TDA357/DIT621 – Databases

Lecture 12 – Repetition & History of databases

Jonas Duregård

A history of databases

- In this lecture we will look at the (relatively short) history of databases
- While doing so, we will re-visit topics we have seen throughout the course
- You do <u>not</u> have to memorize any of the names or years, they are merely a way of giving context to how the subject of this course developed

Pre-modern times (by computer scientist standards)

- The mathematical concept of a relation is De Morgan in the 1860s
- It is later expanded by other mathematicians like Frege, Cantor and Russel
- I doubt any of them had applications in databases in mind…

1960s

- The first digital databases appear around the time storage technology develops from tape stations to more HDD-like devices
  - Turns out running queries on tapes that take an hour to rewind is no fun
  - Computers are now undeniably faster than humans at information retrieval
- The first databases are a mess of data and pointers back and forth
- Querying databases requires knowledge of how the data is layed out on disk
- Altering the design of the database typically requires modifying the software that accesses it

The birth of the relational model

- Unlike many discoveries, it is easy to pinpoint when databases started
- This is the first line from Edgar F. Codd's 1970 paper titled "A Relational Model of Data for Large Shared Data Banks":

> *"Future users of large data banks must be protected from having to know how the data is organized in the machine"*

- In a single paper, Codd outlines much of what we teach in this course today
- Codd's contributions to database research landed him a Turing Award in 1981
- As you may have realized, he is the 'C' in BCNF

Early 1970s

- Relational databases are proposed by E. F. Codd
- The term schema is coined, to describe the design of a database
- Schemas are logical descriptions of data organization, disconnected from the physical layout of data in storage devices
  - This became a guiding principle in the field of databases
- First, second and third normal form were all introduced by Codd
  - Later also Boyce-Codd Normal Form
- The process of normalization is intended to address various anomalies that may occur in a relational databases with data redundancies
  - Most notably update and deletion anomalies

Quiz

- Give an example of a deletion anomaly on this schema

Employees(<u>empId</u>, name, officeNumber, officeFloor)

  - Answer: Removing an employee removes the information of which floor its office is on
  - Note that update anomalies are a bit more contrived here: If the floor of an office is changed(?), it needs to be changed for all its occupants

- Give an example of a functional dependency that we would expect to hold in this domain, but which violates BCNF
  - officeNumber -> officeFloor seems like a reasonable FD for this domain

Even more from the early 1970s

- Codd also (in the same paper) elevated relational algebra from a mathematical curiosity into a basis for database query languages
- Introduced most of the relational algebra operators you have seen in this course

Quiz

- Given the same (flawed) schema as before
  Employees(<u>empId</u>, name, officeNumber, floor)
- Write an expression for the floor(s) with the highest number of employees

$R1 = \gamma_{\text{floor, COUNT(*)}\rightarrow\text{number}}(\text{Employees})$

$R2 = \gamma_{\text{MAX(number)}\rightarrow\text{maxNumber}}(R1)$

$\text{Result} = \pi_{\text{floor}}(\sigma_{\text{number=maxNumber}}(R1 \times R2))$

Always a single row

Sanity check: ***(floor, number, maxNumber)***

Quiz

Given this data, compute the result of the expression on the last slide

Table: Employees

empId	office	floor
e1	5113	5
e2	5128	5
e3	6128	6
e4	6142	6
e5	4111	4

$$R1 = \gamma_{\text{floor, COUNT(*)}\rightarrow\text{number}}(\text{Employees})$$

floor	number
4	1
5	2
6	2

$$R2 = \gamma_{\text{MAX(number)->maxNum}}(R1)$$

maxNum
2

$$\text{Result} = \pi_{\text{floor}}(\sigma_{\text{number=maxNumber}}(R1 \times R2))$$

floor
5
6

floor	number	maxNum
5	2	2
6	2	2

floor	number	maxNum
4	1	2
5	2	2
6	2	2

Mid/late 1970s

- The first relational database systems were developed in the later half of the 1970s: *INGRES* (UC Berkeley) and *System R* (IBM)
  - INGRES used a query language called QUEL
  - IBM called their query language SEQUEL
    - Later changed to SQL due to intellectual property issues
    - Developed by R. F. Boyce (The 'B' in BCNF)
  - Later, the INGRES project had a spinoff called Post Ingres
    - The name was contracted to simply "Postgres"
- The term Relational Database Management System (RDBMS) is coined to describe these emerging systems

