Database Systems (ISYS1001/ISYS5008)

Lecture 1

Overview, Relational Model , Creating tables with SQL

Updated: 28th July,2021

Discipline of Computing
School of Electrical Engineering, Computing and Mathematical Sciences (EECMS)

CRICOS Provide Code: 00301J

Copyright Warning

COMMONWEALTH OF AUSTRALIA
Copyright Regulation 1969

WARNING

This material has been copied and communicated to you by or on behalf of **Curtin University of Technology** 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

Objectives

- Identify the overview of Database Systems
- Describe the basic idea of Relational model
- Create SQL queries for creating Tables
- Additionally knowing,
  - unit outline
  - Assessments

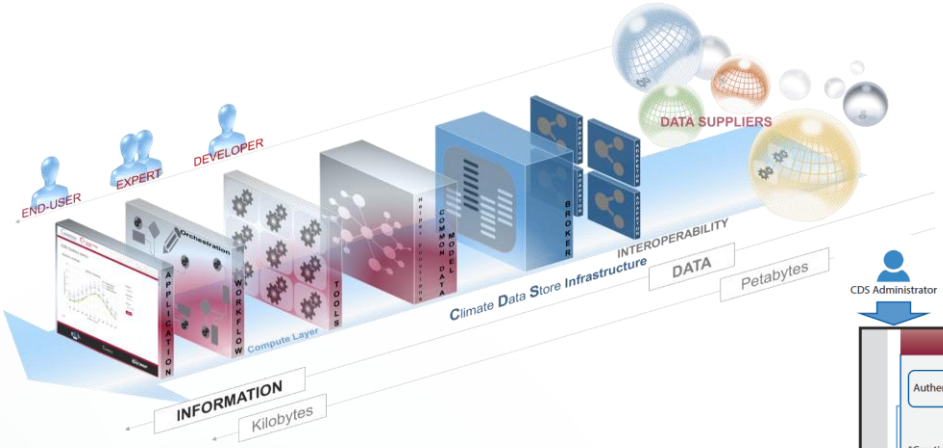
Databases and Database Management Systems

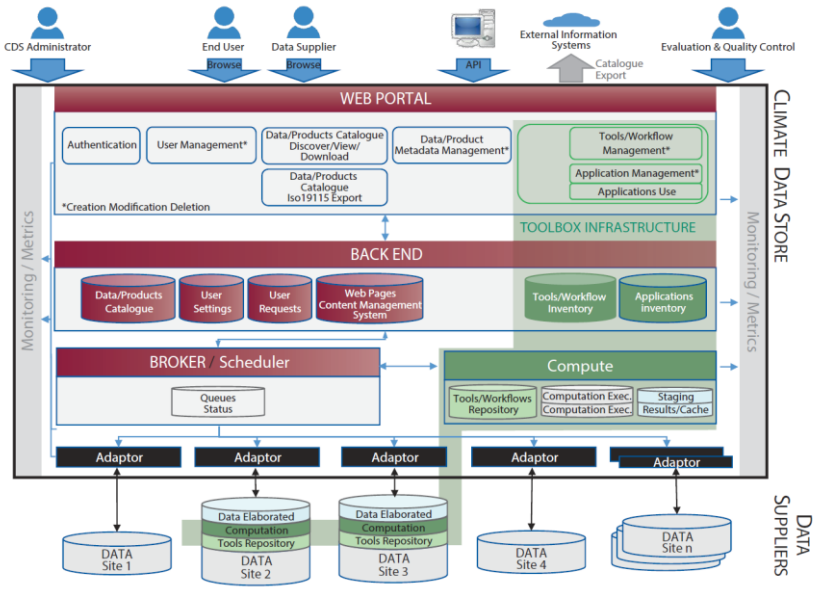
- How to store large volume of data effectively in a computer system?
- Files Vs Databases
- Database
  - Storage facility to keep large (can be extremely large) amount of data in a manageable manner
  - It used to be about boring stuff: employee records, bank records, etc.
  - Today, the field covers all the largest sources of data, with many new ideas
    - Web search
    - Data mining
    - Scientific and medical databases.
    - Integrating information
  - You may not notice it, but databases are behind almost everything you do on the Web.
    - Google searches
    - Queries at Amazon, eBay, etc.

