CS 5530

Database Systems
Spring 2020

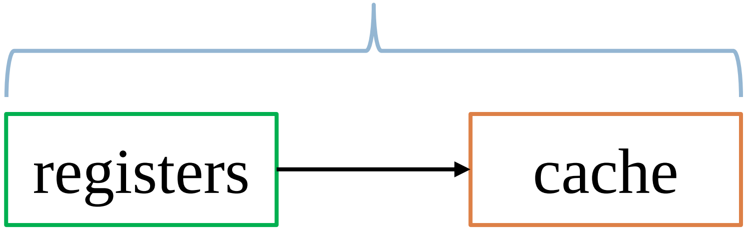
Welcome!

Course Overview

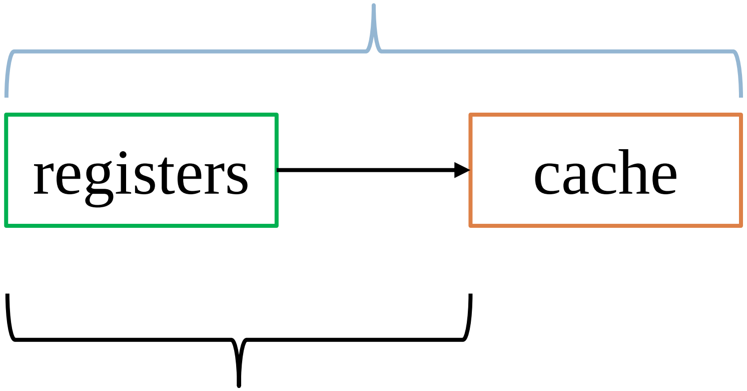
Intro to Databases

Domains of Data

Domains of Data


registers → cache

Domains of Data

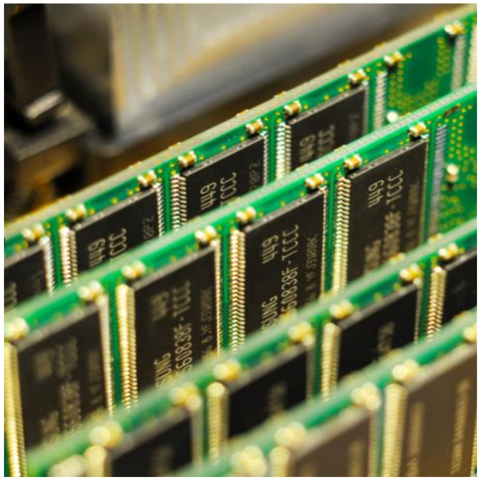


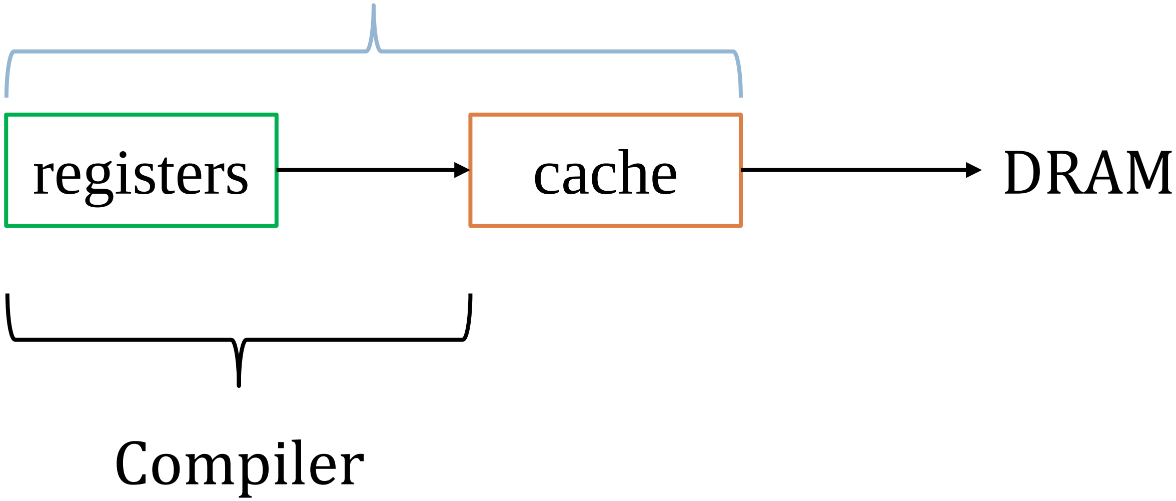
registers

cache

Compiler

Domains of Data





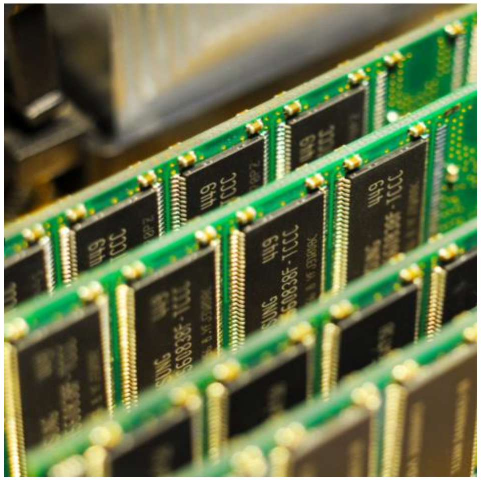
registers

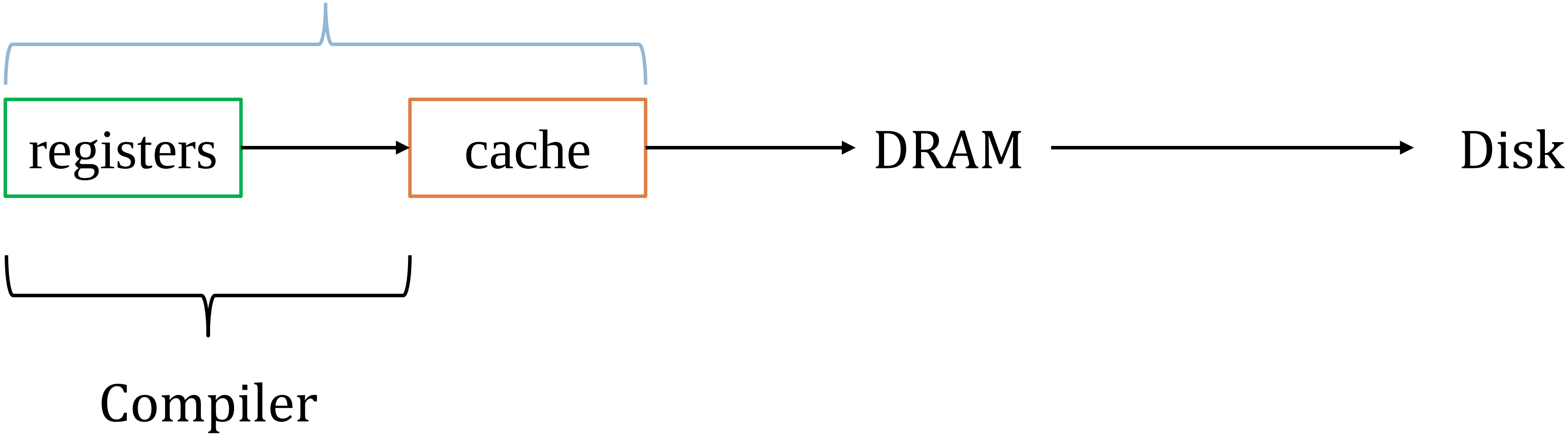
cache

DRAM

Compiler

Domains of Data

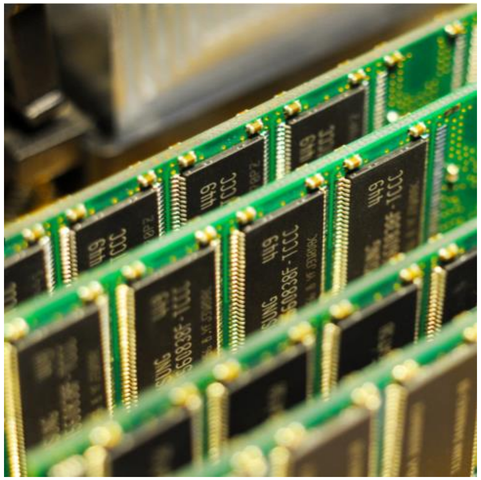


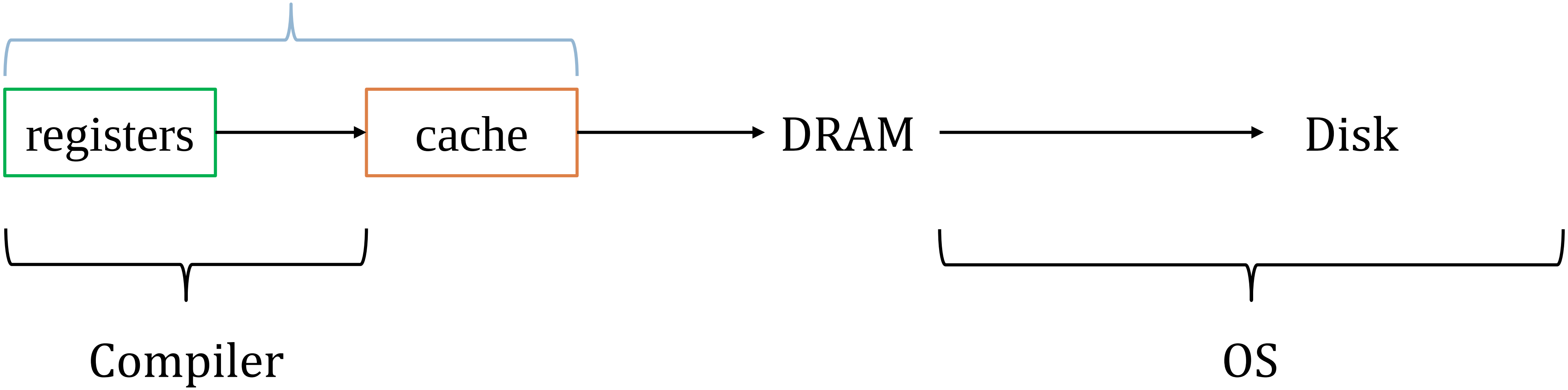


registers → cache → DRAM → Disk

Compiler

Domains of Data



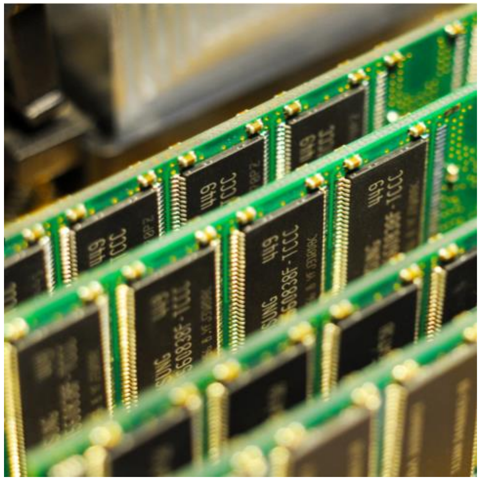


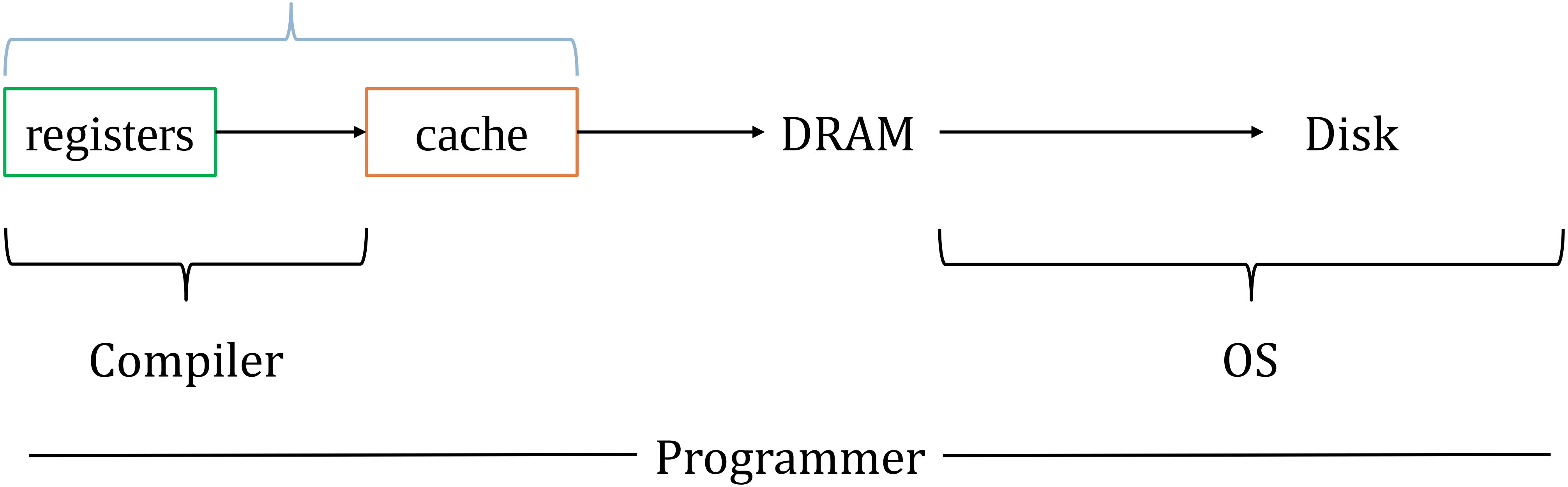
registers → cache → DRAM → Disk

Compiler

OS

Domains of Data





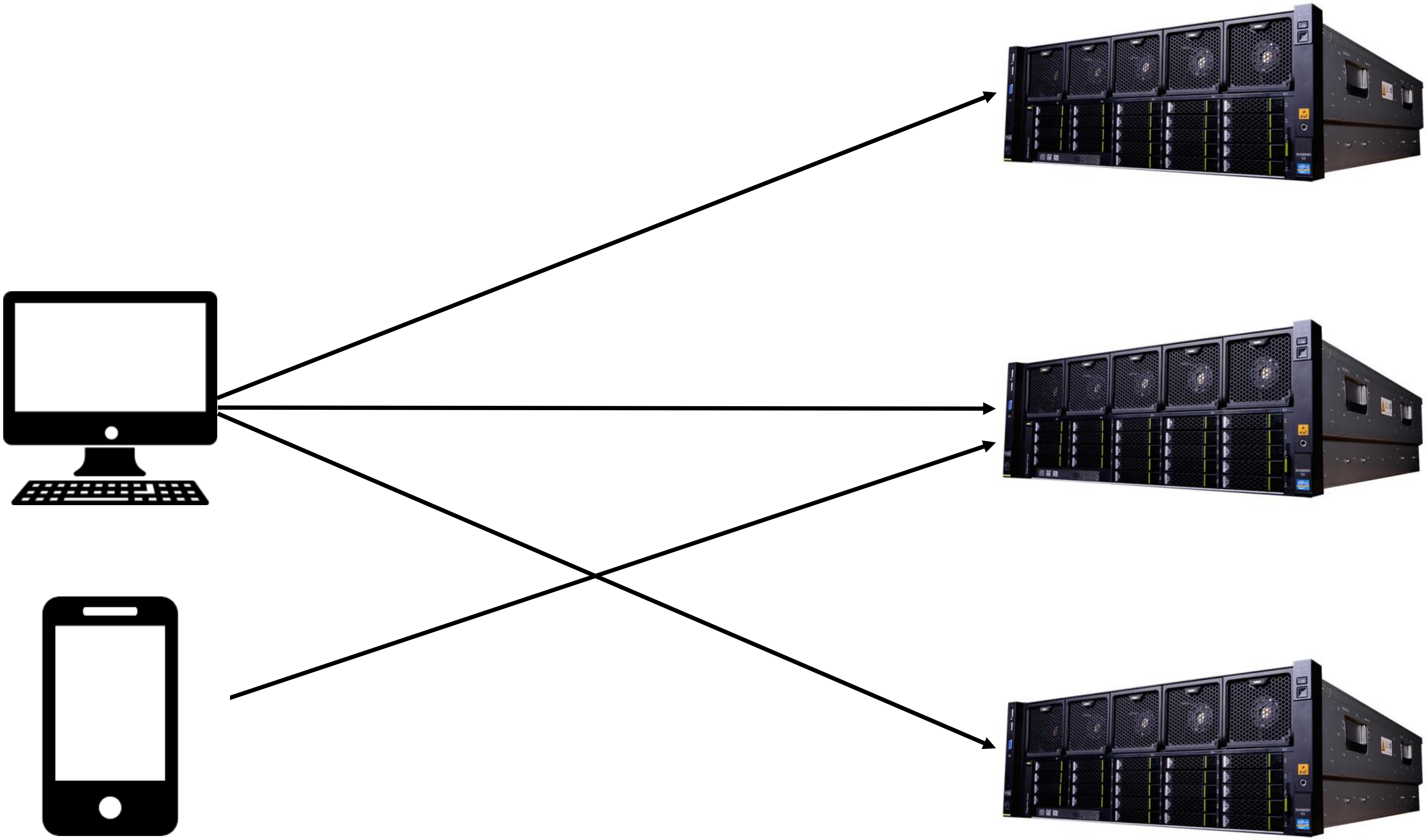
registers → cache → DRAM → Disk

Compiler

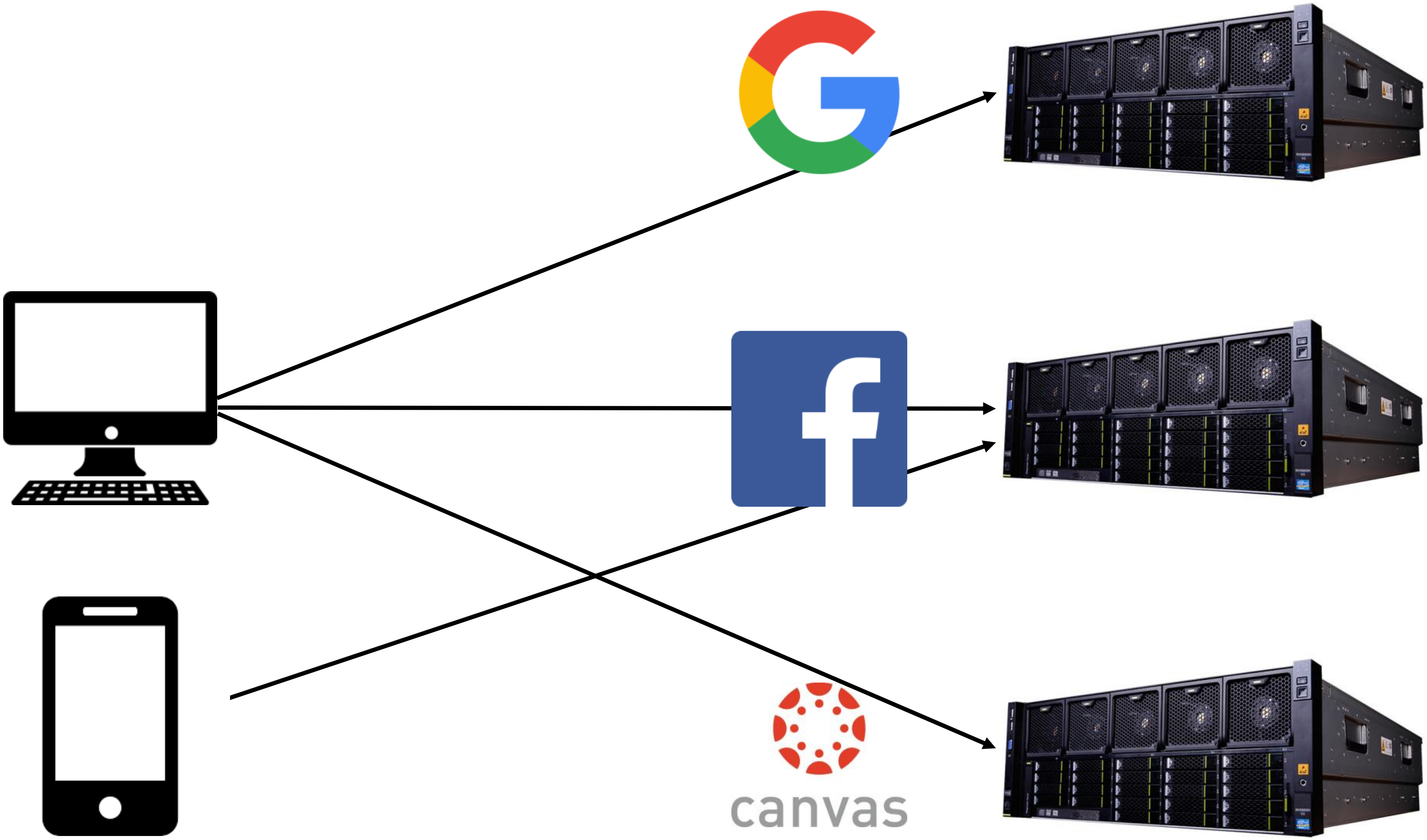
OS

Programmer

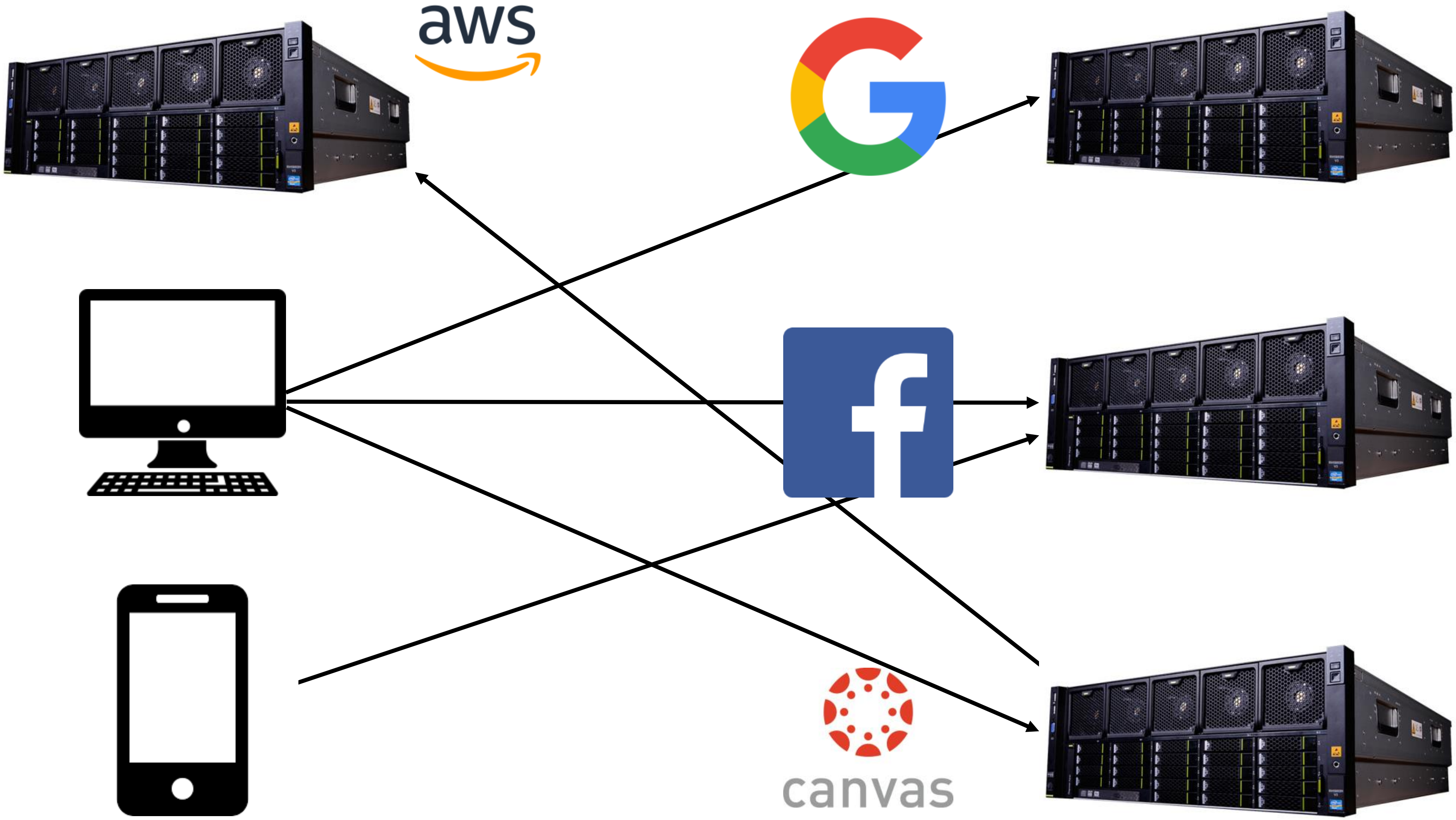
Domains of Data



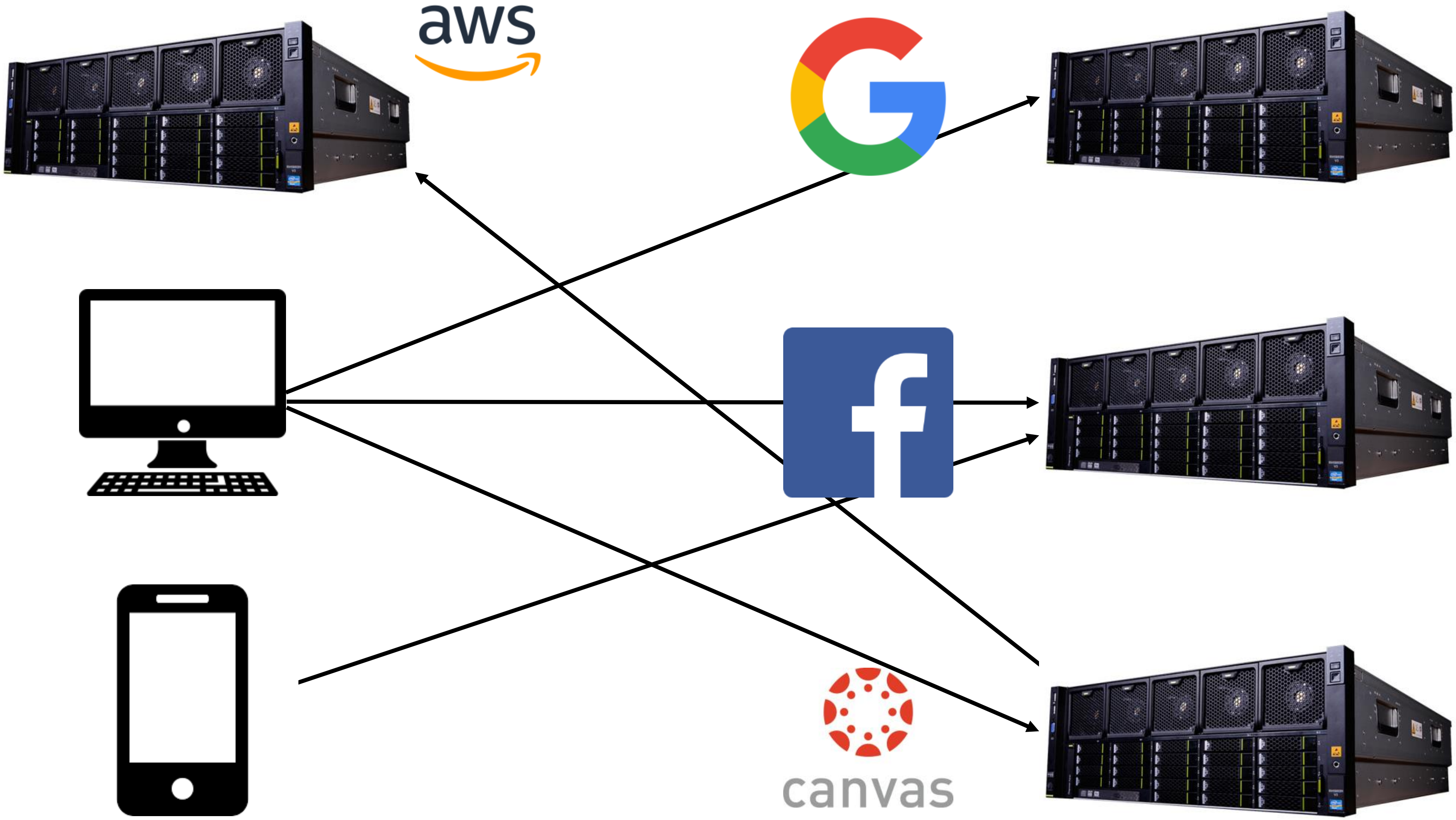
Domains of Data



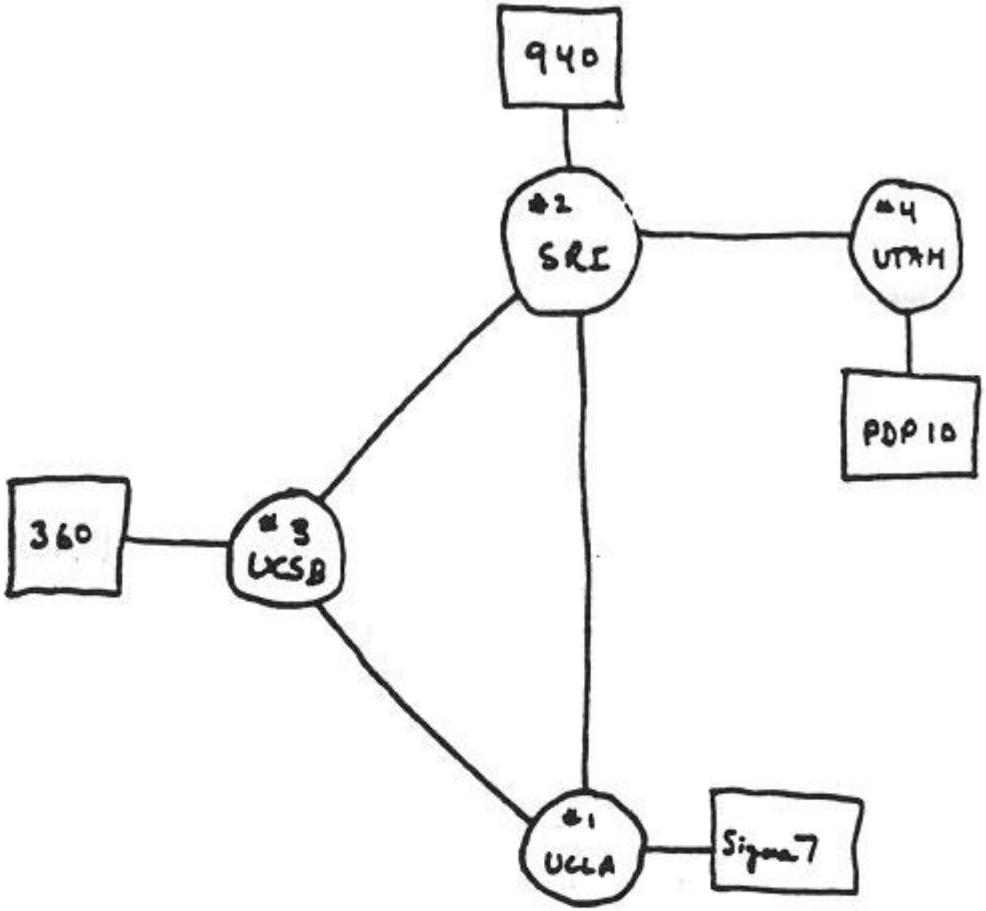
Domains of Data



Domains of Data



Programmer



THE ARPA NETWORK

DEC 1969

4 NODES

Data Driven

- This end of the spectrum has different challenges:
  - **Vast amounts of data**
