COMP9120

Database Management Systems

Semester 1, 2025

Part A: Unit outline and induction

Professor Athman Bouguettaya
School of Computer Science

THE UNIVERSITY OF SYDNEY

Content co-developed by Dr Lijun Chang

Who is your (bearded) lecturer?

- **Athman Bouguettaya**

  Professor, School of Computer Science

- Teaching this unit for the 7th time.
  - On the *Dean's Teaching Commendation list* for the past 3 years!
- Research Interest
  - Service Computing, Internet of Things (IoT), Cloud Computing

For further information about what I do besides teaching, please go to **http://scslab.net**

Get to know you (the students)!

Let’s menti!

Acknowledgement of Country

I would like to acknowledge the Traditional Owners of Australia and recognise their continuing connection to land, water and culture. I am currently on the land of the Gadigal people of the Eora nation and pay my respects to their Elders, past, present and emerging.

Copyright warning

COMMONWEALTH OF AUSTRALIA

Copyright Regulations 1969

WARNING

This material has been reproduced and communicated to you by or on behalf of the University of Sydney pursuant to Part VB of the Copyright Act 1968 (**the Act**).

The material in this communication may be subject to copyright under the Act. Any further copying or communication of this material by you may be the subject of copyright protection under the Act.

Do not remove this notice.

Teaching Team

- Lecturer/Coordinator: Professor **Athman Bouguettaya**
- **Lecture location**: F19.02.201.Eastern Avenue Auditorium and Theatre Complex.Eastern Avenue Auditorium
- Lecture: Thursday 5-7 PM
- Tutors:
  - Abbey Lin (TA)
  - Dipankar Chaki
  - Qixuan Hu
  - Shruti Panday
  - Vinit Iyer
  - Fangyu (Tommy) Zhou
  - Thilina Lokuruge
  - Mehnaz Tabassum
  - Muhammad Umair
  - Tina Sheng
  - Hengzhi Chen
  - Yan Rong
  - Harshita Balakumar
  - Zhizhao Zhang
- Tutorials: Thursday 7-8 PM, 8-9 PM,

  Friday 5-6 PM, 6-7 PM, 7-8 PM and 8-9 PM

Platforms used

- Canvas
  - Lecture slides, tutorial sheets, lecture recordings
  - Assignment submissions
  - Publish grades

- Ed
  - Discussion Forum
  - SQL practice

What this unit of study is (and is not)

We will cover **how database management systems work.** We will discuss the:

- **Design**, **query**, **efficient access** and **storage** of databases.
- **Theoretical** aspects underpinning **relational databases**

› This UoS is ***not*** an SQL programming course, although we will be using SQL quite extensively!

› Note the following:

Unit code	COMP9120
Prohibitions ?	INFO2120 OR INFO2820 OR INFO2005 OR INFO2905 OR COMP5138 OR ISYS2120. Students who have previously studied an introductory database subject as part of their undergraduate degree should not enrol in this foundational unit, as it covers the same foundational content
Assumed knowledge ?	Some exposure to programming and some familiarity with data model concepts

Outline of Lectures

	Week	Topic
Database Design and Query	Week 1	Introduction
	Week 2	Conceptual Database Design
	Week 3	Relational Data Model / Logical Database Design
	Week 4	Relational Algebra and SQL
	Week 5	Advanced SQL
	Week 6	Database Integrity
	Week 7	Database Application Development and Security
	Week 8	Schema Refinement and Normalisation
Database Internals	Week 9	Transaction Management
	Week 10	Quiz
	Week 11	Storage and Indexing
	Week 12	Query Evaluation and Optimisation
	Week 13	Review

