Course Review

CSCI3170 2024T1

Course – Learning Outcomes

The students will be able to

1. use an E-R diagram to model a database;
2. translate an E-R diagram into a relational model;
3. fine tune a relational schema based on the principles of relational database normalization;
4. implement queries by using database languages (SQL in particular);
5. understand file organizations and index structures of a DBMS;
6. ~~understand the ideas of query processing and query optimization;~~
7. understand the principles of concurrency control and recovery schemes;

Overview: Database Design

- Data models: ER, Relational Data Model and their mapping
- Relational Algebra: Being able to use relational algebra to answer question.
- Relational Database Design: Functional Dependency, Normal Forms, Design Algorithms for 3rd normal form and B-C normal form (3.5 normal form)

Overview: RDBMS + Other DB

- Disk, Files, Buffer Replacement Policy
- Index: Introduction.
- Transaction Management
  - ACID properties
  - Conflict-serializable vs serializable
  - Concurrency control (locking, time-stamp ordering)

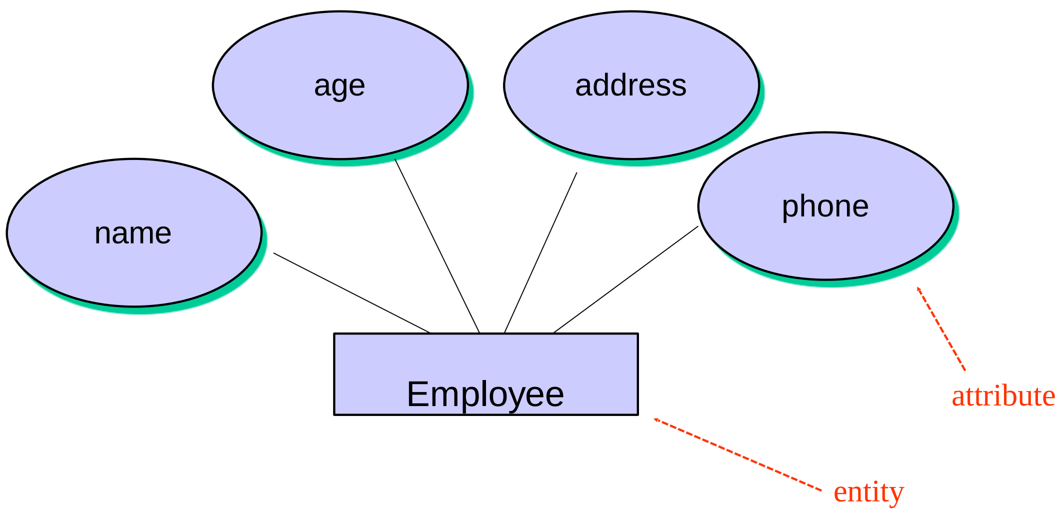
What’s Important

- Why Database: Files vs. DBMS
- Database Modeling: Entity-relationship Model
- How it Works: Relational Algebra and SQL-based Application
- Make it Faster: Schema Refinement
- The Underlying: Storage vs. Memory, Indexes, and B+ Tree
- More: Hashing, Concurrency Control, and Recovery

- (Note: This short slide aims to help you organize what you’ve learned, please don’t forget to review the more comprehensive lecture slide.)

Entity-Relationship diagram (E-R diagram)

- The E-R model can be presented graphically by an E-R diagram.



Relational data model

- Most DBMS today are based on the relational data model.
- Relation
  - the central data description construct in this model.
  - It can be thought of as a set of records.
  - A table with rows and columns.
    - Row – a record
    - Column – field, attribute.

Relational data model

sid	name	login	age	gpa
53666	Jones	Jones@cs	18	3.4
53688	Smith	Smith@ee	18	3.2
53650	Smith	Smith@math	19	3.8
53831	Madayan	Madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

An example of a student relation

- A description of data in terms of a data model is called a schema.
- The schema of the above table is

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

Basic Operators

- There are six basic operators in relation algebra:
  - select (σ)
  - project (Π)
  - union (U)
  - set different (-)
  - Cartesian product (x)
  - rename (ρ)

Motivation Example

- Redundant Storage
  - The rating value 8 corresponds to the hourly wage 10, and this association is repeated three times.
  - Storage is not used efficiently.