  - **Fast access/combination/filtering**
  - **Must be available**
    - Online
    - Securely
    - To simultaneous users

Ponder…

- Suppose you want to save a bunch of Students to a file

- Option 1:

"Jane Doe is a Film major with a GPA of 3.7, and is enrolled in CS2420, and her ID is 12345

John Smith is …"

Ponder…

- Suppose you want to save a bunch of Students to a file

- Option 1:

"Jane Doe is a Film major with a GPA of 3.7, and is enrolled in CS2420, and her ID is 12345

John Smith is …"

- How do we search for a student?

Ponder…

- Suppose you want to save a bunch of Students to a file

- Option 1:

"Jane Doe is a Film major with a GPA of 3.7, and is enrolled in CS2420, and her ID is 12345

John Smith is …"

- How do we search for a student?
  - First we have to know the data's format
  - O(N) scan of entire file

Representing Data

- Option 2: JSON-like (self-describing data)

Major: Film
Class: CS2420
Name: Jane Doe
GPA: 3.7
ID: 12345

Representing Data

- Option 2: JSON-like (self-describing data)

Major: Film
Class: CS2420
Name: Jane Doe
GPA: 3.7
ID: 12345

- How do we find all students enrolled in 2420?
    - linear scan

How About XML?

```
<Course>
  <Name>CS2420</Name>
  <Students>
    <Student>
      <Name>Jane Doe</Name>
