMS



Summary

- We live in a world of data
- The explosion of data is occurring along the 3Vs dimensions
- DBMSs are needed for ensuring logical and physical data independence and ACID properties, among others
- The data in a DBMS is described at three levels of abstraction
- A DBMS typically has a layered architecture

Summary

- Studying DBMSs is one of the broadest and most exciting areas in computer science!

- This course provides an in-depth treatment of DBMSs with an emphasis on how to *design*, *create*, *refine*, *use* and *build* DBMSs and real-world enterprise databases

- Various classes of people who work with databases hold responsible jobs and are well-paid!