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

- A decomposition of a relation schema R consists of replacing the relation schema by two (or more) relation schemas that each contains a subset of attributes of R and together include all attributes in R

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

Id	name	age	rating	Hours_worked
123-22-3666	Peter	48	8	40
231-31-5368	Paul	22	8	30
131-24-3650	Mary	35	5	30
434-26-3751	David	35	5	32
612-67-4134	Ada	35	8	40

rating	Hourly_wages
8	10
5	7

Functional dependency:
- rating determines Hourly_wages

Disks and Files

- DBMS stores information on (“hard”) disks.
  - A disk is a sequence of bytes, each has a disk address.
  - READ: transfer data from disk to main memory (RAM).
  - WRITE: transfer data from RAM to disk.
  - Both are high-cost, relative to in-memory operations.
- Data are stored and retrieved in units called disk blocks or pages.
  - Each page has a fixed size, say 512 bytes. It contains a sequence of records.
  - In many (not always) cases, records in a page have the same size, say 100 bytes.
  - In many (not always) cases, records implement relational tuples.

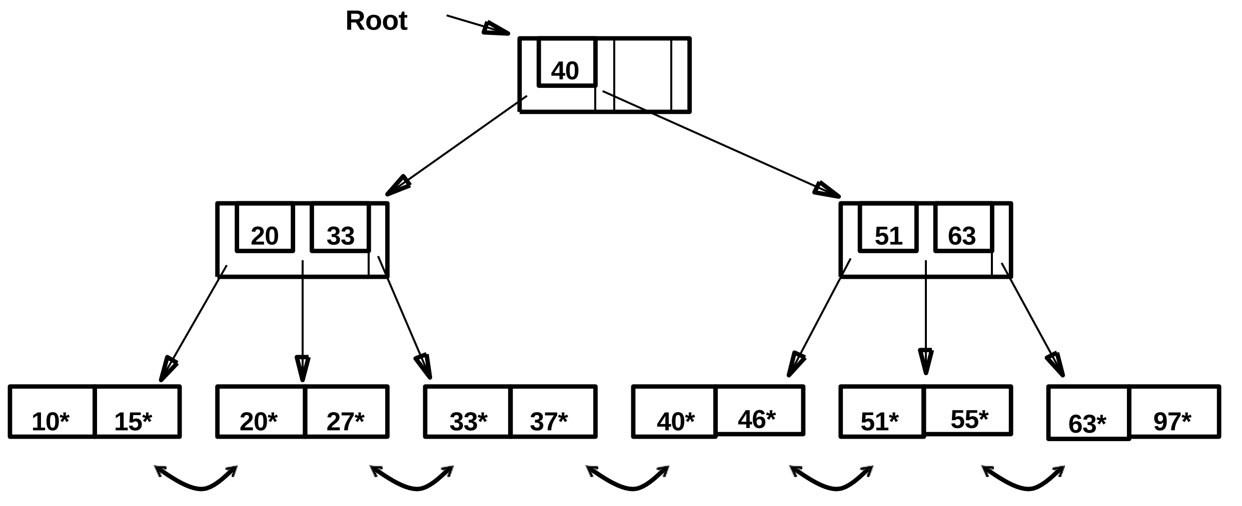
Why Not Store Everything in Main Memory?

- Costs too much.
  - Ram is much more expensive than disk.
- Main memory is volatile.
  - We want data to be saved between runs. (Obviously!)
- Typical storage hierarchy:
  - Main memory (RAM) for currently used data. Disk for the main database (secondary storage).
  - Tapes for archiving older versions of the data (tertiary storage).

Indexes

- An index on a file speeds up selections on the search key fields
- for the index.
  - Any subset of the fields of a relation can be the search key for an index on the relation.
  - Search key is not the same as key (minimal set of fields that uniquely identify a record in a relation).
- An index contains a collection of data entries
  - A data entry is denoted as k*, where k is a search key value and * tells where to find the record containing k
  - Index must support efficient retrieval of all data entries k* with a given key value k. Structure of data entry in more detail
- Primary vs. secondary: If search key contains primary key, then called primary index, otherwise secondary.
- Unique index: Search key contains a candidate key.