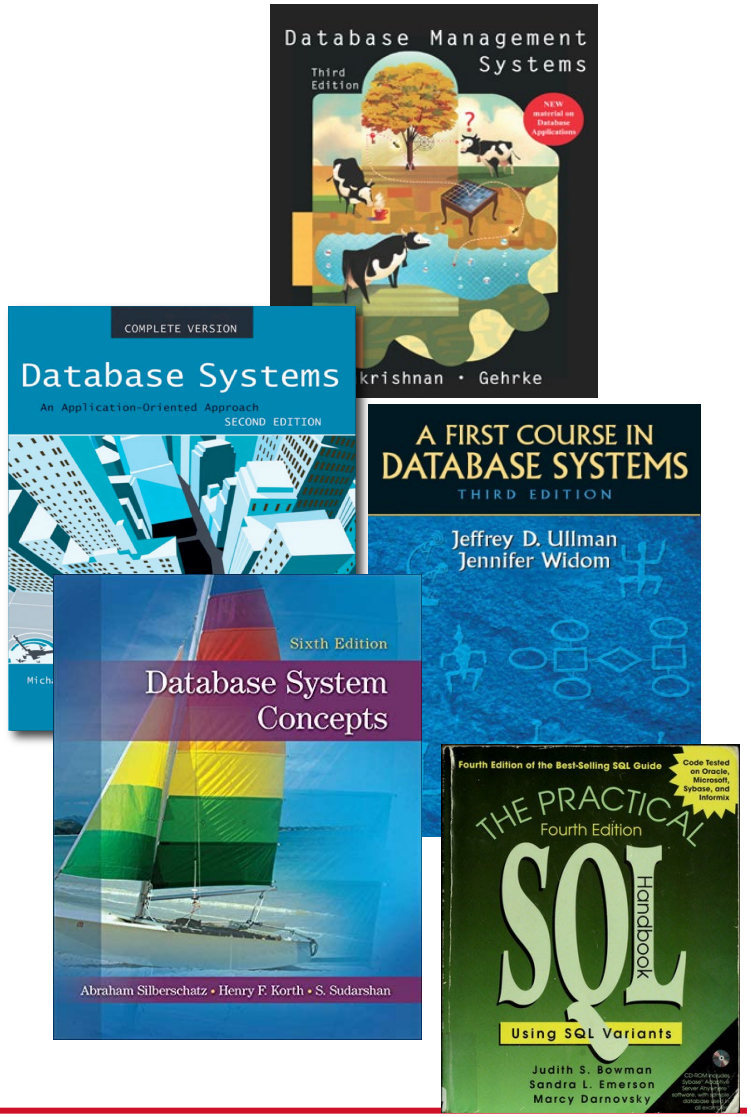
Textbooks

Click on Reading List in Canvas

- R. Ramakrishnan and I. Gehrke
  - ***Database Management Systems***
- A. Silberschatz, H. Korth, S Sudarshan
  - ***Database System Concepts***
- J.D. Ullman, and J.Widom:
  - ***A First Course in Database Systems***
- M. Kifer, A. Bernstein, and P.M. Lewis
  - ***Database Systems: An Application-Oriented Approach***

Suggested additional **SQL reference**:

- Judith S. Bowman, Sandra L. Emerson, and Marcy Darnovsky
  - ***The Practical SQL***



THE UNIVERSITY OF SYDNEY

Software

- We will be using the following software in labs & assignments:
  - PostgreSQL (server) and pgadmin (client)
  - Java/Python

How Much Programming is Involved?

- Although you do not need to be a highly experienced programmer, you will need some *minimum programming skills* for the practical assignments:
  - As previously noted, this UoS is ***not*** a programming course,
  - It covers database design, creation and usage

- The *DB application programming* assignment (**Assignment 2**) assumes some programming experience and expects some competence in **Java or Python**!
  - It is **your responsibility** to **learn Java/Python**

Assessments

- Assessment tasks:

  1. Two (2) **Group Assignments**

     - *DB Schema Design* **16%**
       - Released on **Week 3** and **due** on **Week 6** (Sunday **6 April** 11:59pm)
     - *DB Usage* **16%**
       - Released on **Week 7** and due on **Week 11** (Sunday **18 May** 11:59pm)

  2. **Quiz** (Week 10) (Thursday **8 May** 5:30pm-7:00pm) **18%**

  3. Final Exam (Exam Period) **50%**

- You must obtain ≥ **40% in the final exam** and an **overall mark of** ≥**50%**, to pass the unit

  For example:

  *Progressive Mark **44%**, Exam Mark **50%**, total **47%**: **Fail***
  *Progressive Mark **75%**, Exam Mark **35%**, total **55%**: **Fail***

Forming Groups

- All the assignments are **group assignments**.
  - Each group should consist of 3 members. Please find others to form a group *as soon as possible*.
  - First assignment is released on Week 3, *groups must be formed by Week 2*.
    - *At the start of week 3, we will randomly create groups of unassigned students*.
  - To ensure the **contribution** of each member, each one of you will *assess* the *contribution* of the *other teammates*. In addition, we **will conduct random interviews to** ensure fairness.

- Each group must organize itself:
  - Arrange internal communication, meetings and upload meeting minutes in Canvas
  - Set internal deadlines
  - Follow up if deadlines are missed
  - Have a fallback plan
  - Have a dispute resolution mechanism in place

- Let your unit coordinator/lecturer know asap if there are issues (unresolvable disagreements, member who doesn't contribute, etc)

Late Submission

- Late submission will incur **a penalty of 5% per day**.
  - An assignment that would normally get 9/10 and is 2 days late loses 10% of the full 10 marks, i.e., new mark =
  - An average assignment that would normally get 5/10 and is 5 days late loses 25% of the full 10 marks, i.e., new mark =
- Assessments more than **10 days late get 0 marks.**
- Warning: submission site gets very slow near deadlines
- Submit early; you can resubmit before the deadline

Special Consideration (University policy)

- If your performance on assessments is affected by illness or misadventure
  - Follow proper university procedures
    - Have professional practitioner sign special USyd form
    - Submit application for **special consideration** online, upload scans
    - Note you have only **3 working days** for applying
    - http://sydney.edu.au/current_students/special_consideration/
- Also, notify coordinator by email *as soon as anything begins to go wrong*
- There is a similar process if you need special arrangements, e.g., for religious observance, military service, representative sports, etc.

Work, Health and Safety (WHS) Induction

School of Computer Science

THE UNIVERSITY OF SYDNEY

Staying safe in the event of an emergency

- In the unlikely event of an emergency, we may need to evacuate the building.
- If we need to evacuate, please take your belongings and follow the green exit signs.
- We will move a safe distance from the building while waiting until the emergency is over.
- In some circumstances, we might be asked to remain inside the building for our own safety. We call this a lockdown or shelter-in-place.
- More information is available at www.sydney.edu.au/about-us/campuses/emergencies-and-personal-safety.html

Supporting your health and wellbeing

Student life, wellbeing and support page

Everything you need to know about the student services, resources and events available to support you while you study, including peer support, counselling, after hours support, and learning support.