Databases and Database Management Systems

- Database Management System
  - DBMS is a Software System manage Databases
  - DBMS is a Software system enabling creating, storing, updating and maintaining databases effective manner
  - Manage large data with efficiency, persistence, security
  - Available for Phones, PC's, workstations, mainframes, supercomputers
- Knowledge and techniques used in DBMS span many areas of computer science:
  - Languages, object orientation and other programming paradigms, compilation, operating systems, concurrent programming, data structures, algorithms, theory, parallel and distributed systems, dynamic programming, user interfaces, expert systems and AI, statistical techniques

Example : Satellite data for Climate analysis

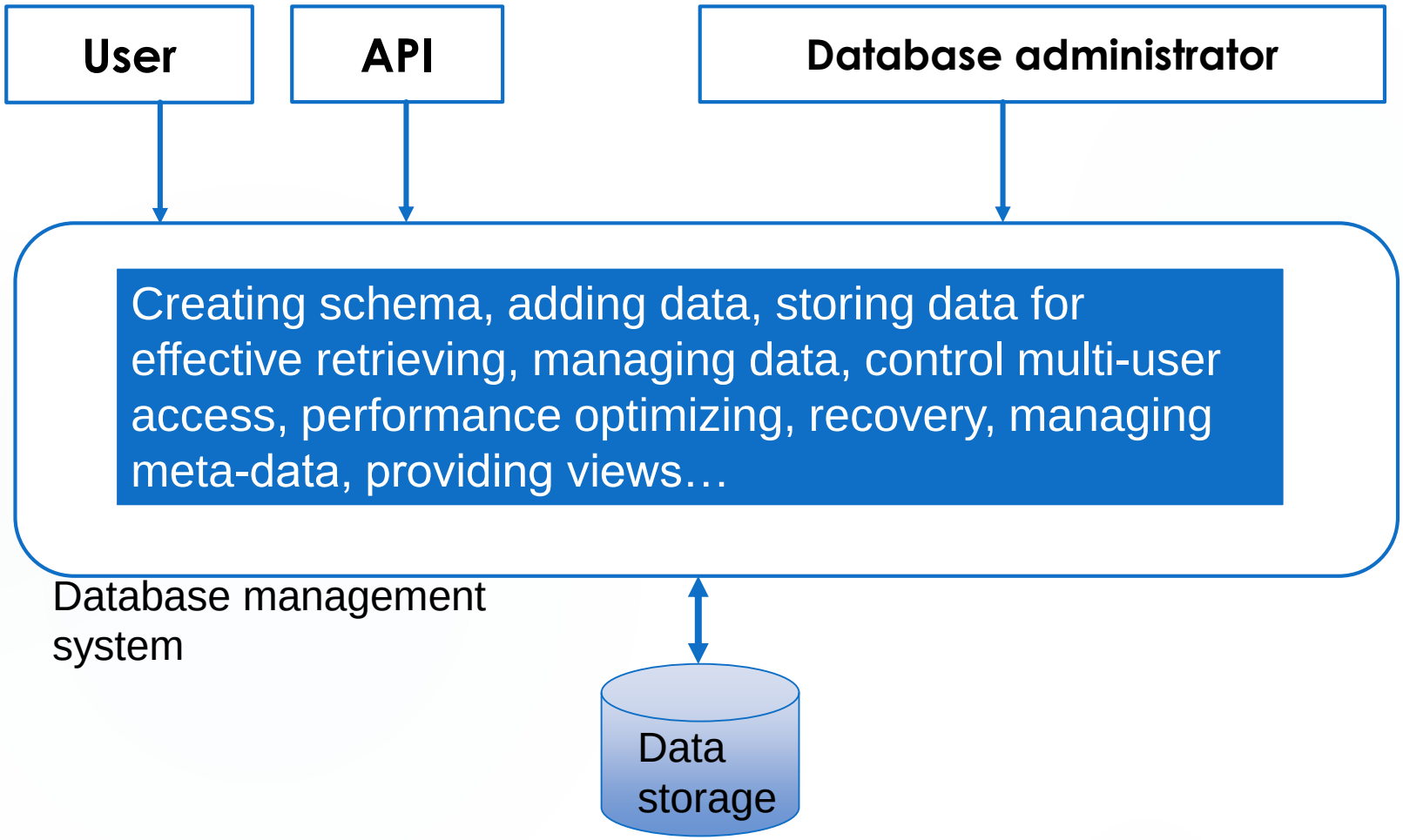




Providing services with Sentinel-2 satellite data

Image source: https://climate.copernicus.eu/climate-data-store#062e8d73-1d2f-497c-a202-54dfff86c960