      <Major>Film</Major>
    </Student>
    <Student>
      <Name>John Smith</Name>
      <Major>CS</Major>
    </Student>
  </Students>
</Course>
```

Still not scalable!

Exercise

- Store a bunch of student records by name, and quickly
  - Add
  - Remove
  - Search / range query
  - Enumerate

Exercise

- Store a bunch of student records by name, and quickly
  - Add
  - Remove
  - Search / range query
  - Enumerate
- Binary search tree

Exercise

- Store a bunch of:
  - students
  - courses
  - professors

Exercise

Professors

Teaching: CS5530, CS4400
Name: Daniel Kopta
ID: 55555

Teaching: CS3500, CS4150
Name: Joe Zachary
ID: 44444

Students

Classes: CS5530, Phys2010
Name: Jane Doe
GPA: 3.7
ID: 12345

Classes: CS3500, FILM1010
Name: Jon Smith
GPA: 3.4
ID: 12421

Courses

Name: Database Systems
Num: 5530
Dept. CS

Name: Software Practice
Num: 3500
Dept. CS

Exercise

- All courses student *Y* is enrolled in?
- All teachers of student *Z*?
- Order courses by enrollment number?

Professors

Teaching: CS5530, CS4400
Name: Daniel Kopta
ID: 55555

Teaching: CS3500, CS4150
Name: Joe Zachary
ID: 44444

Students

Classes: CS5530, Phys2010
Name: Jane Doe
GPA: 3.7
ID: 12345

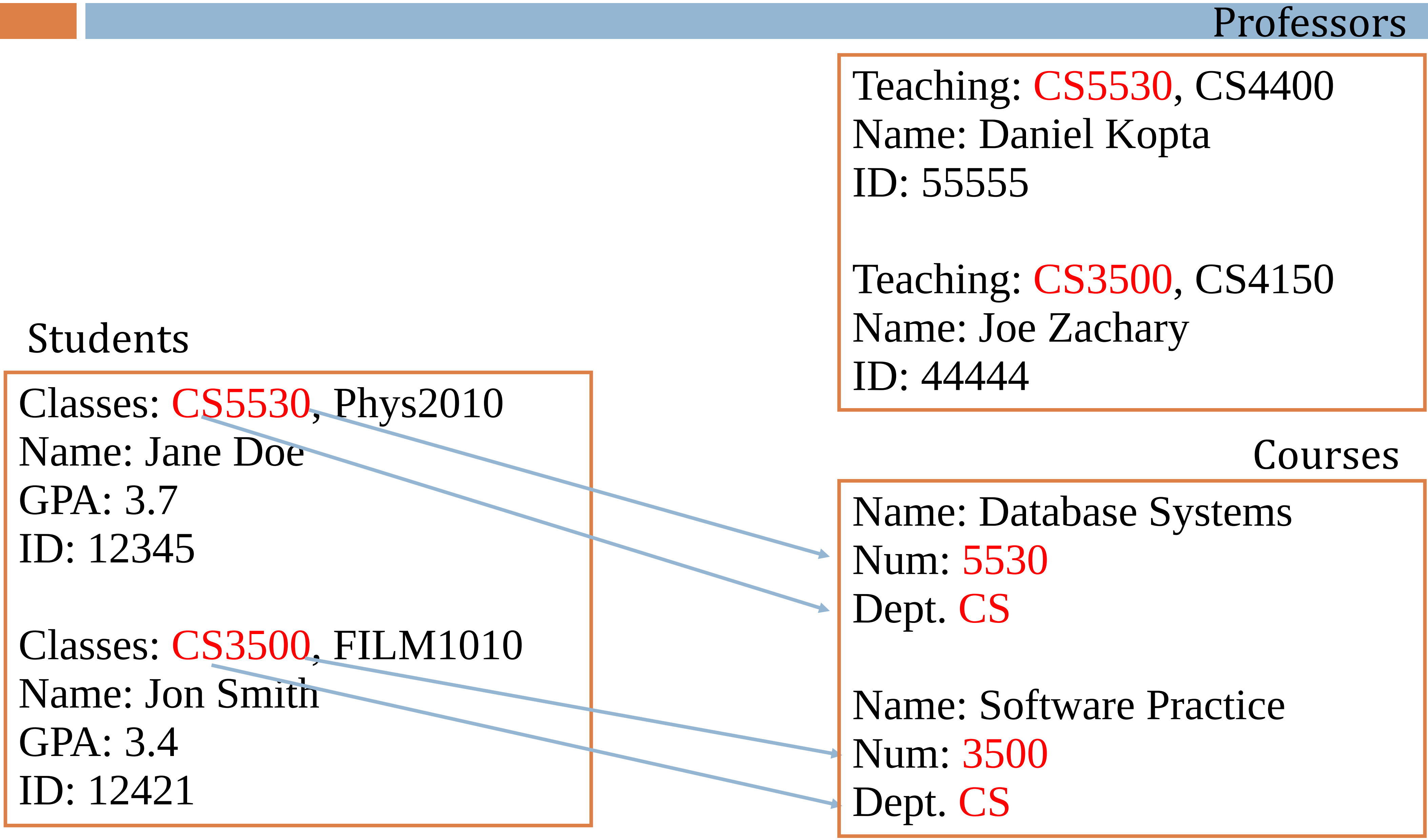
Classes: CS3500, FILM1010
Name: Jon Smith
GPA: 3.4
ID: 12421