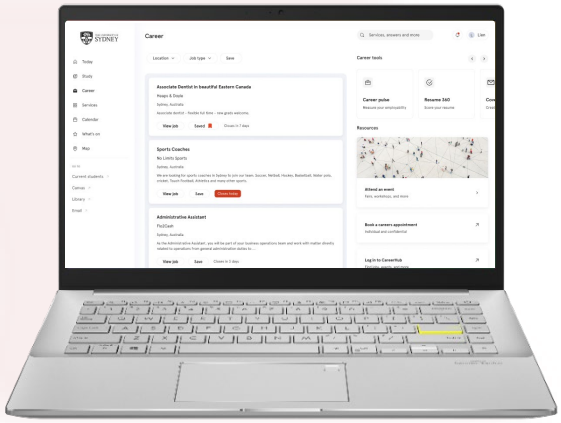
https://www.sydney.edu.au/students/support.html

Safer Communities Office

Support for people who have experienced sexual misconduct, domestic/family violence, bullying/harassment or issues relating to modern slavery

www.sydney.edu.au/students/health-wellbeing/safer-communities.html

Accessing tools and support through the Student Portal



Experience the University of Sydney's enhanced Student Portal where you can:

- View your integrated **Student calendar** including student Outlook calendar, class timetable, and events.
- Explore available job opportunities in the improved **Career section** and have access to exciting new tools like the Resume Builder.
- Access University services via the new **Study and Support** page.

myuni.sydney.edu.au

Understanding your academic integrity responsibilities

- Academic integrity refers to behaving honestly, ethically and responsibly in relation to all elements of your study at the University, including assessments.
- Always submit your own work, sit your own tests, and take your own examinations.
- Acknowledge any contributions in your assignment which are not your original thoughts, ideas or words.
- **Academic Honesty Education Module** – all commencing students must complete the module by census date, **31 March 2025**. Continuing students can self-enrol at any time.
- Your coordinator will advise you on whether digital tools including grammar checkers and generative AI are allowed in assessments. Failure to comply with the coordinator's instructions on digital tools could amount to an academic integrity breach.
- For University's guidelines on generative AI, see next slide.

Strategies for maintaining academic integrity

- Planning and time management
- Know your strengths and what you need to develop
- Use citations and referencing
- Know when and where to ask for help

sydney.edu.au/students/academic-integrity

Using generative AI in COMP9120

- Generative AI can create digital content, including text, images, and video.
- Common examples are ChatGPT, Microsoft Copilot, Notion AI, Claude, etc.
- Different guidelines for use of generative AI apply to learning and assessment.

Generative AI for learning

- You are free to use generative AI to help you learn.
- Use it to explain things, apply knowledge, plan your study, make practice questions, and more.
- However, be aware of **its limitations** and **flaws**.

Generative AI for assessment

- *The use of generative AI tools are **NOT permitted** in COMP9120 for assessment purposes.*

To learn more about using generative AI visit the Canvas resource developed by students: https://bit.ly/students-ai

Managing access to our University systems

MultiFactor Authentication (MFA) is an additional security step to verify your identity at the University

More information

✓ Enter your UniKey and password on the login screen. ✓ Confirm your identity using the Okta Verify app on your phone.	✓ Set up a secondary email, phone number, and security question in your Okta profile to be able to reset your Unikey easily.	✓ Always have your Okta-enabled device with you. ✓ Do not uninstall the Okta Verify app.	Changing your mobile device? ✓ Install Okta Verify on your new device before removing it from your old one.

The Learning Hub introduces *Studiosity*!

- **Writing Feedback:** Get writing and referencing feedback within 24 hours by uploading a draft into Studiosity.
- **Connect Live:** Enter a question in Connect Live and you will be put in touch with a Subject Specialist in an interactive online classroom.
- Available to Postgraduate Coursework students via Canvas

Access your Studiosity account to receive feedback on your draft assignment:

THE UNIVERSITY OF SYDNEY

Do you have a disability, medical condition, or caring responsibilities that impact on your studies?

You may not think of yourself as having a 'disability' but the definition under the **Disability Discrimination Act (1992)** is broad and includes temporary or chronic medical conditions, physical or sensory disabilities, psychological conditions and learning disabilities.

If you have ongoing caring responsibilities for another person (who has a disability, medical condition, mental health condition, or who is aged and frail) as defined by **Carer Registration Act 2010**
which impact on your studies, you can register with IDS.

To get assistance, students need to register with Inclusion and Disability Services. It is advisable to do this as early as possible. Please contact us or review our website to find out more.

THE UNIVERSITY OF SYDNEY

Inclusion and Disability Services Office
sydney.edu.au/disability
02-8627-8422

Extra Support in COMP9120

Drop-in Helpdesk:

- Week **5, 6** and **9, 10, 11**
- Date/Time: TBA
- Venue: TBA

NEW:

Ed Special Helpdesk

- Dedicated Day/Time: TBA

Time for a break!

Let’s play a game!

THE UNIVERSITY OF SYDNEY

Part B: Introduction

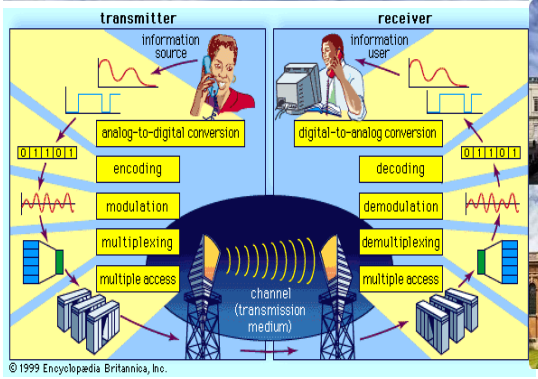
Why do we study **Databases**?

- They are ***everywhere!***
  - They are the *power engine* behind everything that is *digital*!