Simplified overview of a database system



Functionalities provided by a DBMS

- Create new databases
  - Logical structure (schema)
  - **Data – Definition Language**
- Questioning about data (query)
  - **Data-manipulation language (query language)**
- Support storing large volume of data effectively
  - Massive volume of data; therefore need efficient ways to store
- Support Durability
  - Recovery from system failures
- Control access by many users at once (multi-user concurrent access)

Key People

Many roles exist in computing and other domains related to database systems

- Database designer
  - Establishes schema
- DBMS implementer
  - Builds system
- Database application developer
  - Programs that operate on database
- Database administrator
  - Loads data, provide access control, keeps DB running smoothly
- Software Engineers may need to work with databases as well

Major DBMS Products and Freeware

- Commercial products:
  - Oracle
  - IBM: DB2
  - Microsoft: SQL Server, Access
- Freeware:
  - **MySQL**
  - Postgres
- All are "**relational**" (or "object-relational") database systems at their core.
- There are non-relational databases, which are becoming increasingly popular, especially for data science.

Unit Learning Outcomes

- Be able to
  - **create, query, update and manage** Relational Databases in a DBMS environment.
  - **design** relational database schema using the ER model and normalization
  - **integrate** a Relational Database with Java or Python
  - **describe** the principles of correctness for concurrent transactions and explain the techniques for managing concurrent transactions in a DBMS.

Text and References

- Recommended text
  - Ullman, J.D. and Widom, J., A First Course in Database Systems, 3rd Ed., Pearson Prentice Hall, 2008.
- Other References
  - Mana Takahashi and Shoko Azuma. 2009. The Manga Guide to Databases. No Starch Press, San Francisco, CA, USA
  - Garcia-Molina, H., Ullman, J.D. and Widom, J., Database Systems: The Complete Book, 2nd Ed., Pearson Prentice Hall, 2008.
  - Ramakrishnan, R. and Gehrke, J., Database Management Systems, 3nd Editon, McGraw-Hill, 2003.

Unit Resources

- Unit Page
  - On Blackboard (http://lms.curtin.edu.au)
  - Lecture slides and other unit materials will be available from the unit page under “Unit materials”
  - Lectures from the previous semester is available also
  - There may be minor modifications of the lecturers from previous semester and new version would be available weekly basis
  - Lecture recordings will be available as ilectures after the lecture
- MySQL Documentation
  - http://dev.mysql.com/doc/

Pre-Requisite Knowledge

- Ability to program in Java of Python
  - Only when we look at using a Java/Python and connect to a database (close to the end)
- Working knowledge of Linux operating system

- Those are co-requisites, so learning this unit while learning FOP/PDI etc. is sufficient

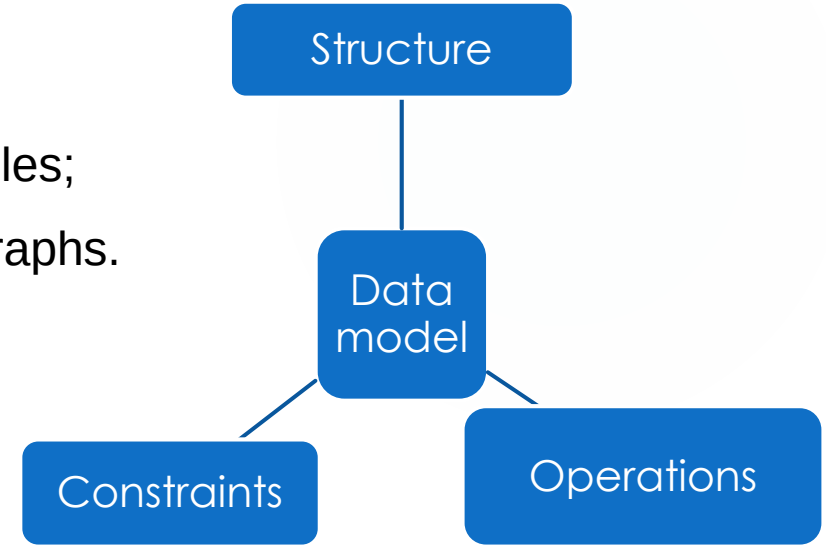
Unit outline

- Look at the complete unit outline in Blackboard
- Practical tests will be held during lab time
- Mid term test will be held during lecture time