Courses

Name: Database Systems
Num: 5530
Dept. CS

Name: Software Practice
Num: 3500
Dept. CS

Exercise



Professors

Teaching: CS5530, CS4400
Name: Daniel Kopta
ID: 55555

Teaching: CS3500, CS4150
Name: Joe Zachary
ID: 44444

Students

Classes: CS5530, Phys2010
Name: Jane Doe
GPA: 3.7
ID: 12345

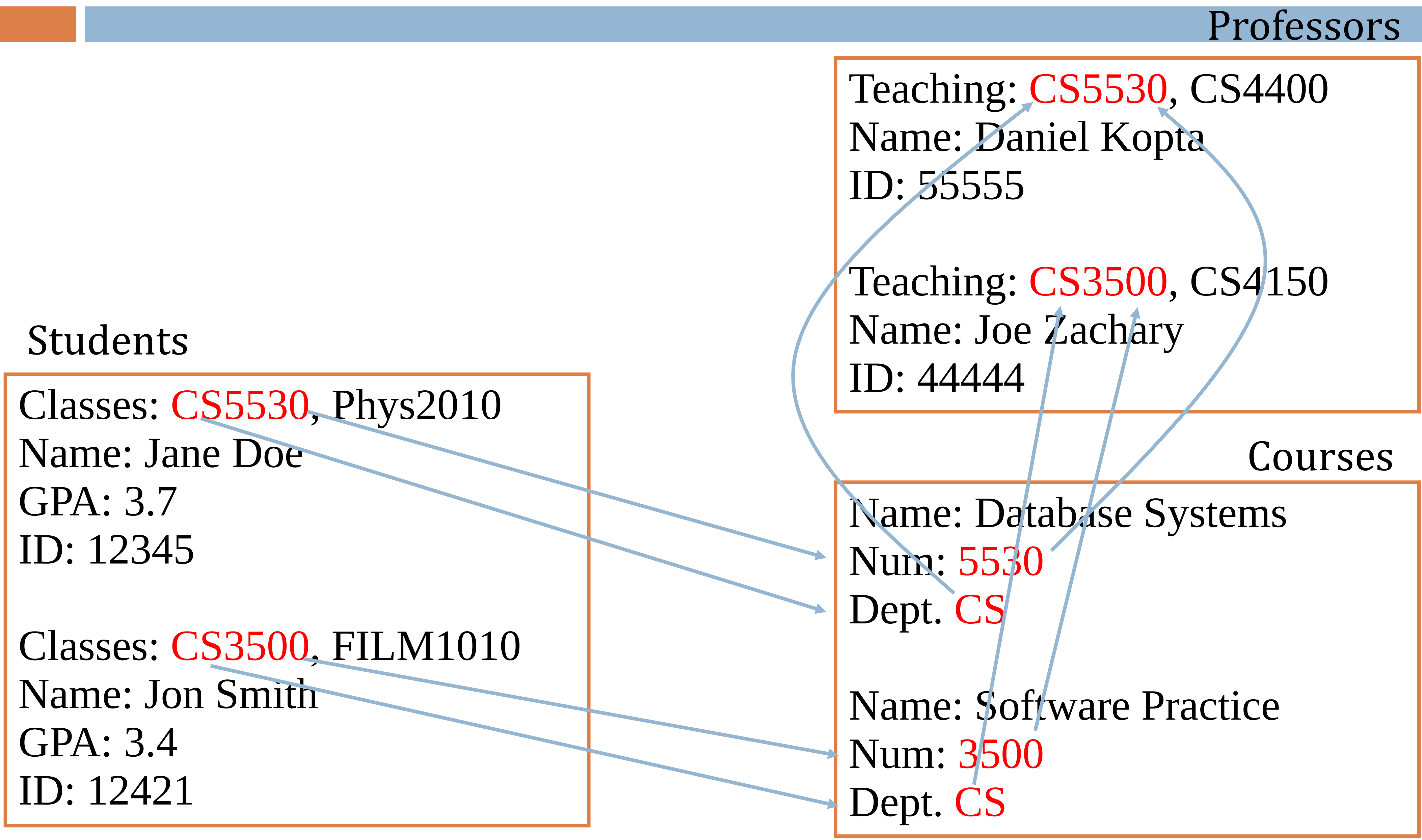
Classes: CS3500, FILM1010
Name: Jon Smith
GPA: 3.4
ID: 12421

Courses

Name: Database Systems
Num: 5530
Dept. CS

Name: Software Practice
Num: 3500
Dept. CS

Exercise



Professors

Teaching: CS5530, CS4400
Name: Daniel Kopta
ID: 55555

Teaching: CS3500, CS4150
Name: Joe Zachary
ID: 44444

Students

Classes: CS5530, Phys2010
Name: Jane Doe
GPA: 3.7
ID: 12345

Classes: CS3500, FILM1010
Name: Jon Smith
GPA: 3.4
ID: 12421

Courses

Name: Database Systems
Num: 5530
Dept. CS

Name: Software Practice
Num: 3500
Dept. CS

Exercise

- How can we quickly
  - Find all students in course *X*?
  - Find all course(s) student *Y* is enrolled in?
  - Find all teachers of student *Z*?
  - Order students by GPA?
  - Order courses by enrollment number?
  - …

- Now imagine there are **millions** of each

  - And these operations happen frequently

Solution

1. Structured data

2. Data structures

3. Query language

Solution

- *Structured data*

  - Records can not have arbitrary/unpredictable fields/values

  - e.g. courses have: dept, num, and name
    (string) (int) (string)

Structured Data

- Unstructured

  Jane Doe is a Film major with a GPA of 3.7, and her ID is 12345

- Structured

  Name (string): “Jane Doe”
  Major (string): “Film”
  GPA (float): 3.7
  ID (uint): 12345

Data Storage

- Save the data itself + data structures
  - Trees, hash tables, etc…

Structured Data

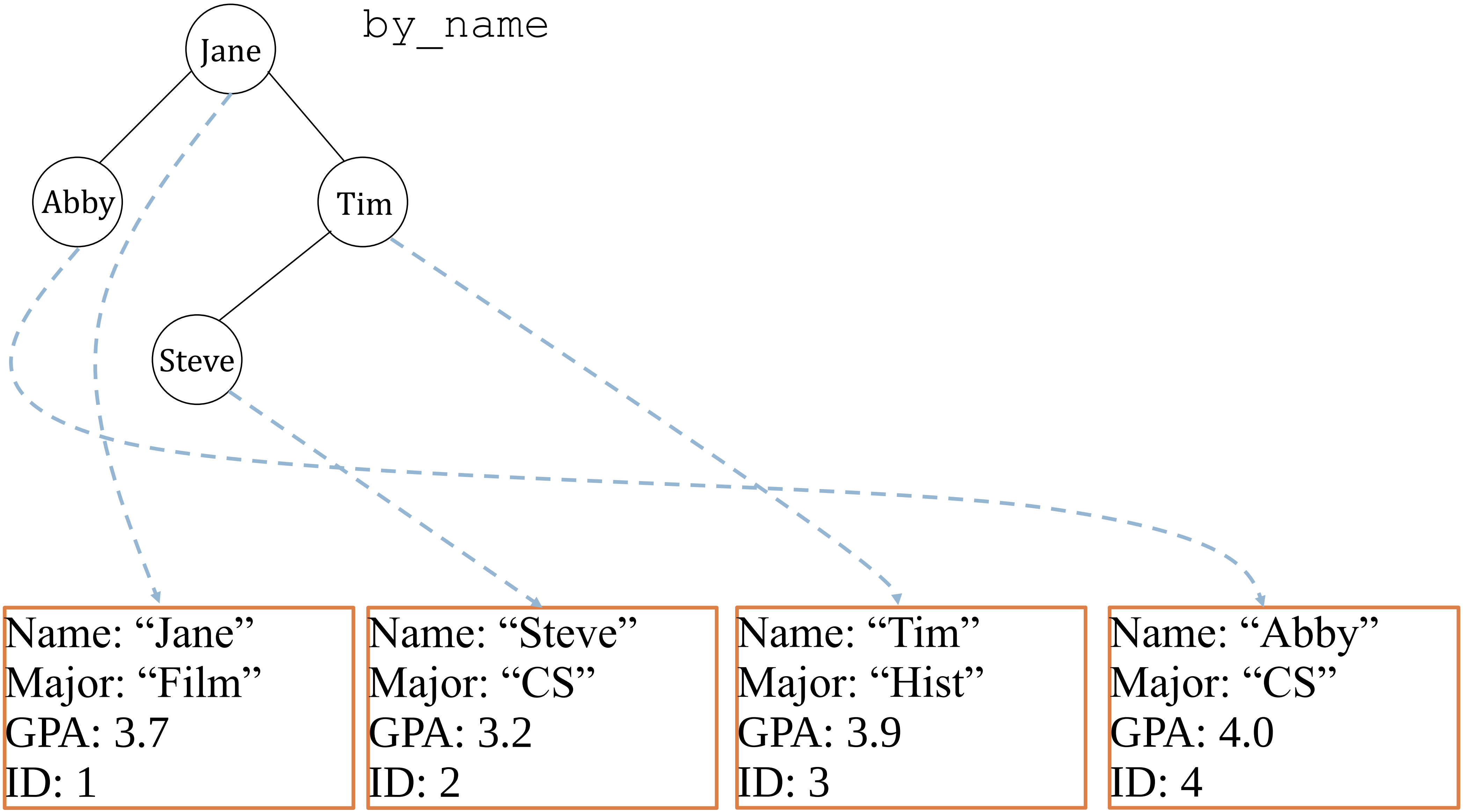
Name: “Jane”
Major: “Film”
GPA: 3.7
ID: 1