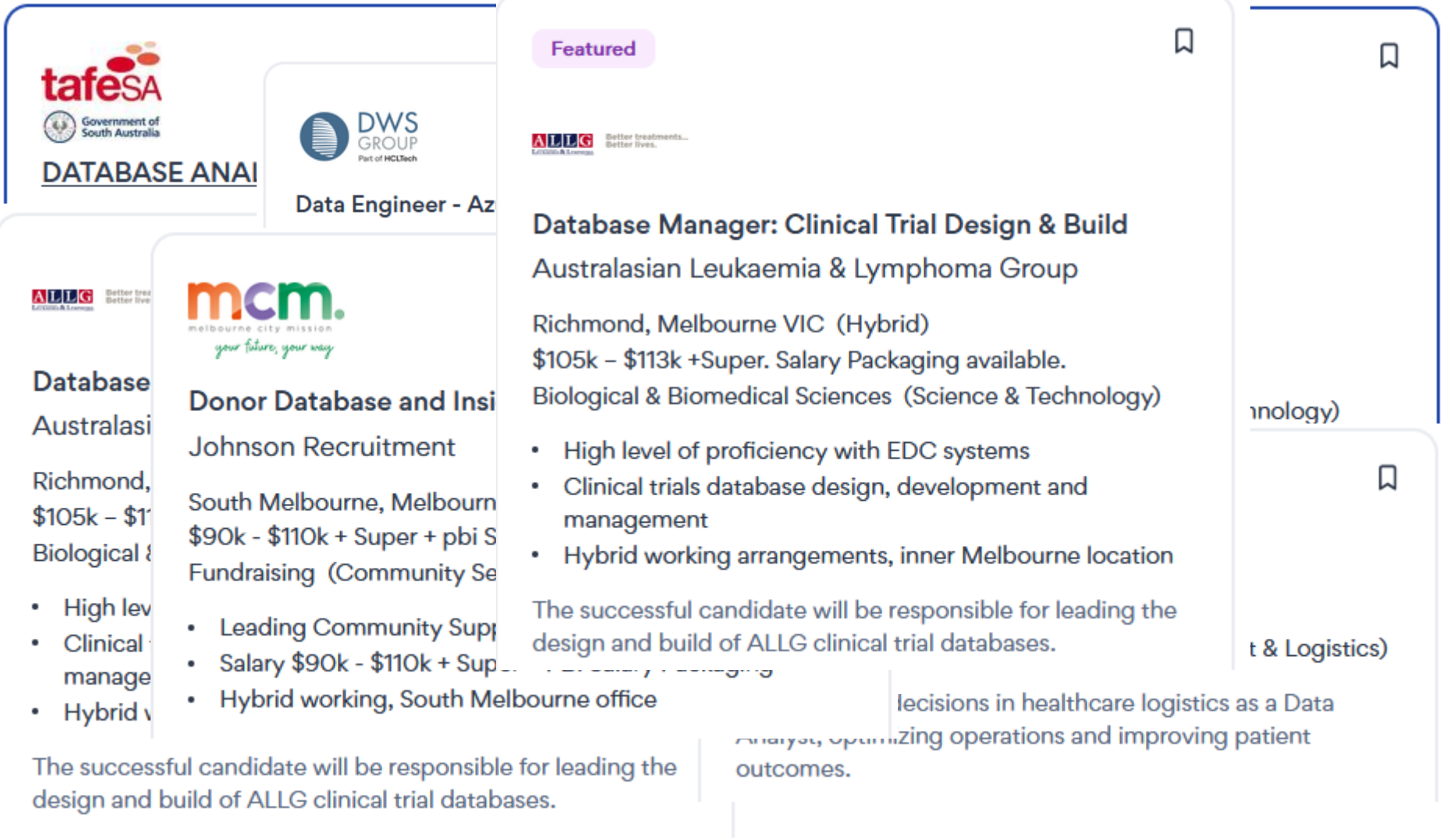
Examples

- **Banking systems**: accounts & loans, customers, all types of transactions (banks, ATMs, internet)
- **Social networks**: Facebook, Youtube, Whatsapp, Viber, etc
- **Large Language Models** (LLMs): ChatGPT, Gemini, Co-Pilot, etc
- **Airlines reservation systems**: reservations by customers, flight schedules, frequent flyer info
- **Corporate records**:
  - Universities: student enrolments, course offerings, timetabling, grades
  - Sales: customers, products, purchases - and reports on these
- **Telecommunication**: calls, bills, calling/SIM cards
- **Healthcare**:
  - patients, prescriptions, drugs

Databases touch *all aspects* of our lives



Career Opportunities



What is a Database?

- A large integrated collection of data, central to every enterprise/organisation managed through a software system, called *Database Management System* (DBMS)

- Models a real-world *Universe of Discourse (UoD) – e.g. University operations* – It consists of:
  - **Entities** (e.g., Students, Courses)
  - **Relationships** (e.g., Alice is enrolled in COMP9120)

Key Questions on How Databases should be Managed

- Do we want to write *custom* programs to access & manipulate database?

- How do we answer questions such as:
  - Count of students in a course?
  - Average workload of staff?
- How do we protect against concurrency anomalies (i.e., issues related to concurrent operations)?
- What if program crashes in the middle of operations?
- How do we make sure only the *right people* have the *right permissions* to *access* and *manipulate* the data?