Data models, Relational model and creating tables

What is a Data Model?

- A notation for describing data, consists of three parts:
  - Structure of the data:
    - Conceptual model
    - Examples: relational model = tables;
    - Semi-structured model = trees/graphs.
  - Operations on the data:
    - Limited but high level operations
  - Constraints:
    - Limitations on what data can do



Data models for DBMS

- There are a number of data models, and their relative importance has changed in the last decade.
  - hierarchical models
  - Network models
  - …
- Traditionally the most important has been the **Relational model**, including object-relational extensions
- Other models (including Network and unstructured) are becoming increasingly useful.
- Our focus is on the Relational model.

Relational data model

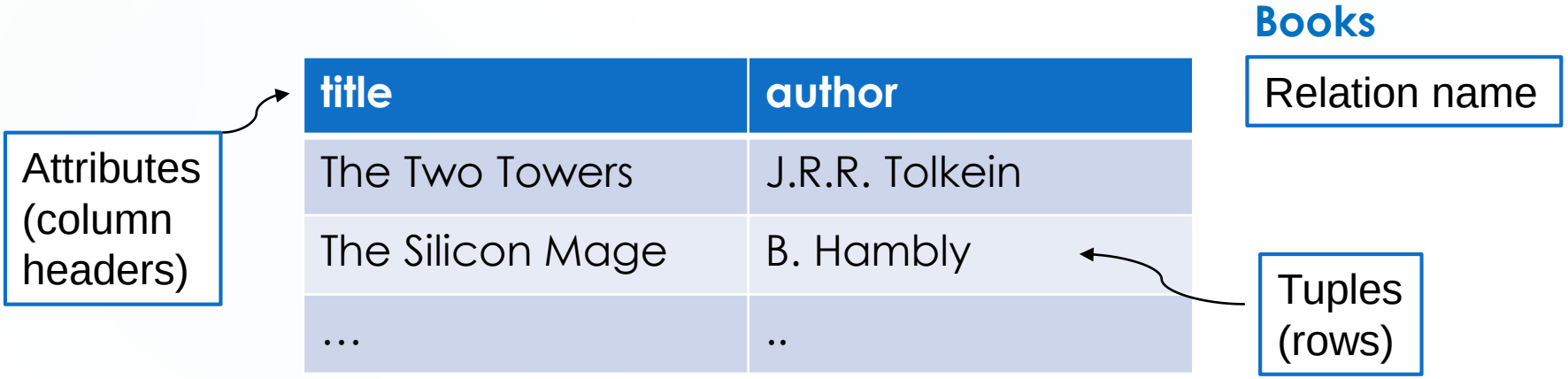
- "database system should present the user with a view of data organized as tables called relations."

  -Ted Codd -1970

- Data is representing as tables
  - Physical implementation

    might be different

A Relation is a Table



This isn't actually the Books relation, but it shows the concept.