Name: “Steve”
Major: “CS”
GPA: 3.2
ID: 2

Name: “Tim”
Major: “Hist”
GPA: 3.9
ID: 3

Name: “Abby”
Major: “CS”
GPA: 4.0
ID: 4

Structured Data + Data Structure



by_name

Jane

Abby

Tim

Steve

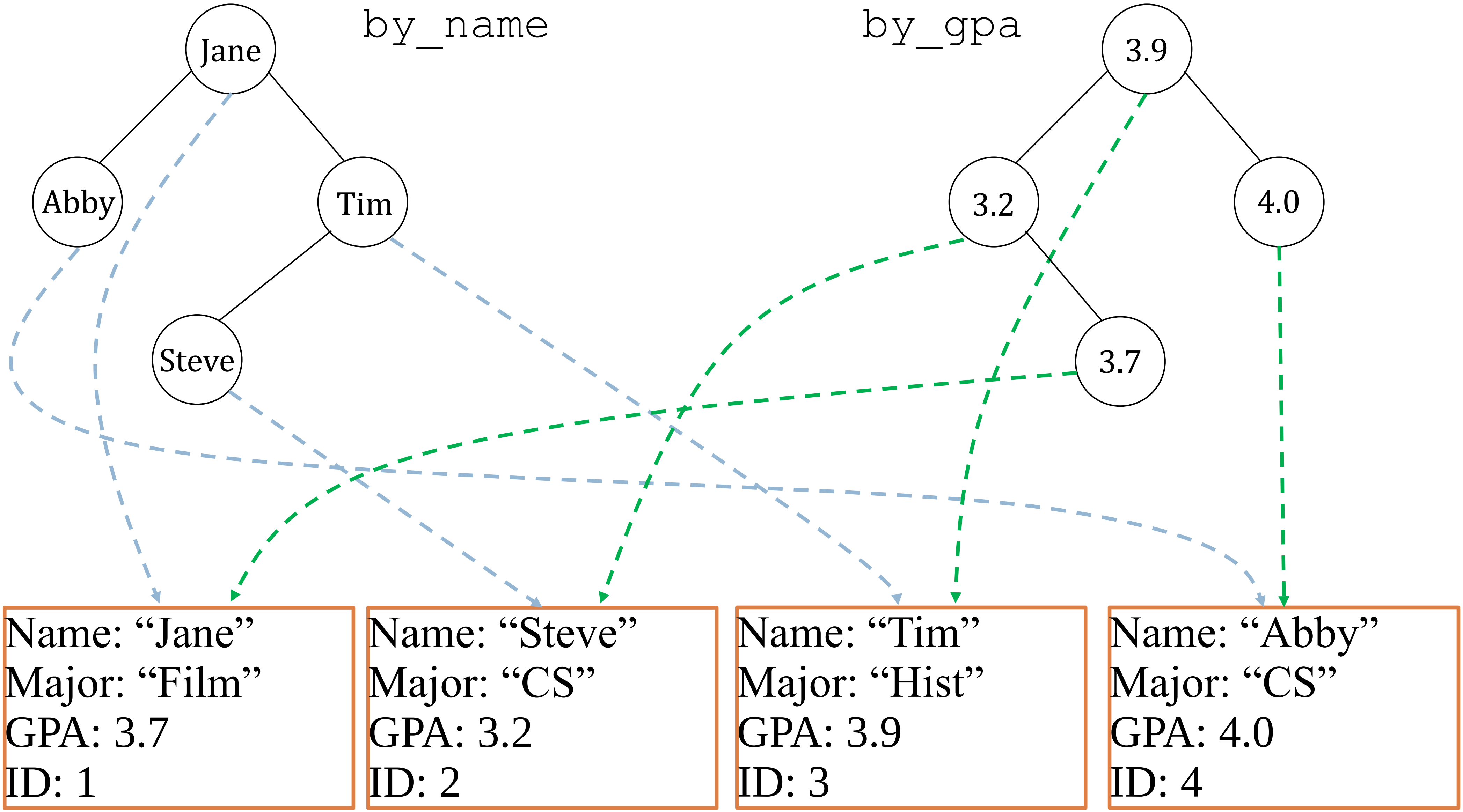
Name: “Jane”
Major: “Film”
GPA: 3.7
ID: 1

Name: “Steve”
Major: “CS”
GPA: 3.2
ID: 2

Name: “Tim”
Major: “Hist”
GPA: 3.9
ID: 3

Name: “Abby”
Major: “CS”
GPA: 4.0
ID: 4

Structured Data + Data Structure



Name: "Jane"
Major: "Film"
GPA: 3.7
ID: 1

Name: "Steve"
Major: "CS"
GPA: 3.2
ID: 2

Name: "Tim"
Major: "Hist"
GPA: 3.9
ID: 3

Name: "Abby"
Major: "CS"
GPA: 4.0
ID: 4

Exercise

- Language for expressing:
  - Find all students in course *X*?
  - Find all course(s) student *Y* is enrolled in?
  - Find all teachers of student *Z*?
  - Order students by GPA?
  - Order courses by enrollment number?
  - …

- C++, Java, C# etc…?

Solution

- Devise language for combining/filtering data

```
SELECT Name FROM Students
WHERE GPA > 3.5;
```

Solution

1. Structured data
2. Data structures
3. Query language

- …this is exactly what a *database system* does for you
    - Plus much more!

Why Databases?

- Take advantage of decades of research
  - Availability
  - Reliability
  - Performance
  - Concurrency
  - Interface

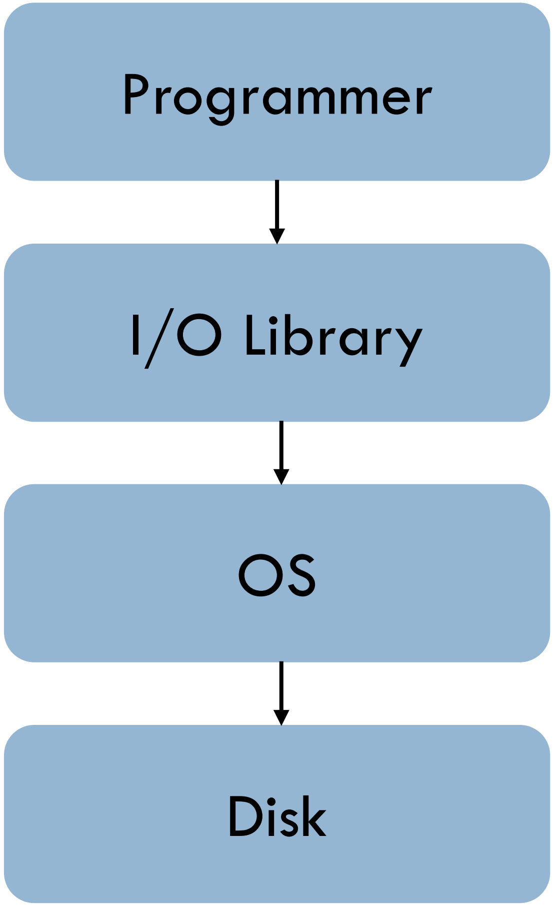
- Don’t reinvent the wheel

Database System

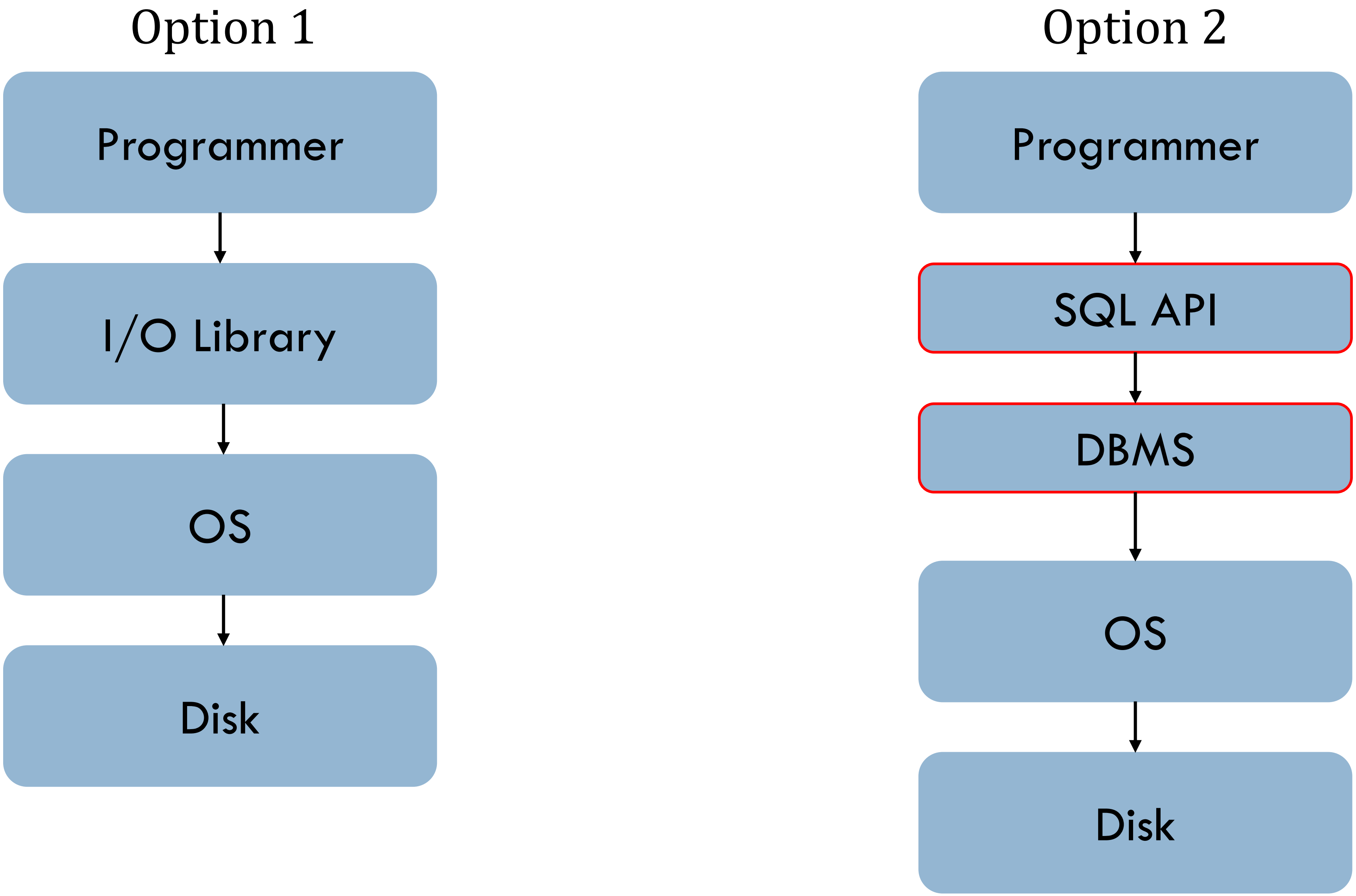
- Two major components:
  - **Database Management System (DBMS)**
    - Underlying machinery
  - **Query Language**
    - Common interface

Data Storage

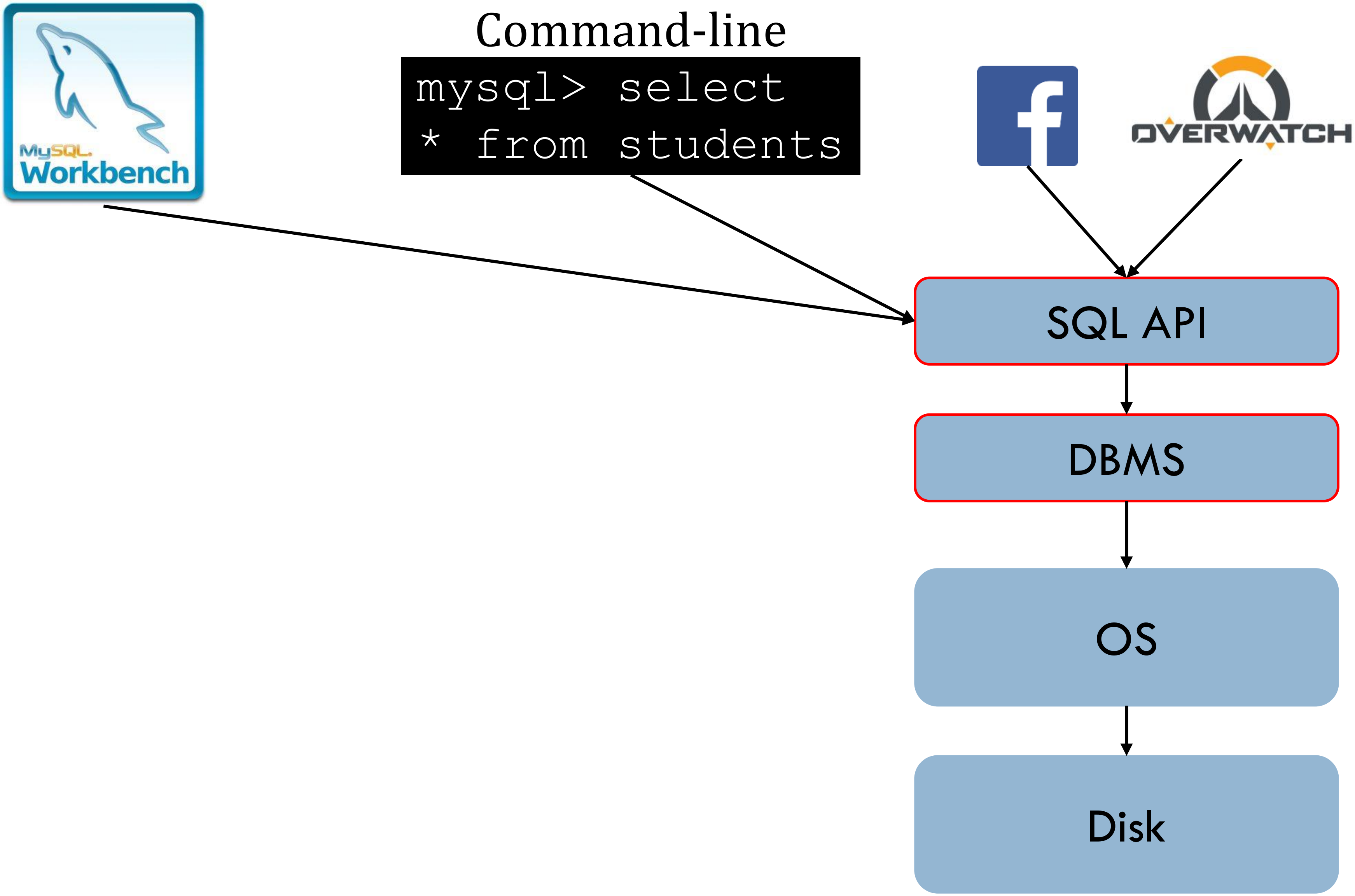
Option 1



Data Storage



Data Storage



Command-line

```
mysql> select
* from students
```

SQL API

DBMS

OS

Disk

This Class

- Principles of structured information storage
- Application of databases
- Software interfacing with databases
- Understand DBMS enough to use it effectively

Syllabus

- The syllabus online has been updated – please download a new copy!