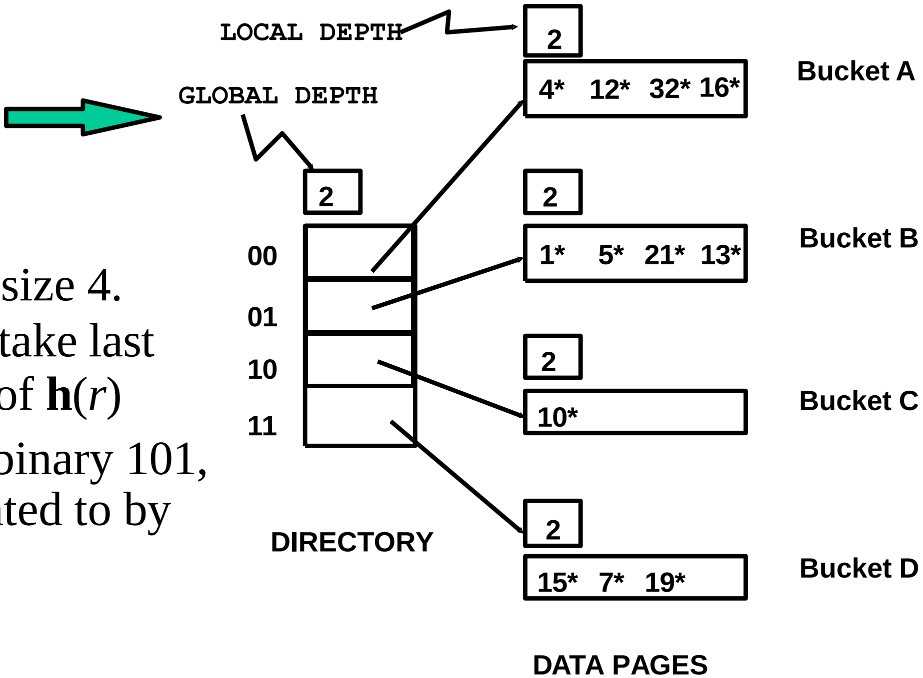
An example of a B+ tree



Extendible Hashing

- Situation: Bucket (primary page) becomes full. Why not re-organize file by doubling # of buckets?
  - Must re-hash all data entries to the right buckets
  - Example: assume hash function h(k) = k mod M
    - For M = 4, entries 3* and 7* both in bucket 3 (3 mod 4 = 7 mod 4 = 3)
    - But for M = 8, entry 7* will be in bucket 7
  - Can we only re-hash those values that have changed addresses?
    - Difficulties: without re-hashing all the values, we don't know which values keep the old addresses and which get new addresses
  - Reading and writing all pages are expensive!
  - Question: how do we add more buckets, but only re-hash a few data entries?
  - Answer: use a level of indirection, directory of pointers to buckets

Example



- **h**(*r*) = *r mod 32*
- Directory is array of size 4.
- To find bucket for *r*, take last \`*global depth*' # bits of **h**(*r*)
  - If *r* = 5, **h**(*r*) = 5 = binary 101, 5* is in bucket pointed to by 01.

❖ **Insert**: If bucket is full, *split* it (*allocate new page, re-distribute*).

❖ *If necessary*, double the directory. (As we will see, splitting a bucket does not always require doubling; we can tell by comparing *global depth* with *local depth* for the split bucket.)

Concurrency

- Multiple users.
- Concurrent accesses.
- Problems could arise if there is no control.
- Example:
  - Transaction 1 (T1): withdraw $500 from account A.
  - Transaction 2 (T2): deposit $800 to account A.

T_1	T_2
Read(A) A = A – 500 Write(A)	Read(A) A = A + 800 Write(A)

Transactions

- A *transaction* is a sequence of read/write operations.
- A transaction is *atomic*.
  - Either all or none of the operations are executed.
  - No transaction can observe partial effect of other transactions.

Recoverable

- Transactions may be aborted due to logical failure. e.g. deadlock
- Recoverability is required to ensure that aborting a transaction does not change the semantics of committed transaction's operations.

- Example
  - $Write_1(x,2)$; $Read_2(x)$; $Write_2(y,3)$; $Commit_2$
  - Not recoverable.
  - T_2 has committed before T_1 commits.
  - The problem is: what can we do if T_1 abort?
  - Delaying the commitment of T_2 can avoid this problem
- A schedule *H* is called *recoverable* (RC) if, whenever T_i reads from T_j. ($i \neq j$) in *H* and $c_i \in H$, $c_j < c_i$.
- Intuitively, a history is recoverable if each transaction commits after the commitment of all transactions (other than itself) from which it reads.

In Exam, I will ask about

- I will ask mostly about
  - ER Model, Relational Model
  - Relational Algebra
  - Storage, Indexes *(e.g. Buffers, B+ Trees)*
  - Transactions, Schedules, Concurrency
  - Recovery (conceptual mostly)
- None
  - View serializability
  - Differences on clustered or unclustered Indexes
  - Java
  - SQL Triggers/ Check Constraints

Final Exam/Assignment

- 2.5 hrs., 9:30am – 12:00pm 6 Dec
- **Consultation: 25 Nov and 27 Nov 3-4pm**
- You are permitted ONE Double Sided Cheat Sheet (print/non-print)
- For the additional 30 minutes: I will add some conceptual questions