Relational data model

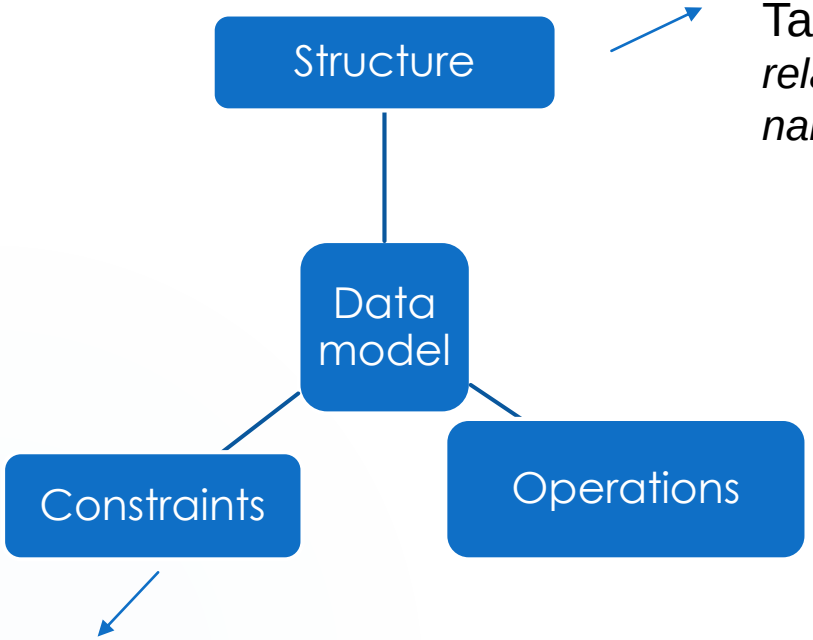


Table
relation name, attributes names

Key, referential integrity
….
(more later)

- Limited but high level operations
- Queries
- Based on relational algebra

Schemas

- *Relation schema* = relation name and attribute list.
  - Optionally include types of attributes
  - Example:

    Books(title, author) or

    Books(title:varchar(30), author:varchar(60))

- *Database* = collection of relations

- *Database schema* = set of all relation schemas in the database

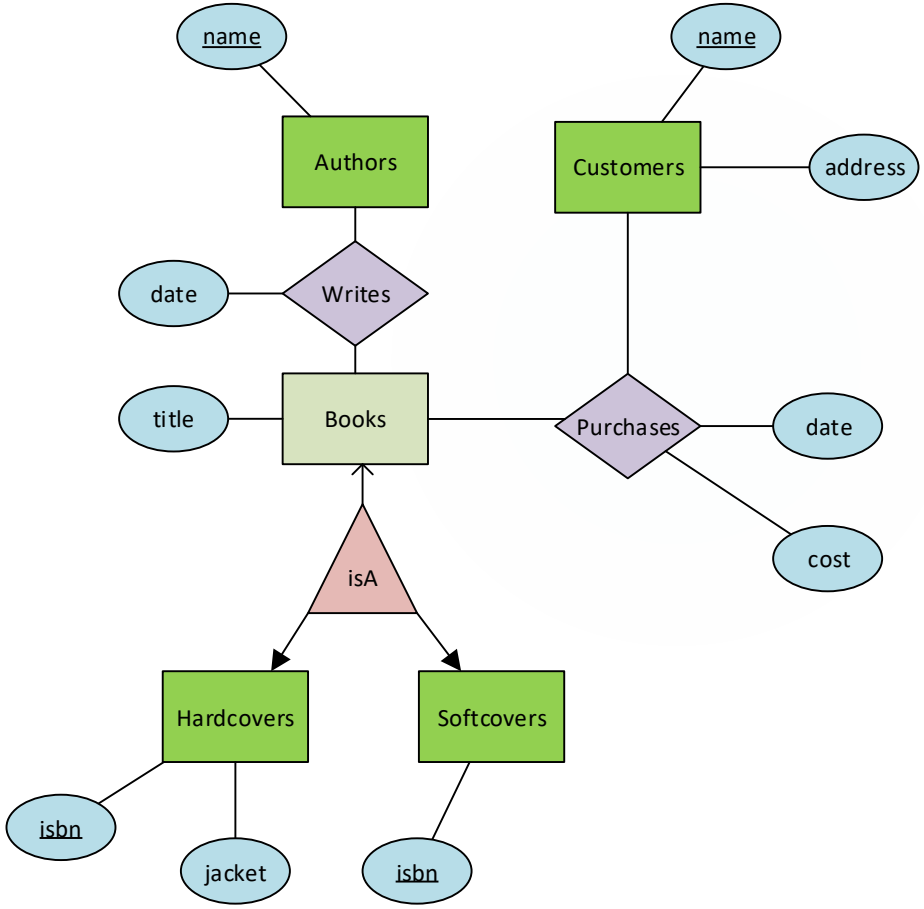
Why Relational model?