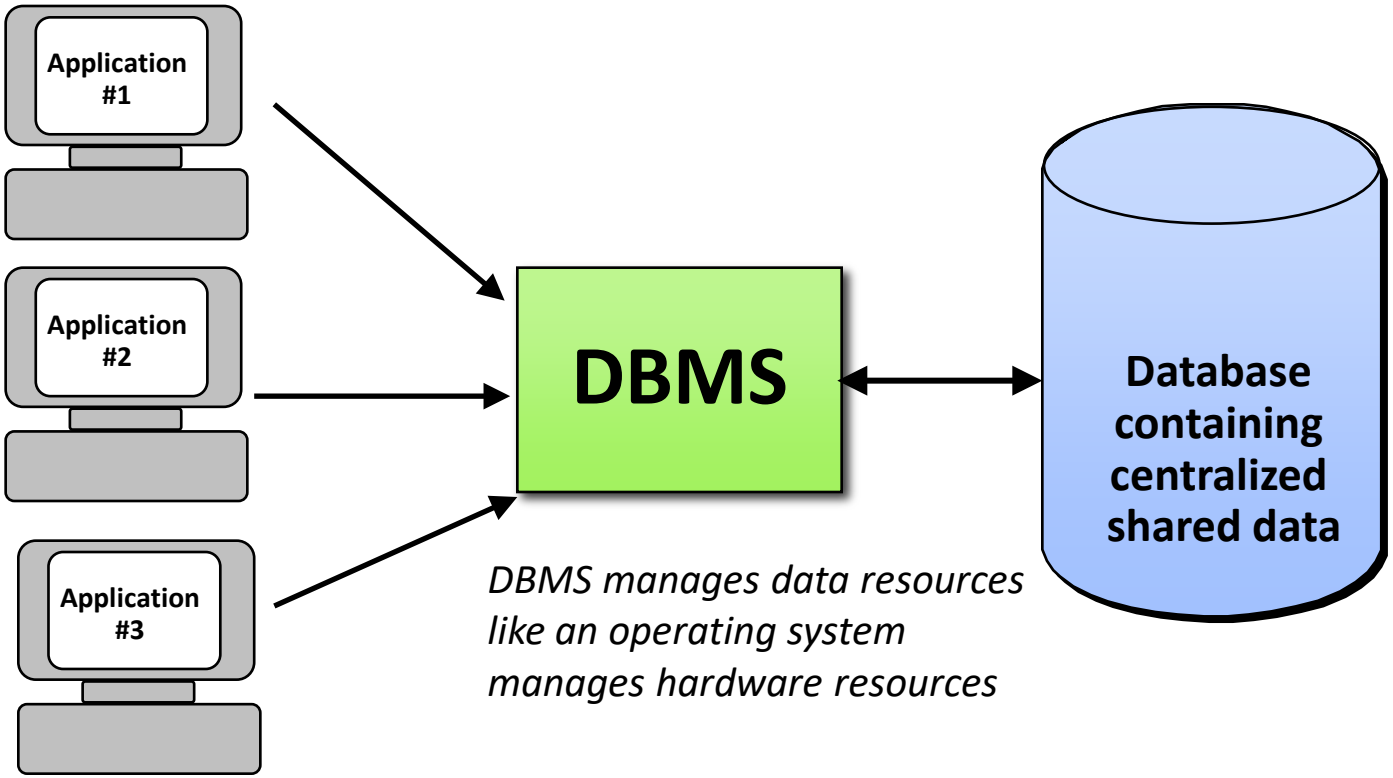
THE UNIVERSITY OF SYDNEY

Manage a Database by a DBMS!

- **Database Management System (DBMS)**
  - Stores the database on some mass (persistent) storage
    - Provide efficient storage along with fail safety (backup/recovery)
  - Supports a high-level query language (e.g. SQL)
    - DBMS interprets statements of the query language to perform requested database access.
  - Provides transaction management
    - Guarantee correct concurrent access to shared data

The DBMS Approach

- Central repository of shared data
- Stored in a convenient and efficient form
- Data is centrally managed by a DBMS

