

L1: Relational Databases

CS1106/CS6503: Intro to Relational Databases

Dr Kieran T. Herley

Semester One, 2023-24

School of Computer Science & Information Technology
University College Cork

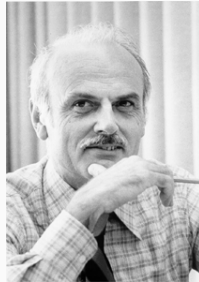
Summary

Relational databases: concepts and terminology. The SQL query language. Syntax of simple project-filter queries.

Relational Database Concept

Relational databases

- Predominant database concept since 1980s
- Conceptually organizes data into sets of two-dimensional tables
- Many relational DB software packages
 - Open Source: MySQL, Postgres
 - Proprietary: Oracle, Microsoft, IBM, Sybase and others



- Edgar “Ted” Codd (1923–2003), English computer scientist
- Introduced relational database concept c1970
- Turing Award 1981

Organization Relational databases are based on *tables*
modules

| <i>mod_code</i> | <i>lecturer</i> | <i>mod_title</i> |
|-----------------|-----------------|--|
| cs1101 | Morrison | Systems Organisation |
| cs1105 | Brown | Foundations of Computer Science |
| cs1106 | Herley | Introduction to Relational Databases |
| cs1107 | Bowen | Intro. to Multimedia Authoring Systems |
| cs1108 | Doherty | Computing: Profession and Society |
| cs1109 | Bridge | Programming and Web Development |

Benefits Conceptually intuitive, simple and elegant

Anatomy of a relational database

Database = set of named *relations* (aka tables)

Relation

- A set of *tuples* (aka rows) with identical column structure
- Each column named with an *attribute* (must be distinct)
- Values in each column of uniform “type” drawn from some set (*domain*) such as integer, text, date and so on.

Anatomy of a relational database cont'd

students

| <i>id_number</i> | <i>first_name</i> | <i>last_name</i> | <i>date_of_birth</i> | <i>hometown</i> | <i>course</i> | <i>points</i> |
|------------------|-------------------|------------------|----------------------|-----------------|---------------|---------------|
| 112345678 | Aoife | Ahern | 1993-01-25 | Cork | ck401 | 500 |
| 112467389 | Barry | Barry | 1980-06-30 | Tralee | ck402 | 450 |
| 112356489 | Ciara | Callaghan | 1993-03-14 | Limerick | ck401 | 425 |
| 112986347 | Declan | Duffy | 1993-11-03 | Cork | ck407 | 550 |
| 112561728 | Eimear | Early | 1993-07-18 | Thurles | ck406 | 475 |
| 112836467 | Fionn | Fitzgerald | 1994-06-13 | Bandon | ck405 | 485 |

Some technicalities

- According to relational model, all rows should be distinct; DB systems don't always enforce this however
- Values must be *atomic* (“elementary” and “unstructured”)
 - Examples: integers, decimals, text, booleans, dates
 - Can't have lists, sets, records, arrays as single values
 - (Can represent lists etc. by other means—later)
- Every relation has a *key*: some subset of attributes such that no two tuples in the relation can ever have the same key value.

Schema vs instance

students

| <i>id_number</i> | <i>first_name</i> | <i>last_name</i> | <i>date_of_birth</i> | <i>hometown</i> | <i>course</i> | <i>points</i> |
|------------------|-------------------|------------------|----------------------|-----------------|---------------|---------------|
| 112345678 | Aoife | Ahern | 1993-01-25 | Cork | ck401 | 500 |
| 112467389 | Barry | Barry | 1980-06-30 | Tralee | ck402 | 450 |
| 112356489 | Ciara | Callaghan | 1993-03-14 | Limerick | ck401 | 425 |
| 112986347 | Declan | Duffy | 1993-11-03 | Cork | ck407 | 550 |
| 112561728 | Eimear | Early | 1993-07-18 | Thurles | ck406 | 475 |
| 112836467 | Fionn | Fitzgerald | 1994-06-13 | Bandon | ck405 | 485 |

Database Schema

- “structure” of DB
- Static remains fixed over time

Database Instance

- “snapshot” i.e. specific collection of values populating DB
- dynamic i.e. changes over time

Brief digression on adjective “relational”

Binary relations in mathematics

- Let $\mathbb{Z}^2 = \mathbb{Z} \times \mathbb{Z}$ be the set of pairs of integers; Any subset of \mathbb{Z}^2 is a *binary relation*
- For example

$$\mathcal{L} = \{(x, y) : x, y \in \mathbb{Z} \text{ such that } x \leq y\}$$

is the set of integer pairs where the first is no larger than the second

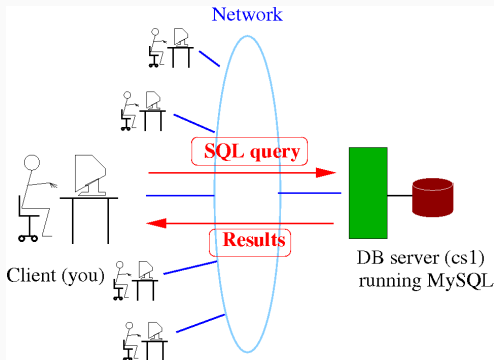
Other relations Can define relations with any number of components; Component values need not be restricted to integers