1976

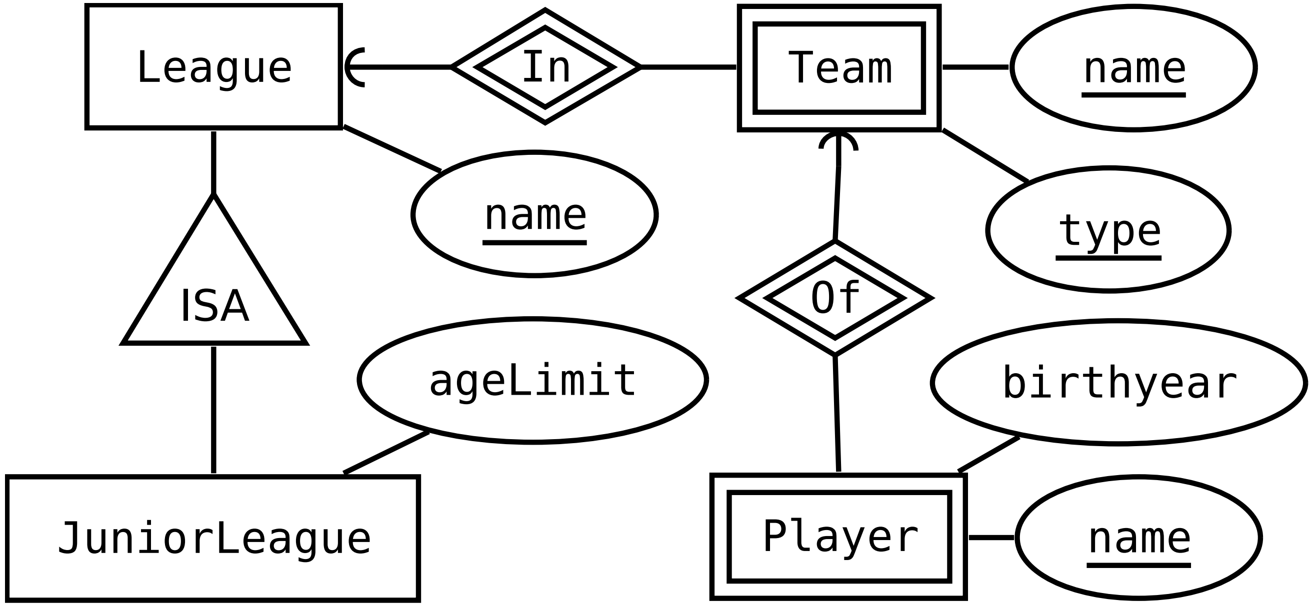
- P. Chen proposed a new model for databases: Entity-Relationships
- Abstracts further away from storage layout, allowing designers to focus on data application instead of table structure

Quiz

- What is the difference between a weak entity and a subentity?
  - Answer: A weak entity can have additional identifying attributes
  - If a weak entity would have no identifying attributes of its own, it would essentially be a subentity
    (we may still decide to use one or the other, to more naturally describe the domain, but technically they are identical)

Quiz

Translate this ER-diagram to a relational schema



League(name)

JuniorLeague(name, ageLimit)

name -> League.name

Team(name, type, league)

league-> League.name

Player(name, team, type, league, birthYear)

(team,league,type) -> Team.(name,league,type)

1980s

- The database market explodes
- IBM is the leading DBMS vendor (with a system called DB2)
  - Remains leading well into the 2000s
- In 1986, SQL is accepted as an ANSI standard, and in 1987 as an ISO
- In 1983, ACID (Atomicity, Consistency, Isolation, Durability) is coined as a golden standard for Database transaction properties