- Very simple and structured approach
- Provides a limited, but useful collection of operations
- *Often* matches how we think about data
  - most of the things can be modelled
- Abstract model that underlies SQL, the most important database language today.
- **Used by most major commercial database systems**

Relational model and ER model

- In order to represent data as tables, the required information has to be identified and model first
  - **Entity-relationship** diagrams (**ER model**)
  - UML diagrams
  - ….
- Diagrams are really useful to conceptualize ideas
- ER diagrams are very useful when designing a database
- ER model is converted to database schema by some standard methods
- More details :

Lecture 3-4, practical 4-6



ER Diagrams differentiate between Entities (such as Authors and Books) and the Relations between them.

Entities vs Relations

- An Entity is generally a collection of **things**.
  - Cars, Airplanes, Boats, Books
  - People, Students, Staff, Managers, Authors, Customers
  - Places, Buildings, Rooms, Venues, Destinations, Terminals
  - Units, Courses, Invoices

- A Relation **connects** two (or more) Entities.
  - A Student Enrolls in one or more Units.
  - A Customer Purchases one or more Books.
  - A Building Contains one or more Rooms.
  - A Tutorial is an instance of a Unit at a particular date and time, with many Students and a Staff member scheduled.

Database Example

- Adopt as our running example:

  Authors(name)

  Writes(author_name, book_id, date)

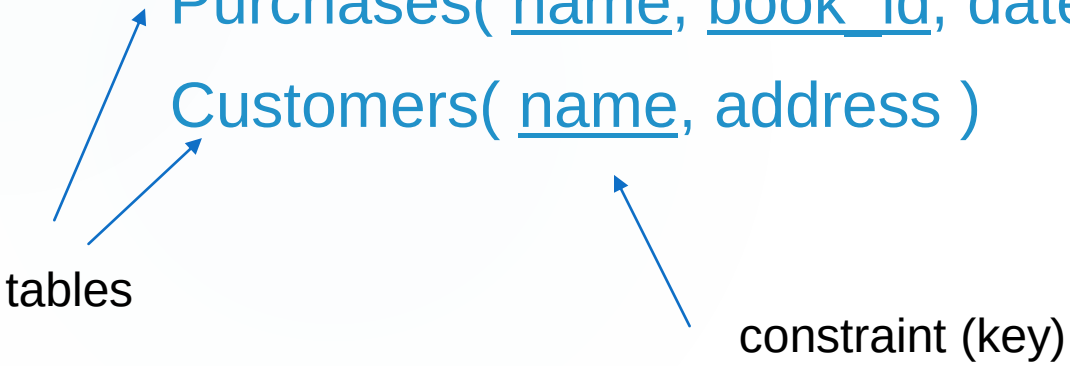
  Books(isbn,title)

  Hardcovers(isbn, jacket, price)

  Softcovers(isbn, price)

  Purchases(name, book_id, date, cost)

  Customers(name, address)



SQL

- SQL ("S.Q.L." or "sequel") - Structured Query Language
- Query language of most commercial relational DBMS's
- SQL is a big language, not just queries and updates.
- Components of the language:
  - Schema definition, Data retrieval, Data modification
  - Indexes, Constraints
  - Views, Triggers
  - Transactions, Authorization
  - *etc*.
- Structured language, therefore limited set of operations-> short and fast

SQL

- Basically two aspects :
  - Data –definition sub language (DDL part)

    for declaring database schema
  - Data manipulation sub-language (DML part)

    for querying (asking questions about data)
- Some operations from both aspects will be covered in this unit
- This unit discusses generic SQL but uses MySQL for details
- We will start with creating tables (assuming a database is already created and available)

Creating (Declaring) a Relation

- Simplest form is:

```
CREATE TABLE <name> (
        <list of elements>
    );
```

Note: replace the content within <> with actual values

Most basic element: an attribute and its type.

- To delete a relation:

```
DROP TABLE <name>;
```

Common Data Types

1. `INT` or `INTEGER` (synonyms).

   Also `SMALLINT`, `BIGINT`, *etc*.

2. `REAL` or `FLOAT` (synonyms)

   There are approximate.

3. `DECIMAL(`*n*`,`*m*`)` = total *n* digits of which *m* are to the right of the decimal point.

4. `CHAR(`*n*`)` = fixed length character string, padded with blanks.