Relations and DBs

- Can interpret elements of k -ary (i.e. k components) relation as rows of a table
- Mathematical underpinning useful in handling DB manipulations

SQL

SQL and typical DB setup



Our “instructions” to DBMS must be expressed in notation called SQL

What is SQL?

- Computer language for expressing management, manipulation and querying of relational databases
- Pronounced either as “ess-qu-ell” or “sequel”

Support

Good News “Standard” DB language; most common DBMSs support it; free open sources packages available

Bad News Most DBMSs adopt their own “dialects”

- DB software is generally not completely portable between DBMSs

Why computer languages?

Natural languages English can be imprecise and ambiguous, so very difficult to write software to “understand” and carry out our instructions/intentions reliably

Computer languages

- Designed to be clear, concise and unambiguous
- Amenable to automatic interpretation and execution
- Can seem cryptic and error-intolerant
- Examples: SQL, HTML, PHP, Java, . . .

What SQL allows us to express

- SQL incorporates several aspects

Data Manipulation Language (DML)

- notation for expressing
- management of DB
- manipulations: adding/removing/updating tuples
- queries to interrogate DB

Data Definition Language (DDL)

- notation for expressing DB structure (databases, tables, domains *etc.*) and so on
- We will focus on SQL queries for now

Some SQL Examples

```
CREATE TABLE students
(
  id_number CHAR(9),
  first_name VARCHAR(20),
  last_name VARCHAR(20),
  date_of_birth DATE,
  hometown VARCHAR(20),
  course CHAR(5),
  points INT,

  PRIMARY KEY (id_number)
);
```

Defines structure of students table

```
SELECT first_name, last_name
FROM students
WHERE points >= 500;
```

Seeks names of 500-point students


```
SELECT first_name, last_name  
FROM students  
WHERE points >= 500;
```

Meaning “Extract names (first and last) of all students with at least 500 points”

Notes

- **SELECT**, **FROM**, **WHERE** are *keywords*
- Other words are names of table/columns
- \geq means \geq
- Terminate query with semicolon

SQL Sample 2 cont'd

Table

| students | | | | | | |
|------------------|-------------------|------------------|----------------------|-----------------|---------------|---------------|
| <i>id_number</i> | <i>first_name</i> | <i>last_name</i> | <i>date_of_birth</i> | <i>hometown</i> | <i>course</i> | <i>points</i> |
| 112345678 | Aoife | Ahern | 1993-01-25 | Cork | ck401 | 500 ● |
| 112467389 | Barry | Barry | 1980-06-30 | Tralee | ck402 | 450 |
| 112356489 | Ciara | Callaghan | 1993-03-14 | Limerick | ck401 | 425 |
| 112986347 | Declan | Duffy | 1993-11-03 | Cork | ck407 | 550 ● |
| 112561728 | Eimear | Early | 1993-07-18 | Thurles | ck406 | 475 |
| 112836467 | Fionn | Fitzgerald | 1994-06-13 | Bandon | ck405 | 485 |

SQL Query

```
SELECT first_name, last_name
FROM students
WHERE points >= 500;
```

Result

| <i>first_name</i> | <i>last_name</i> |
|-------------------|------------------|
| Aoife | Ahern |
| Declan | Duffy |

- Result is a table (anonymous)
- Includes specified columns and rows only
- Original table left completely untouched

A Closer Look At That Query

```
SELECT first_name, last_name      /* First bit */  
FROM students                    /* Second bit */  
WHERE points >= 500;             /* Third bit */
```

- First bit specifies attributes of interest (note comma separators)
- Second bit specifies table the query applies to
- Third bit specifies *criterion* for rows of interest
 - For each row, condition “points >= 500” is true or false based on the points value in that row
 - Only rows where condition is satisfied appear in result

A Closer Look cont'd

```
SELECT first_name, last_name      /* First bit */  
FROM students                    /* Second bit */  
WHERE points >= 500;             /* Third bit */
```

- SQL is very picky about dotting of *is* and crossing of *ts*, so
 - Spell keywords, names correctly (“SELECT” not “SELCT”)
 - Take care of ordering of elements within query (don’t alter order of bits)
 - Use symbols (commas, semicolons etc.) correctly (don’t omit commas etc.)
- (Stuff inside /* . . . */ are *comments*– annotations intended for human not computer consumption– and have no bearing on meaning or execution of query). Can also use --