Assignments

- A mixture of:
  - Programming
  - Written/diagram
  - Database manipulation

Server

- `atr.eng.utah.edu`
  - You all have your own MySQL database on this server

Project

- Implement your own Canvas-like system
  - UI is provided for you
  - You implement the back-end
- Multiple phases throughout the semester

Exams

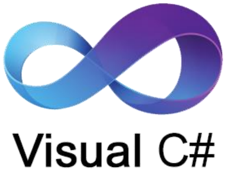
- One midterm, one final exam
- 5 pages of notes front + back

Grading

•Project:	30%
•Assignments:	30%
•Midterm:	20%
•Final:	20%

Tools

- Visual Studio (Windows)
- C#
- MySQL
- Linux



Windows

- Windows via Bootcamp recommended
  - Give it at least 80GB
- Parallels *should* also work

Visual Studio

- You will need Visual Studio 2019
  - Don’t forget to install .NET
- See Canvas instructions

Class Web Page

- All relevant materials and assignments

utah.instructure.com

- Discuss strategies with other students

- Course announcements

- Grades

Getting Help

- Please ask for help!
- Office hours TBD
- TA consulting hours TBD
- Piazza forums

Academic Misconduct

- Taken very seriously by the SoC

- Any cheating whatsoever results in a failing grade

- Cheating policy is online

- Caught 27 students last year

Relational Databases

- Structured data storage
- Data is organized into relations
- Related data are stored “next to” each other
  - e.g. in a table

ID	Name	DOB
…	…	…
…	…	…

Tables

- Database comprised of one or more tables

- One table represents one relation
  - (pieces of directly-related data)

ID	Name	DOB
1	Harry	31 JUL 1980
2	Hermione	19 SEP 1979
3	Ron	01 MAR 1980
4	Malfoy	05 JUN 1980

Multiple Tables

- Non directly-related data are separated

People

ID	Name	DOB
1	Harry	31 JUL 1980
2	Hermione	19 SEP 1979
3	Ron	01 MAR 1980
4	Malfoy	05 JUN 1980

Courses

Course Num	Name
2420	Alg. and DS
3500	SW Practice
3810	Architecture
4400	Systems
5530	Databases

- Each table is a “relation”

Relation (table)

- Each row is a *tuple* – a set of data units

ID	Name	DOB	GPA
1	Harry	31 JUL 1980	3.5
2	Hermione	19 SEP 1979	4.0
3	Ron	01 MAR 1980	4.0
4	Malfoy	05 JUN 1980	3.9

Relation (table)

- Each row is a *tuple* – a set of data units
  - Does every cell need to be unique?

ID	Name	DOB	GPA
1	Harry	31 JUL 1980	3.5
2	Hermione	19 SEP 1979	4.0
3	Ron	01 MAR 1980	4.0
4	Malfoy	05 JUN 1980	3.9

Relation (table)

- Each row is a *tuple* – a set of data units
  - Does every cell need to be unique? No

ID	Name	DOB	GPA
1	Harry	31 JUL 1980	3.5
2	Hermione	19 SEP 1979	4.0
3	Ron	01 MAR 1980	4.0
4	Malfoy	05 JUN 1980	3.9

Relation (table)

- Each row is a *tuple* – a set of data units
  - Does every *row* need to be unique?

ID	Name	DOB	GPA
1	Harry	31 JUL 1980	3.5
2	Hermione	19 SEP 1979	4.0
3	Ron	01 MAR 1980	4.0
4	Malfoy	05 JUN 1980	3.9

Relation (table)

- Each row is a *tuple* – a set of data units
  - Does every *row* need to be unique? Yes

ID	Name	DOB	GPA
1	Harry	31 JUL 1980	3.5
2	Hermione	19 SEP 1979	4.0
3	Ron	01 MAR 1980	4.0
4	Malfoy	05 JUN 1980	3.9

Library Example

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003	Dune

Library Example

- Is Malfoy directly-related to Dune?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003	Dune

Library Example

- Is Malfoy directly-related to Dune?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003	Dune

- No; only indirectly (he has it checked out)

Library Example

- What if one person checks out multiple books?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003	Dune

Library Example

- What if one person checks out multiple books?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003	Dune
Malfoy	765-4321	623	978-004	Hyperion
Malfoy	765-4321	623	978-005	Bunny Meadows

Library Example

- What if one person checks out multiple books?
  - Duplicate data

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003	Dune
Malfoy	765-4321	623	978-004	Hyperion
Malfoy	765-4321	623	978-005	Bunny Meadows

Library Example

- What if one person checks out multiple books?
  - Make a list?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003, 978-004, 978-005	Dune, Hyperion, …

Library Example

- What if one person checks out multiple books?
  - How to find all people who have checked out “Dune”?
  - How to represent a book that isn’t checked out?
  - How much space to allocate for a row?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Hermione	555-1234	124	978-001	A Tale of Two Cities
Ron	123-4567	228	978-002	Last of Us
Malfoy	765-4321	623	978-003, 978-004, 978-005	Dune, Hyperion, …

Library Example

- What if one person checks out no books?

Name	Phone	CardNum	ISBN	Book
Harry	123-1123	123	978-000	Harry Potter
Malfoy	765-4321	623	???	???

Even Worse

- Multiple phone numbers, multiple checkouts

Name	Phone	CardNum	ISBN	Title
Dan	888-8888	4	003	Dune
Dan	999-9999	4	003	Dune
Dan	888-8888	4	004	Hyperion
Dan	999-9999	4	004	Hyperion

Solution

- First, let’s fix the unrelated-data problem

Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

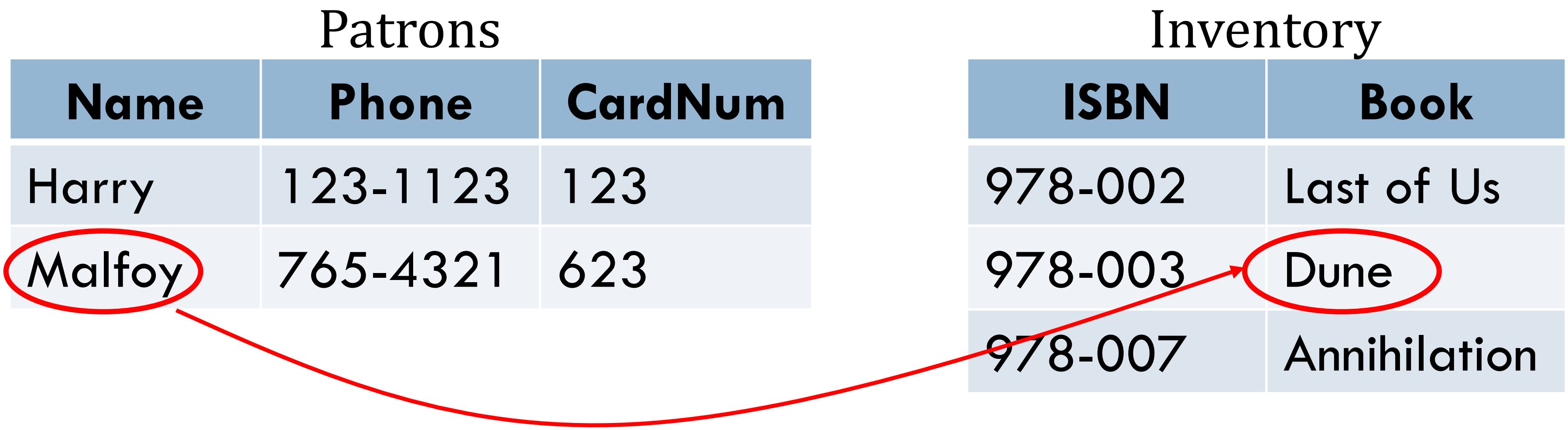
Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

Solution

- First, let’s fix the unrelated-data problem

- But what about indirect relationships?
  - How do we specify Malfoy checked out Dune?



Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

Solution

- Add a table that relates the two

Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

CheckedOut

CardNum	ISBN
123	978-002
623	978-003

Solution

- Add a table that relates the two

Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

CheckedOut

CardNum	ISBN
123	978-002
623	978-003

Solution

- Multiple checkouts
  - Duplicate data minimized

Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

CheckedOut

CardNum	ISBN
123	978-002
123	987-007
623	978-003

Solution

- No checkouts

Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

CheckedOut

CardNum	ISBN
623	978-003

Solution

- What about multiple phone numbers?

Patrons

Name	Phone	CardNum
Harry	123-1123	123
Malfoy	765-4321	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

CheckedOut

CardNum	ISBN
623	978-003

Solution

- What about multiple phone numbers?

Patrons

Name	~~Phone~~	CardNum
Harry	~~123-1123~~	123
Malfoy	~~765-4321~~	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

Phones

CardNum	Phone
123	123-1123
123	555-5555
623	765-4321

CheckedOut

CardNum	ISBN
623	978-003

Solution

- Pick some unique ID-like field to relate tables (key)
  - CardNum and ISBN

Patrons

Name	CardNum
Harry	123
Malfoy	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

Phones

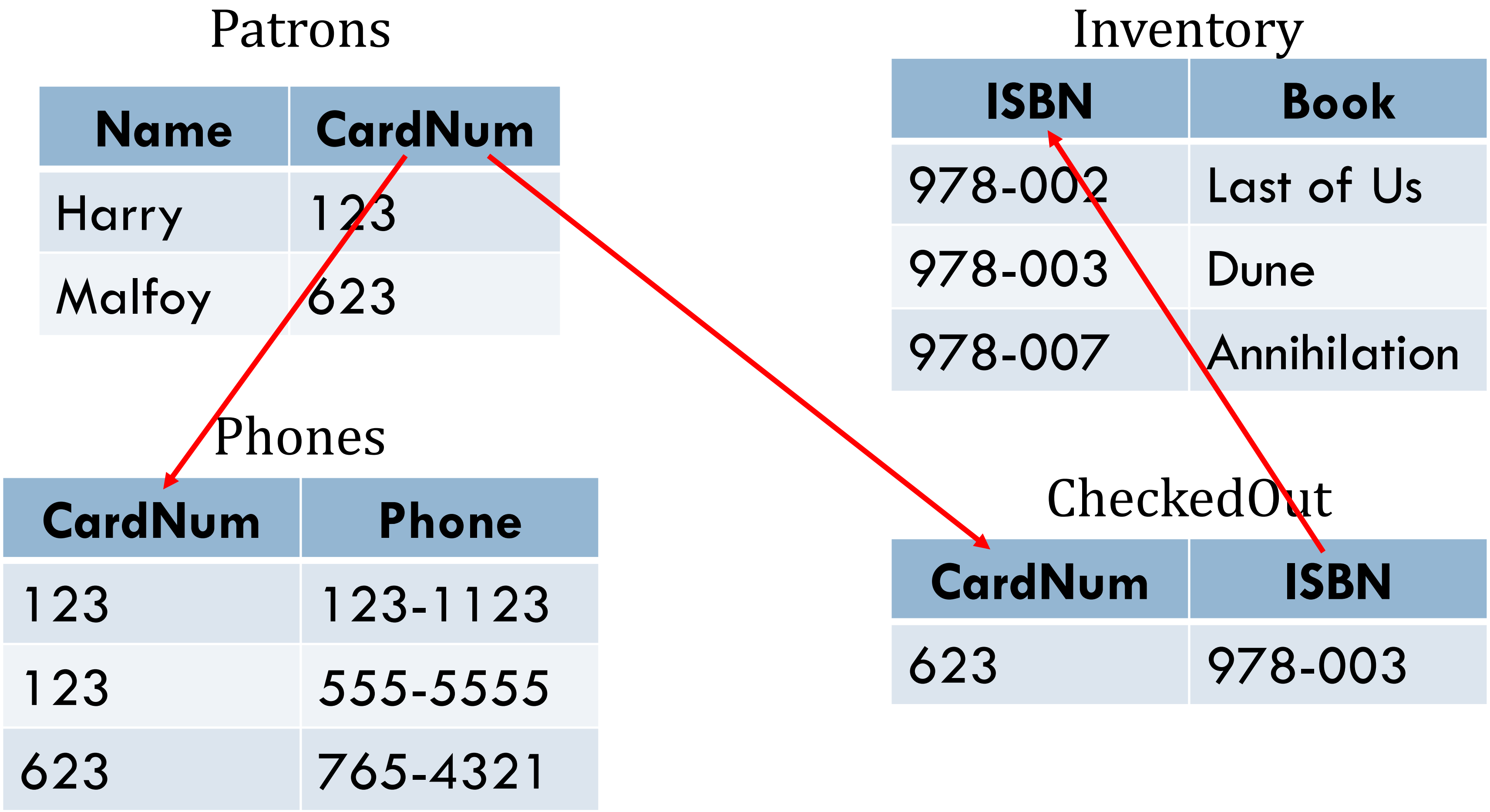
CardNum	Phone
123	123-1123
123	555-5555
623	765-4321

CheckedOut

CardNum	ISBN
623	978-003

Solution

- Pick some unique ID-like field to relate tables (key)
  - CardNum and ISBN



Patrons

Name	CardNum
Harry	123
Malfoy	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

Phones

CardNum	Phone
123	123-1123
123	555-5555
623	765-4321

CheckedOut

CardNum	ISBN
623	978-003

Solution

- What if we have multiple copies of the same book?