5. `VARCHAR(`*n*`)` = variable-length strings up to *n* characters.

6. `DATE, TIME and DATETIME`

Example: Create Table


```
CREATE TABLE Purchases(
    name    VARCHAR(20),
    book_id    DECIMAL(13),
    date    DATETIME,
    cost    DECIMAL(5,2)
);
```

name	book	date	cost

Identifiers in SQL

- Identifiers (relation name, column names, etc.) in SQL **start with a letter** followed by alpha-numeric characters,
  - max length 30 in SQL and 64 for almost all in MySQL
  - MySQL allows identifiers to start with a non-character, but identifiers may not be entirely composed of numbers
  - MySQL also allows $ and _ (underscore) and a number of other symbols depending on whether the string is quoted (surround by quotes) or not.
- Reserved words not allowed as identifiers
  - *e.g.*, table, create, char, date, select, etc.
- Two objects in a name-space cannot have identical names
  - Example: two tables in a database; two columns in a table, *etc*.

Inserting rows

- To insert a single tuple:

```
INSERT INTO <relation>
VALUES(<list of values>);
```

- Example:
  - Add a tuple to **Purchases(name, book_id, date, cost):**

```
INSERT INTO  Purchases
VALUES  (‘Charlie Brown’, 9780312094119,
‘2021-04-13 14:22:01’, 32.99);
```

  - Values must be listed in column sequence

More on adding tuples

- Specify columns when values not in column sequence or data is missing:

```
INSERT
INTO Purchases (name, book_id, cost)
VALUES('Peter Pan',9780142001196, 12.50);
```

name	Book_id	date	cost
Peter Pan	9780142001196	NULL	12.50

Retrieving


```
SELECT < attribute1,
attribute2 ,..>
FROM <relation>
WHERE <condition>;
```

Purchases Table

name	Book_id	date	cost
Peter Pan	….	…	12.50
Charlie Brown	…	..	32.99

e.g.,

```
SELECT name, book_id
FROM Purchases
WHERE cost>25;
```

name	book
Charlie Brown	9780142001196

```
SELECT * FROM
Purchases;
```

name	Book_id	date	cost
Peter Pan	9780142001196	NULL	12.50
Charlie Brown	9780142001196	2021-04-13 14:22:01	32.99

Deleting rows

- To delete all tuples satisfying a condition from some relation:

```
DELETE FROM <relation>
WHERE <condition>;
```

- *e.g.*, name='Peter Pan':

```
DELETE
FROM Purchases
WHERE name = 'Peter Pan'
```

WHERE clause determines which rows deleted.

After deletion:

name	book_id	date	cost
Charlie Brown	9780142001196	2021-04-13 14:22:01	32.99

- Delete all tuples

```
DELETE FROM <relation>;
```

- Note: no WHERE clause needed.
- e.g., Make the relation Purchases empty:

```
DELETE FROM Purchases;
```

Modifying tuples

- To change certain attributes in certain tuples of a relation:

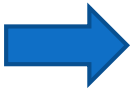
```
UPDATE <relation>
SET <list of attribute assignments>
WHERE <condition on tuples>;
```

Modifying tuples

- e.g.,

```
UPDATE Hardcovers
SET COST = 18.50
WHERE isbn = 9781435114944;
```

isbn	jacket	cost
9780330320559	fairy	16.28
9780786965625	None	49.95
9781435114944	None	**20.00**



isbn	jacket	cost
9780330320559	fairy	16.28
9780786965625	None	49.95
9781435114944	None	**18.50**

- What is no WHERE clause used?

SQL Values

- Integers and reals are represented as you would expect.
- Strings are too, except they require single quotes.
  - Two single quotes = real quote
  - *e.g.*, `'William Shakespeare''s The Phantom Menace'.`
- Any value can be NULL.

Summary

- DBMS used to store, maintain, and query large datasets
- DBMS draws knowledge from many areas of CS
- Data model: consist of structures, operations and constraints
- Relational model : represents data as tables
- In order to identify and conceptually model tables required for a database, ER diagrams can be used
- SQL: Language for defining tables, querying and other data manipulation (insert, delete, etc., based on relational model

Happy Database systems

next week : more on SQL