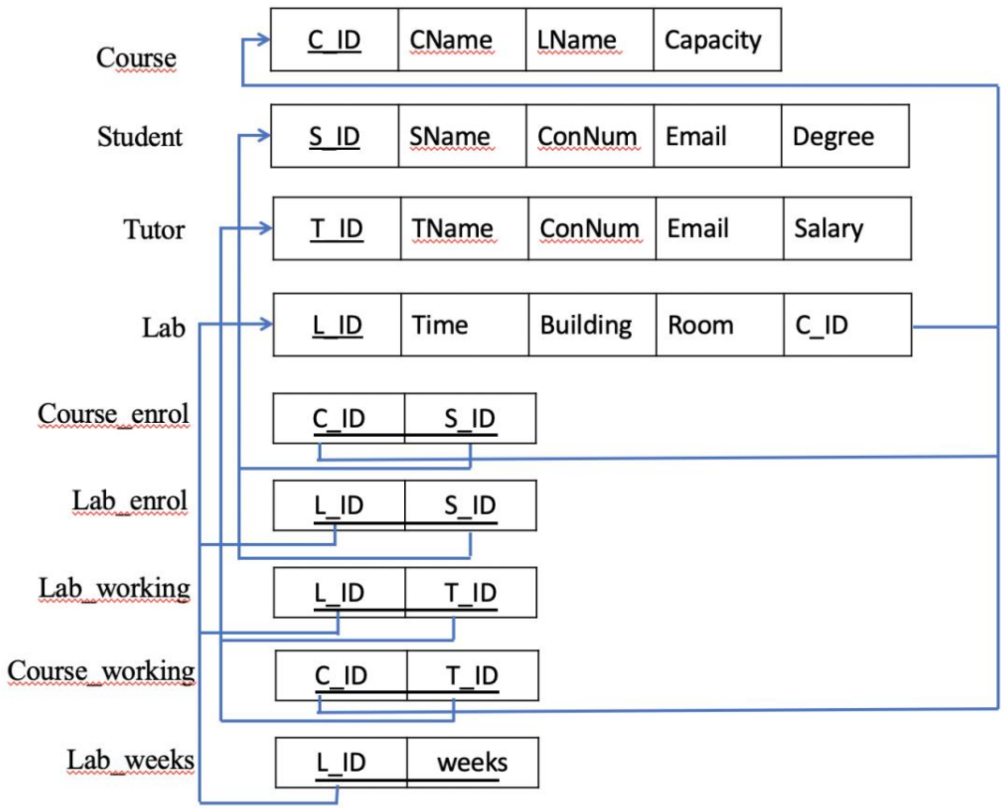
  E.g. "Briefly explain the how the requirements of 3NF enforce the 1NF and the 2NF (hint: you may frame your discussion by the types of functional dependencies 3NF disallows) ***(4 marks)"***
- That need you to be somewhat familiar with the <u>meaning</u> of the definitions

Remaining Matters of Business

- This week:
  - Me submit final exam to RES
  - Release of Assignment One Marks
  - Release of In-class assessment Solutions
  - Course Teaching Evaluation by Tomorrow
  - Some students with current grouping difficulties..
- This weekend:
  - Some students who have lost Oracle Logins
  - Some standard notations to expect/follow in Exam

If I ask you to draw a "relational schema"

- Draw **relational diagram** in exam as shown below



Project - Oracle Database

- *Select table_name from user_tables;*
- *describe <tablename>*

Connecting to Oracle Database

```
1. import java.sql.*;
2. Class.forName ("oracle.jdbc.driver.OracleDriver");
3. DriverManager.registerDriver (new
   oracle.jdbc.driver.OracleDriver());
4. Connection conn =
   DriverManager.getConnection("jdbc:oracle:thin:@//db18.cse.
   cuhk.edu.hk:1521/oradb.cse.cuhk.edu.hk", "<username>",
   "<password>");
5. Statement stmt = conn.createStatement();
6. String query =
7. stmt.execute(query);
```

Some Plans for Next Years

- Scrapping Group-based Project
  - 5 Assignments +
  - At least two coding: implementation of B+ Tree/ Buffer Pool in any pre-approved language of your choosing (C++, Python, Java)
- Participation (
  - Per week N: rank a given list of concepts in terms of understanding + a 100-200 statement on the why you don't understand your least understood concept
- A bigger Database (that CSE servers could hold..)
- Tutorials -> Lab Format?
  - Move all of SQL to lab content, and cover advanced SQL in lectures
  - Existing material -> Practice

- Welcome More Comments… put them in the CTE

Course and Teaching Evaluation (CTE) for Term 1, 2024-25

Notes for Students:

- The CTE links were sent to your **CUHK Link email accounts**. Please check.
- You will receive separate emails for teacher's evaluation and TA's evaluation.
- Subject of the email is "Online CTE Questionnaire for [course code]: [course name] – [Teacher or TA name]".
- No login is required and all feedback is anonymous.
- Course teacher will give you ~10-15 minutes to fill out the CTE questionnaires before the end of the class.
- **Submission deadline: By 23:59 tomorrow**