Patrons

Name	CardNum
Harry	123
Malfoy	623

Phones

CardNum	Phone
123	123-1123
123	555-5555
623	765-4321

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

CheckedOut

CardNum	ISBN
623	978-003

Solution

- What if we have multiple copies of the same book?
  - Make another table!

Patrons

Name	CardNum
Harry	123
Malfoy	623

Inventory

ISBN	Book
978-002	Last of Us
978-003	Dune
978-007	Annihilation

Phones

CardNum	Phone
123	123-1123
123	555-5555
623	765-4321

CheckedOut

CardNum	ISBN
623	978-003

Library

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005

Phones

CardNum	Phone
1	555-5555
2	666-6666
4	777-7777

How do we find all the books checked out by Joe?

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

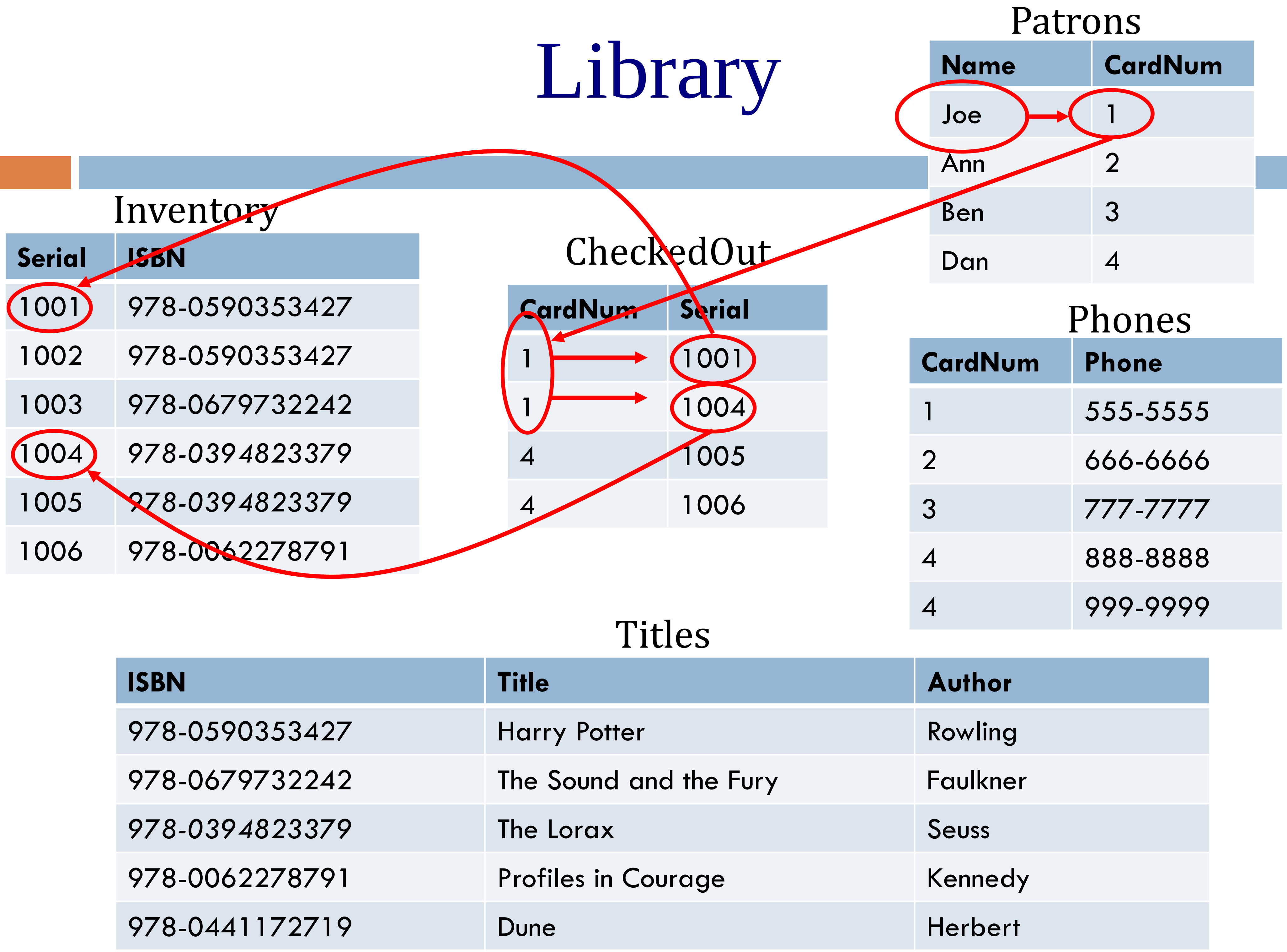
Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library



Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

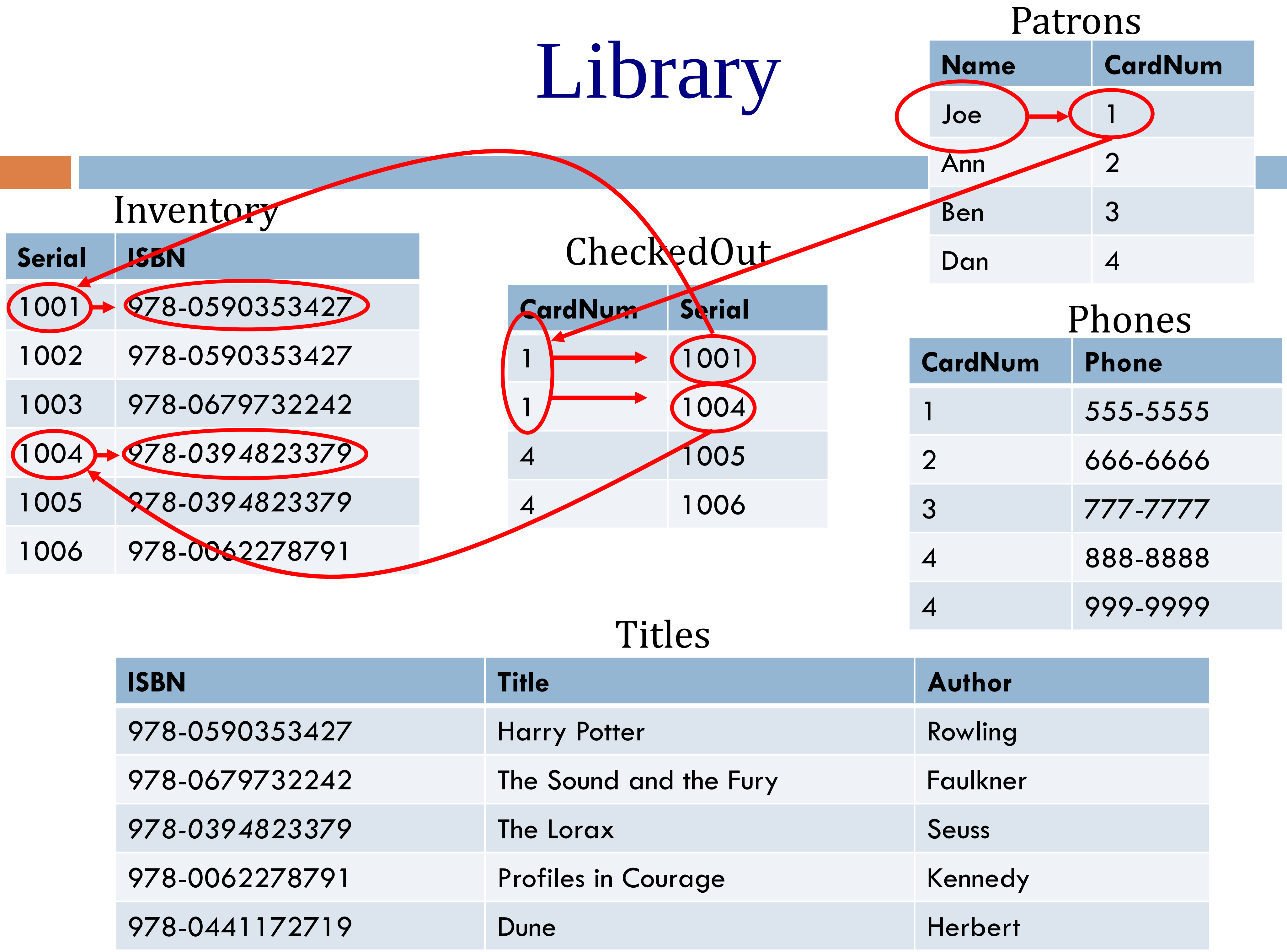
Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library



Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

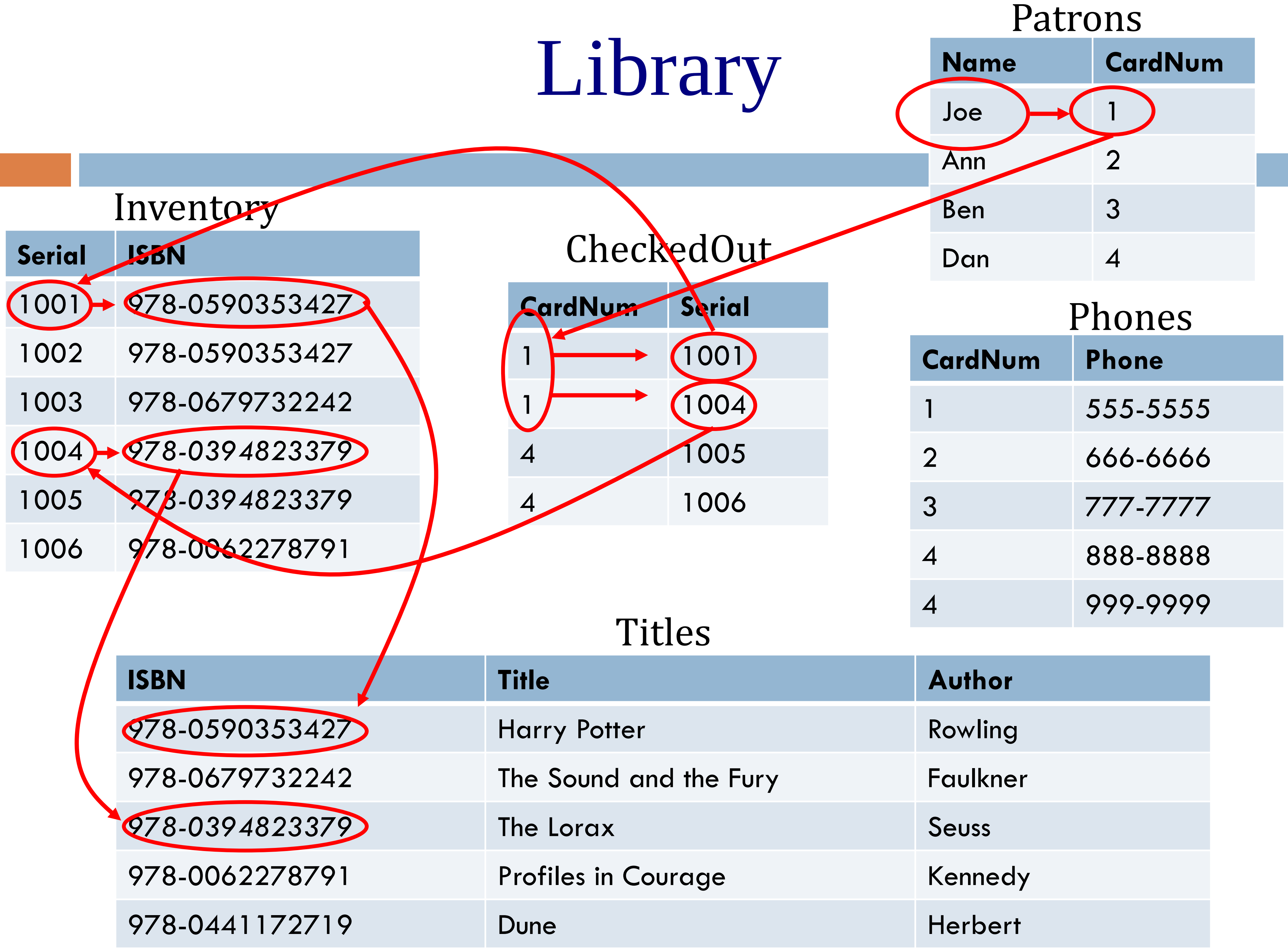
Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library



Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

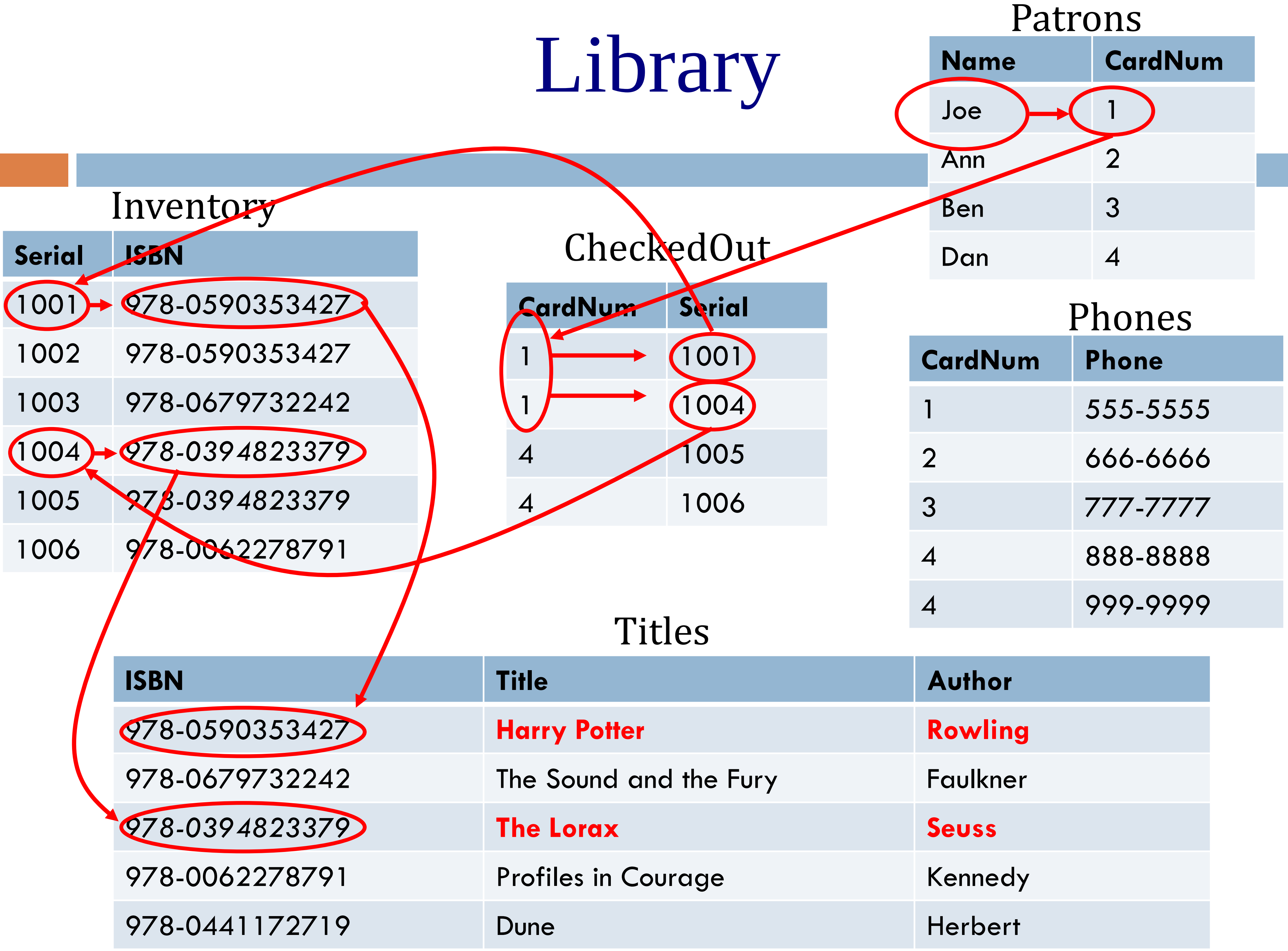
Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Library



Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

Phones

CardNum	Phone
1	555-5555
2	666-6666
3	777-7777
4	888-8888
4	999-9999

Titles

ISBN	Title	Author
978-0590353427	**Harry Potter**	**Rowling**
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	**The Lorax**	**Seuss**
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert

Don’t Worry – It’s Fast

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379
1005	978-0394823379
1006	978-0062278791

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005
4	1006

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Basic Design Goals

- Entries should be atoms (not complex)
  - Don’t store lists/arrays

Basic Design Goals

- Entries should be atoms (not complex)
  - Don’t store lists/arrays
  - Build compound information by referencing other tables

Basic Design Goals

- Entries should be atoms (not complex)
  - Don’t store lists/arrays
  - Build compound information by referencing other tables
  - Enables powerful reasoning about data and relationships, cleaner design

Basic Design Goals

•Entries should be atoms (not complex)

- Don’t store lists/arrays
- Build compound information by referencing other tables
- Enables powerful reasoning about data and relationships, cleaner design
- Enable DBMS to optimize

Basic Design Goals

- Bad news: SQL will let you violate good design rules

- Thus, we design the tables first without even thinking about SQL

Definitions

- **Attribute**: a name and a type (column heading)

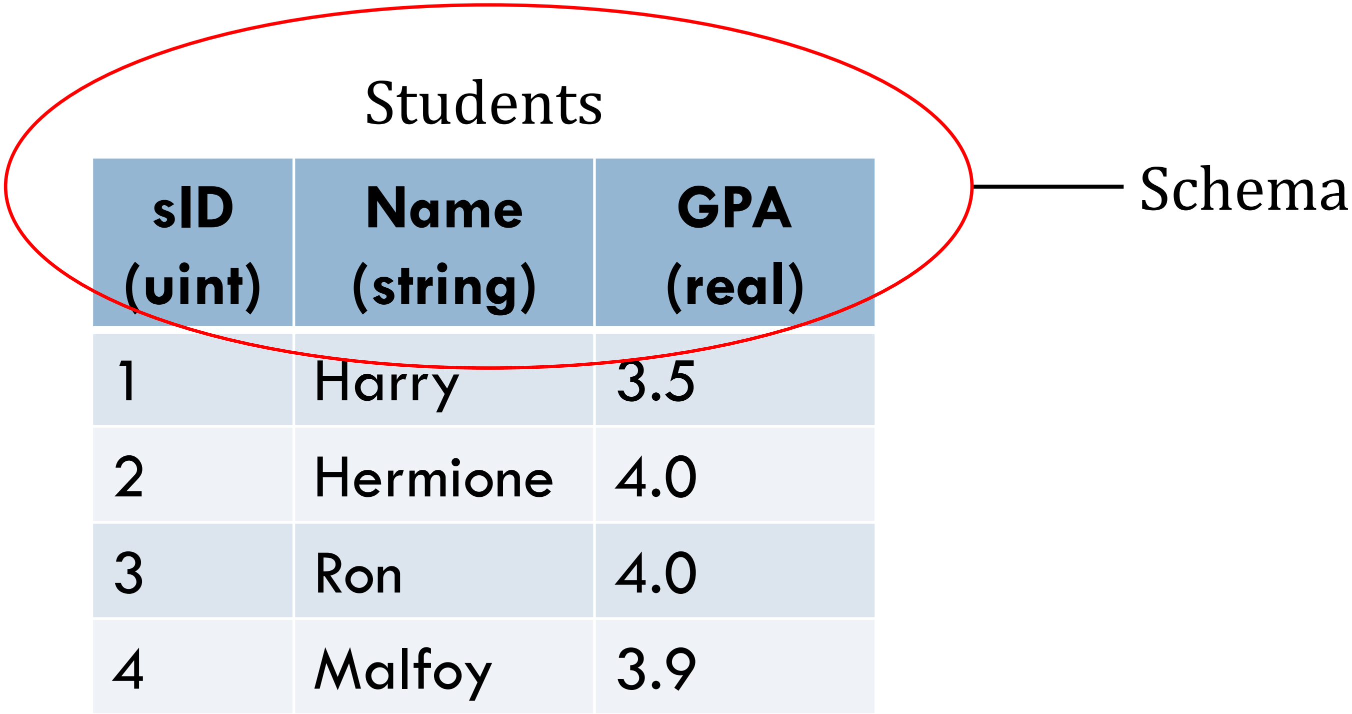
Students

Attribute

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Definitions

- **Schema**: Table name + a set of attributes
  - Specifies the structure/rules of a table

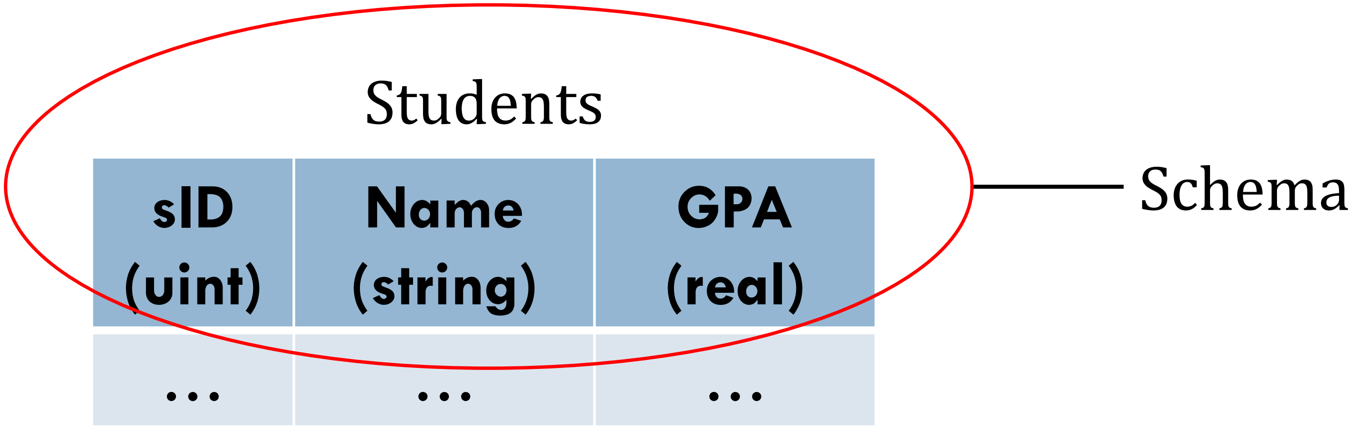


Students

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Definitions

- **Schema**: Table name + a set of attributes
  - Specifies the structure/rules of a table



Students

sID (uint)	Name (string)	GPA (real)
…	…	…

- A schema does *not* specify any values

Definitions

- **Instance**: the values in a table
  - A set of *tuples*

Students

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Instance

Definitions

- **Instance**: the values in a table
    - A set of tuples

- **Tuple**: one row

Students

sID	Name	GPA
(uint)	(string)	(real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Tuple

Definitions

- **Relation**: a.k.a “table”
  - Schema + instance

Students

sID	Name	GPA
(uint)	(string)	(real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Definitions

- **Relation**: a.k.a "table"
  - A schema + instance

Relation1

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Relation2

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0

Definitions

- **Instance**: the values in a table
  - A **set** of tuples

- **Tuple**: one row

sID	Name	GPA
(uint)	(string)	(real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Definitions

- **Instance**: the values in a table
  - A **set** of tuples – every row is unique!

- **Tuple**: one row

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Definitions

- But individual values do not need to be unique

- …then how do we guarantee each row is unique?

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Definitions

- But individual values do not need to be unique

- …then how do we guarantee each row is unique?
  - Some attribute (or set of attributes) must be unique

key

sID	Name	GPA
(uint)	(string)	(real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Keys

- Keys uniquely identify each tuple
  - Critical for the DBMS' underlying operations

key

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Keys

- Keys uniquely identify each tuple
  - Critical for the DBMS’ underlying operations

key

sID (uint)	Name (string)	GPA (real)
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

key

Make	Model	…
Toyota	Camry	
Toyota	F1	
Subaru	Outback	
Subaru	F1	

Keys

- As a DB designer, you will define keys for each table

Keys

- As a DB designer, you will define keys for each table

- …but we need formal definitions before SQL