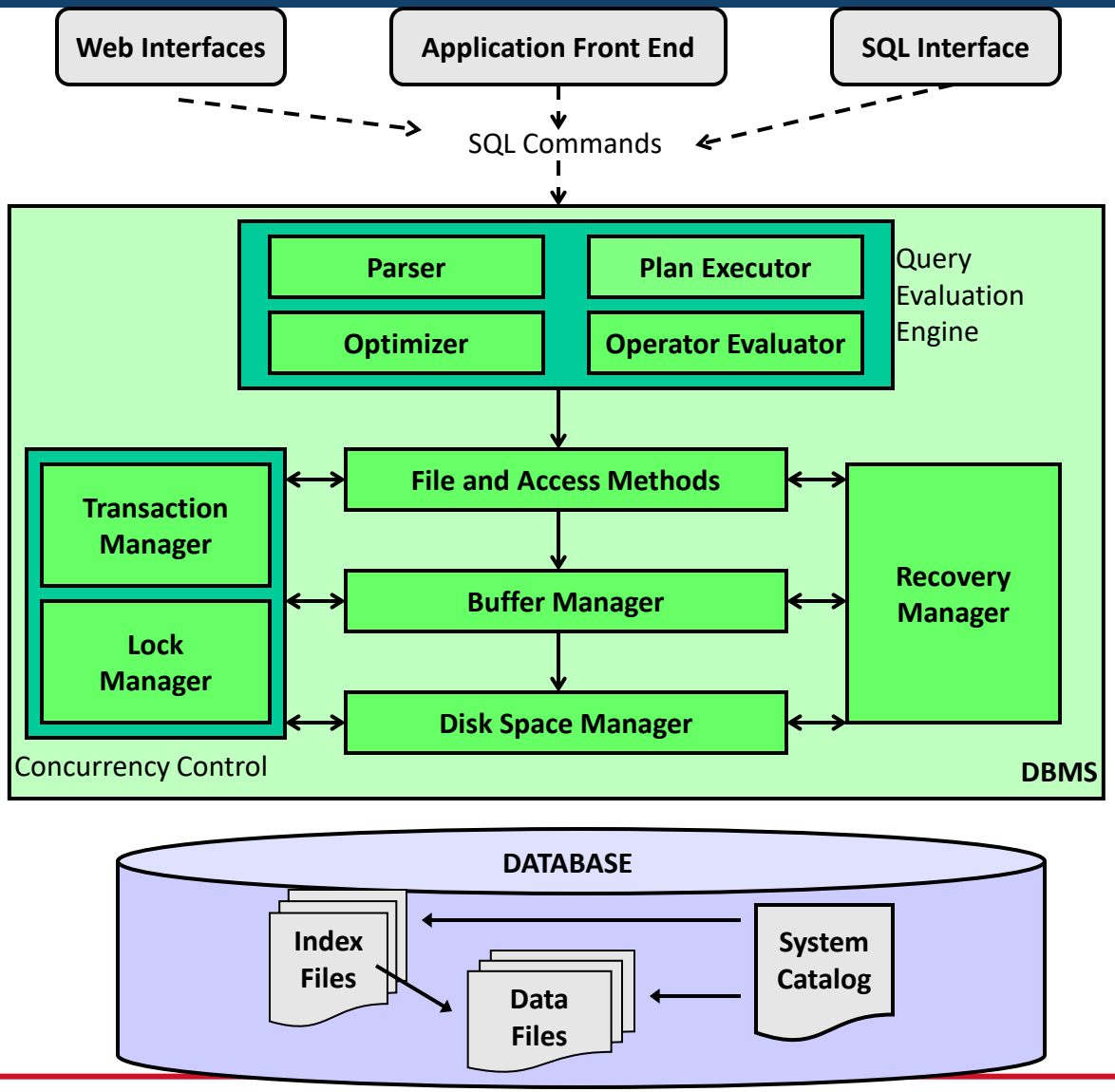


Advantages of DBMS

- Improved Data Sharing & Security
  - Different users get different views of the data
- Enforcement of Standards
  - All data access is done in the same way
- Improved Data Quality
  - Integrity constraints, data validation rules
- Better Data Accessibility/ Responsiveness
  - Use of standard data query language (SQL)
- Backup/Recovery, Concurrency
  - Disaster recovery is easier

Structure of a DBMS

- A typical DBMS has a **layered** architecture
- This is **one** of several possible architectures
  Each system has its own variations



Relational Data Model

- This unit focuses on **relational databases**, which use the *relational data model:* The most widespread among DB vendors
  - For other data models, please check COMP5338: Advanced Data Models

- Definition: A **data model** is a *collection* of *concepts* for *describing data*
  - The relational data model is the *most widely* used model!
    - Main concepts: *relation* (essentially, a *table*), *functional dependencies (*e.g., *relationships)* and *schema (*collection of *related tables):* based on *set theory* and *functions.*

Student				
sid	name	email	gender	address
5312666	Jones	ajon@cs.com	M	123 Main St
5366668	Smith	paul@mail.com	M	45 George
5309650	Jin	jin@it.com	F	19 City Rd

Relational Database

- A relational database is a *collection* of *interrelated* tables , referred to as a *schema*
  - Tables are typically related to each other by *key* attributes

- Example: *Course management database* represented by the following schema*:*
  - Students(sid: *string*, name: *string*, gpa: *float*)
  - Courses(cid: *string*, cname: *string*, credits: *int*)
  - Enrolled(sid: *string,* cid*: string,* grade*: string*)