Quiz

- Explain the concept of a non-repeatable read in terms of isolation, and give an example of a problem this can cause
- Answer: A non-repeatable read occurs when a transaction reads a data item, and some other transaction modifies it while the first transaction is still running (so if it re-reads it, it would get different data)
- Example: Read a value from the database, compute a new value in your application based on the old value, and write the new value back
  - The last step assumes the value has not been modified (or it will overwrite those modifications)

Quiz

- A booking application uses transactions that lists available seats, allows the user to choose one, then books the selected one
- Two schemas are proposed:
  - Book using insert:
    Seats(seatNum)
    BookedSeats(seat)
    seat -> Seats.seatNum

```
INSERT INTO BookedSeats VALUES (?)
```

  - Book by changing boolean attribute:
    Seats(seatNum, isBooked)

```
UPDATE Seats
  SET isBooked=true
  WHERE seatNum=?
```

- Which phenomena are they vulnerable to? Which design is better?
  - The first is vulnerable to non-repeatable reads (seat is removed) and phantoms (seat is booked), the second only to non-repeatable reads
  - The first one is still better, because it will never double book a seat (prevented by primary key constraint: Insert will fail, update will not)

1990s

- The World Wide Web emerges
- Database Driven Web pages become popular
- The first publicly known SQL injection attacks occur ☹
  - Trivia: 20 years later, several groups in this course had SQL injection vulnerabilities in Task 4 – if you have not yet submitted your code, you may want to check for that!
- A standard called ODBC (Open Database Connectivity) is developed to provide an API of functions for applications to interact with databases
  - JDBC is a variant of ODBC specifically for Java

Quiz

- Convert this JDBC code into one that is less vulnerable to injection attacks

```
try (Statement s = conn.createStatement());){
  ResultSet rs = s.executeQuery(
    "SELECT * FROM Grades WHERE student='"+student+"'");
```

- Solution:

```
try (PreparedStatement ps = conn.prepareStatement(
       "SELECT * FROM Grades WHERE student=?");){
  ps.setString(1,student);
  ResultSet rs = ps.executeQuery();
```

Quiz

- Suggest how this JDBC code could be simplified/optimized:

```
try (PreparedStatement ps1 = conn.prepareStatement(
        "SELECT student FROM Grades WHERE course='TDA357'");
     PreparedStatement ps2 = conn.prepareStatement(
        "SELECT name FROM Students WHERE id=?");){
    ResultSet rs1 = ps1.executeQuery();
    while(rs1.next()){
        ps2.setString(rs1.getString(1));
        ResultSet rs2 = ps2.executeQuery();
        rs2.next();
        System.out.println(rs2.getString(1));
    }
}
```

- Solution: Use a single query with a join of Grades/Students on id=student

Students(<u>id</u>,name)
Grades(<u>student</u>,<u>course</u>,grade)
student -> Students.id

1997

- XML is introduced as a universal data language
  - A subset of a previous standard called SGML
  - Intended to be readable by both humans and machines

Quiz

- What is wrong with this XML document?

```
<e><inner><br x="Jonas"/></e></inner>
```

Uses </e> to close <inner>

2000s

- More powerful programming features are used in databases
  - Most notably triggers, but also cascading updates etc.
  - A general trend of moving logic into the Database

- NoSQL is introduced as a concept
- The JSON format is specified and quickly gains momentum

Quiz

- Which ones are <u>not</u> valid JSON documents?
    1. `{"answer":42}`
    2. `["answer",42]`
    3. `["answer":"42"]`
    4. `[{}]`
    5. `{"Jonas"}`
    6. `false`

These ones!

Quiz

```
{"type":"array",
 "items":{"oneOf":
     [{"$ref":"#"},
      {"type":"integer"}
     ]}}
```

- Which JSON documents matches this schema:

1. `5`
2. `[]`
3. `[[]]`
4. `[{}]`
5. `[3,["5"]]`
6. `[3,[4],5]`
7. `[[1],[2]]`

These

2010s

- Gathering and processing data is a booming business
- Technical aspects of databases are evolving at a high pace
- Legal and ethical aspects of the data business are slower, but may be catching up?