sid	**Name**	**GPA**
S123	Bob	3.2
S456	Mary	3.8

Students

cid	**cname**	**credits**
C101	COMP9120	6
C102	COMP5338	6

Courses

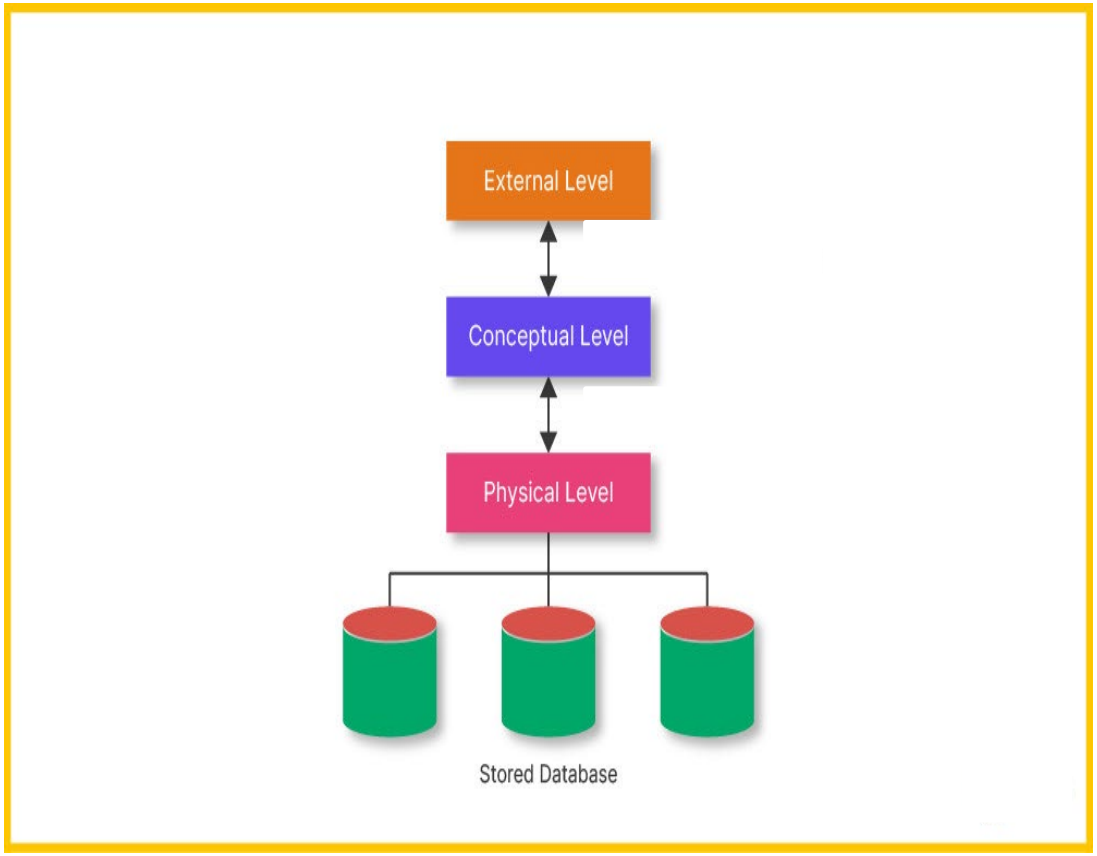
sid	**cid**	**Grade**
S123	C101	A

Enrolled

Schema Levels in Databases

- Three (3) Levels:
  - External Schema/Application
  - Conceptual Schema
  - Physical Schema

Schema Levels



https://unstop.com/blog/data-independence-in-dbms

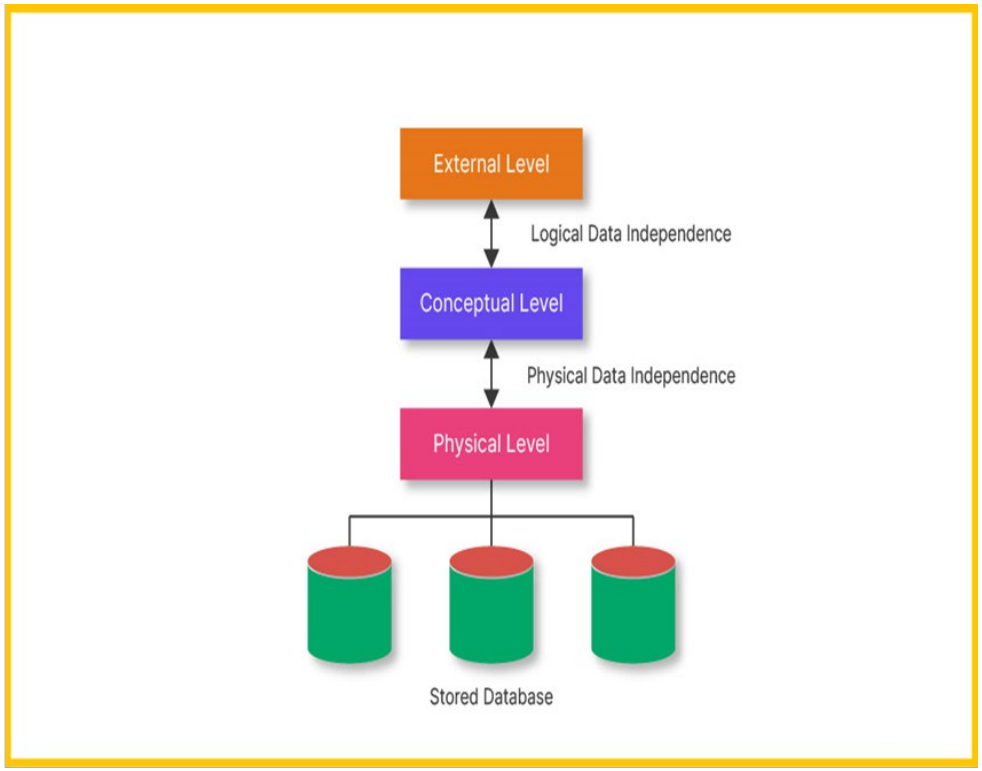
DBMS Services

Main services:

- Data Independence
- Declarative Querying
- Transaction Management & Concurrency Control

Data Independence

Data Independence: *Applications* do *not* need to worry about *how data is structured* and *stored*



https://unstop.com/blog/data-independence-in-dbms

Logical data independence:
protection of the applications from changes in the *logical structure of the data*

i.e., should not need to worry whether we can add a new entity or attribute without rewriting the application/view

Physical data independence:
protection of the conceptual schema (and applications) from *physical layout changes*

i.e., should not need to worry which disks are the data stored on, or whether the data is indexed.

Declarative Querying: "What" *not* "How"

- It is convenient to indicate declaratively ***what*** information is needed, and leave it to the system to work out ***how*** to process through the data to extract what we need

- Users should be offered a way to express their requests *declaratively*
  - A query language which is based on *first-order logic* (e.g., *tuple calculus*)
    - SELECT…FROM…WHERE…

Queries in a DBMS

- DBMS provides a query language combining 3 *specialized languages* for accessing data
  - **DML** - Data Manipulation Language - *Select, insert, update*
  - **DDL** - Data Definition Language - *Create, Alter, Drop*
  - **DCL** - Data Control Language - *Grant, Revoke*

- Standard Query Language for Relational DBMS: **SQL**
  - Based on formal query languages: Relational Algebra and Tuple Calculus

SQL Example

› The *working-horse* command: **SELECT – FROM – WHERE**

- *retrieves* data from one or more tables of a relational database that fulfill a *search condition*

Example 1:

SELECT name, email
FROM Student
WHERE sid=5312666

name	email
Jones	ajon@cs.com

Example 2:

SELECT *
FROM Student

Example 3:

SELECT COUNT(*)
FROM Student
WHERE gender='F'

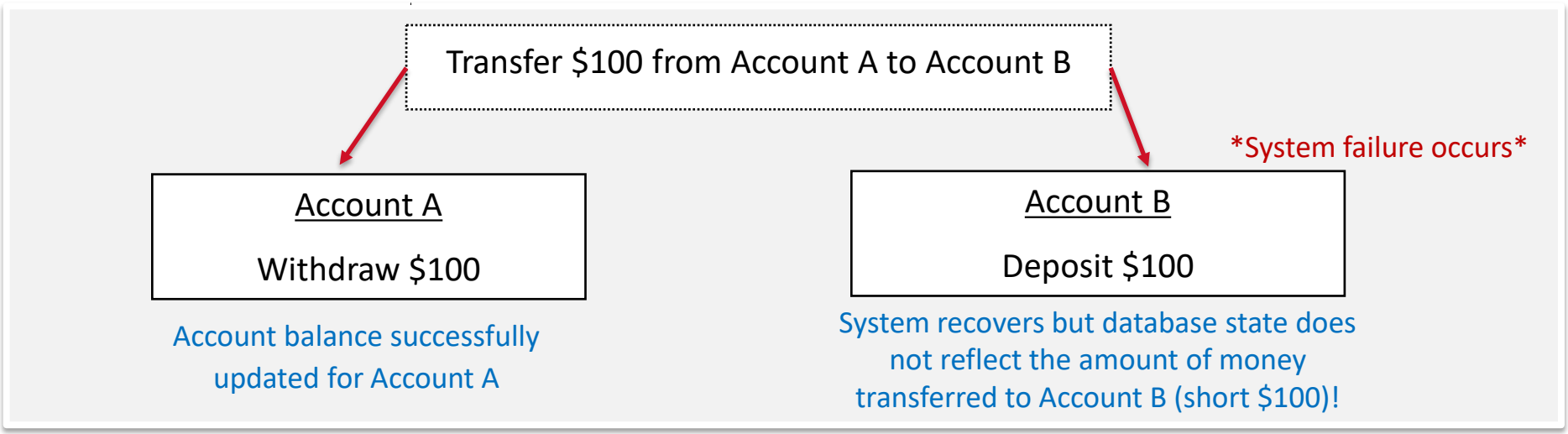
COUNT(*)
1

Student

sid	name	email	gender	address
5312666	Jones	ajon@cs.com	M	123 Main St
5366668	Smith	paul@db.com	M	45 George
5309650	Jin	jin@it.com	F	19 City Rd

What is a Transaction?

- "The basic unit of change as seen by a DBMS"
  (Ramakrishnan & Gherke, Database Management Systems, Ch1)
- This unit contains the execution of a piece of code that either executes *completely* or *not at all* (ie: **atomic execution**).
  - *No such thing as partially complete transactions*! This helps manage failure scenarios.



- Transaction atomicity is achieved via *locking* and *logging*.
- Concurrent control algorithms (e.g., *two-phase locking*) also help prevent interference between two different transaction executions that access & update the same data.

Desirable DBMS Requirements

- **High Availability**: must be operational all the time or close to that.
- **High Reliability**: correctly tracks state, does not lose data, controlled concurrency
- **High Throughput**: accommodate many users => many transactions/sec
- **Low Response Time**: users don't wait for too long
- **Long Lifetime**: complex systems are not easily replaced – need to be *adaptive* to changes
  - Must be designed so DBMS can be easily extended as the needs of the enterprise change
- **Security**:
  - Sensitive information must be carefully protected since system is accessible to many users
    - Authentication, authorization, encryption

THE UNIVERSITY OF SYDNEY

Roles in DBMS

- **System Analysts**
  - Specifies system using input from customer; provides complete description of functionality from customer's and user's point of view
  - Conceptual database design
- **Database Designer**
  - Specifies structure of data that will be stored in database (logical & physical database schemas)
- **DB Application Programmer**
  - Implements application programs (including transactions) that access data and support enterprise rules
- **Database Administrator (DBA)**
  - Maintains database once system is operational: space allocation, performance optimization, database security, deals with failures and congestion
- **End-Users**
  - Often unaware that they are dealing with data in a DBMS

THE UNIVERSITY OF SYDNEY

Summary

- DBMS is used to maintain & query large datasets that are shared by many application programs/users

- Some important concepts:
  - Logical and Physical Data Independence
  - Declarative Queries
  - Transactions

- Every ‘knowledge worker’ or scientists needs to have the database know-how, as do all IT experts- not just DBAs

References

- Ramakrishnan/Gehrke (3rd edition)
  - ***Chapter 1***
- Kifer/Bernstein/Lewis (2nd edition)
  - Chapters 1.1-1.3, 2.1, 2.2, 3.1, 3.2
  - Missing: comparison with file-based info system
- Ullman/Widom (3rd edition)
  - Chapters 1.1, 2.1, 2.2
  - Missing: comparison with file-based info system, roles of workers
- Silberschatz/Korth/Sudarshan (5th edition)
  - Chapters 1.1-1.5, 1.12, 2.1
- Tony Hey et. al (Ed.): *The Fourth Paradigm: Data-Intensive Scientific Discovery*, Microsoft Research, 2009
  - http://research.microsoft.com/en-us/collaboration/fourthparadigm/

Next Week

- Conceptual Database Design using the
  - Entity Relationship Model

- Readings:
  - ***Ramakrishnan/Gehrke, Chapter 2***
  - Kifer/Bernstein/Lewis book, Chapter 4
  - Ullman/Widom, Chapter 4

See you next week!