2016

- A new revision of the SQL ISO is released
- It standardizes lots of new optional features, most of them related to JSON in some way
- Includes a large part of the JSON Path query language

Quiz

- Write a JSON Path query that collects the grades of all students named Jonas. Expected result for this test data:
  `[{"TDA357":"5"},{"TDA143":"5"}]`

.* is the same as [*]

- Solution: `$.*[?(@.name=="Jonas")].grades.*`

```
{  "s1":{"name":"Jonas",
         "grades":[{"TDA357":"5"}]},
   "s2":{"name":"Jonas",
         "grades":[{"TDA143":"5"}]},
   "s3":{"name":"Sanoj",
         "grades":[{"XYZ123":"4"},{"ABC111":"3"}]},
   "s4":{"name":"Jonas",
         "grades":[]}  }
```

Current challenges in databases

- Dealing with big data: Making systems that can manage petabytes of data for the Web, for bioinformatics, for autonomous vehicles etc.
- Distributed databases: Making databases that are split up or mirrored across lots of machines across the world, and still maintaining a level of consistency
- Why are people still enabling SQL injection attackers?!
    - Maybe a question better suited for a behavioral scientist…

The (very) near future: Finish up Assignments

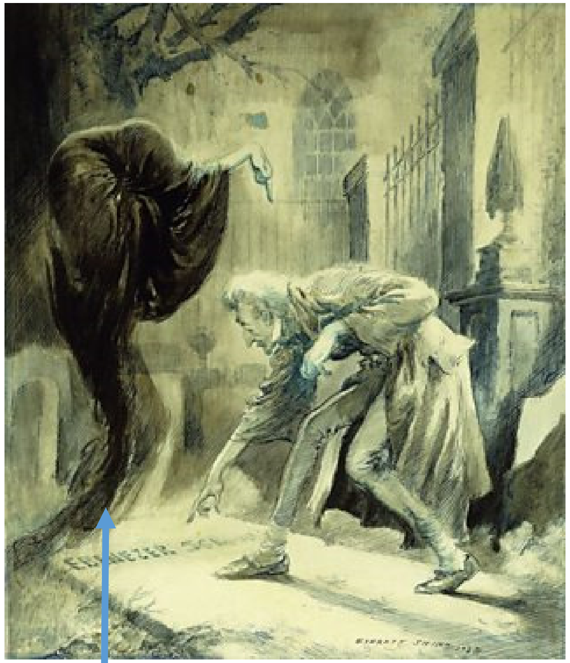
- Please try to demonstrate tomorrow if you have not already demonstrated
- IF YOU NEED TO DEMONSTRATE ON FRIDAY:
  - Sign up for either Johanneberg or Lindholmen in Waglys (the queue system) before Thursday 15.00
  - This is to help us allocate teaching resources between campuses
- Note: if you demonstrate on Friday, there is a very real risk that your demo is unsuccessful and you will not have time to fix whatever is wrong
- If you submit any task tomorrow or later and it, you may not have time to resubmit it before the final deadline if it is rejected

The still pretty near future: Exam

- The exam is on January 16
- Remember: You are allowed to bring one A4 of <u>hand-written</u> notes to the exam
- You will also be given a standard reference that you can find on the course page
  - The reference is printed with the exam, you are <u>not</u> allowed to bring it
  - Updated yesterday (replaced XML with JSON)
  - Small additions may be made if I think of anything while making the exam
  - Multivalued Dependencies and 4NF will not be on the January exam, but is in the reference
- If there is interest, I might organize some exam practice in early January?

The more distant future(?)

- The future is always the hardest thing to predict
- Maybe in a few years, one of you will have solved some of the challenges i presented earlier?
- If you do I will be sure to update my slides, adding your name to them



The Ghost of Christmas Yet To Come

This is the end of the last lecture

Thank you for listening!