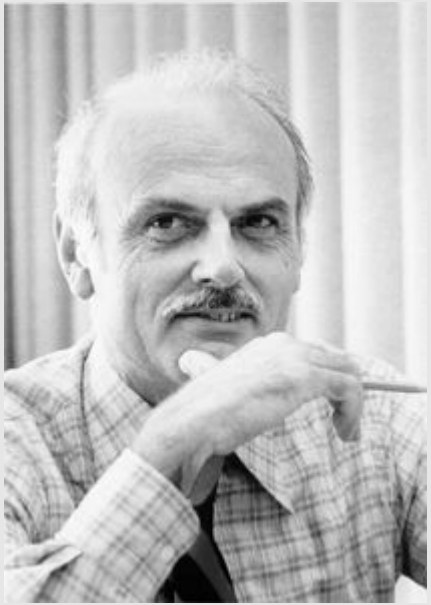
Advanced Python3: Object-oriented programming, databases and visualisation

Alexandra Diem
Simula Research Laboratory

L02: Relational Databases

History

Problem: It was impossible to work with databases without detailed knowledge about its low-level physical implementation. Codd's revolutionary idea: Provide a model of data and a language for manipulating it that are completely independent of the physical representation.



E. F. Codd (1970) A Relational Model of Data for Large Shared Data Banks, *Information Retrieval* **13**(6): 377-387.

Oracle was the first RDBMS to be released in 1979, these days there are many: MySQL, SQLite, PostgreSQL, IBM Db2, ...

Relation?

Assume S_1 and S_2 are sets in the mathematical sense. The Cartesian product $S_1 \times S_2$ is the set

$$S_1 \times S_2 = \{(s_1, s_2) \mid s_1 \in S_1, s_2 \in S_2\}$$

A *binary relation* over $S_1 \times S_2$ is any set r

$$r \subseteq S_1 \times S_2$$

This can, of course, be easily extended to n sets, such that a relation r is

$$r \subseteq S_1 \times S_2 \times \ldots \times S_n = \{(s_1, s_2, \ldots, s_n) \mid s_i \in S_i\}$$

Relation?

Some rules:

- All rows should be distinct
- Values in columns must be *atomic*, e.g. integers, floats, strings, booleans, dates, but not lists, arrays
- Every relation has a **key** (usually an ID number)

Relational Schema

Assuming that an n-tuple $t \in S_1 \times \ldots \times S_n$ is called an entry. Accessing the i-th component of t feels quite unintuitive and requires me to remember the order of domain S_i.

Solution: Associate an attribute name A_i with each S_i

$$A_1 : S_1, A_2 : S_2, \ldots A_n : S_n$$

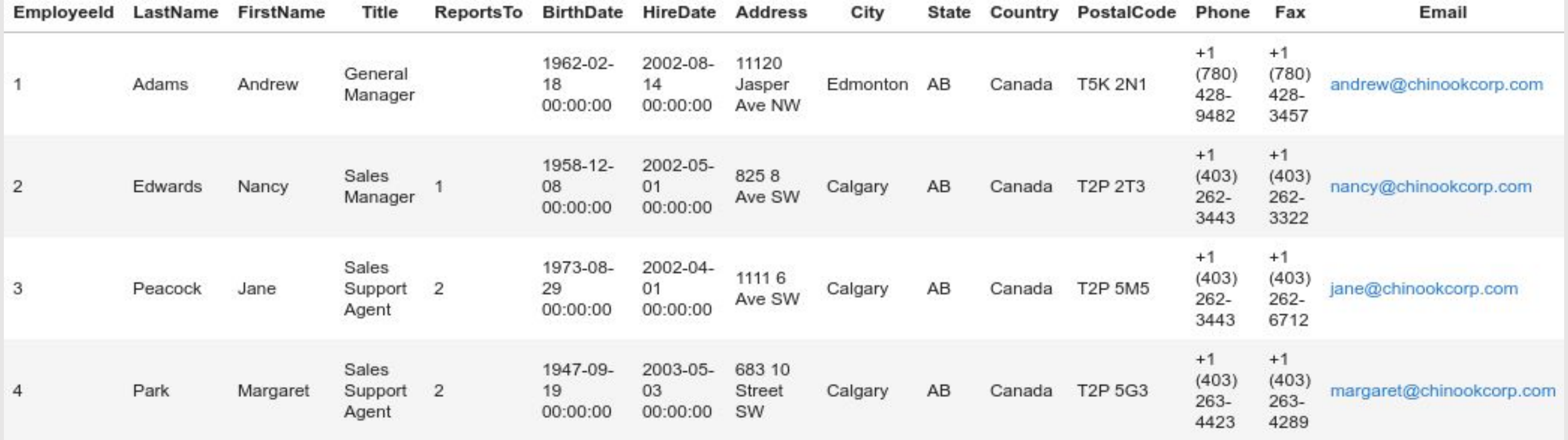
A relation R over this schema is

$$R \subseteq \{\{(A_1, s_1), (A_2, s_2), \ldots, (A_n, s_n)\} \mid s_i \in S_i\}$$

Relational Schema to Database

A **database** consists of a set of two-dimensional relations (tables).

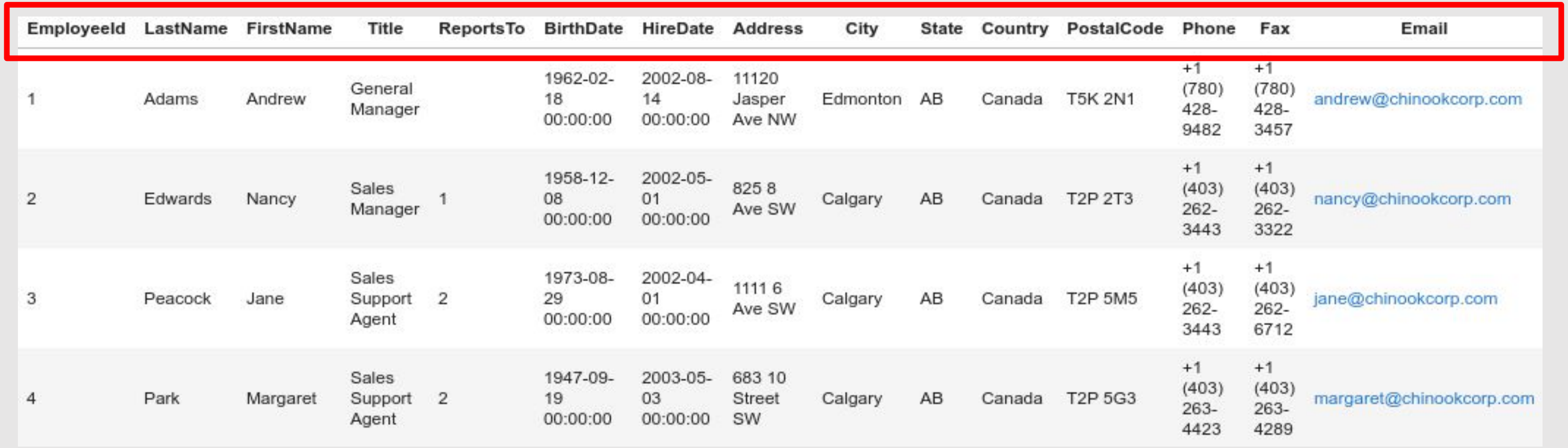
A **relation/table** is a set of tuples (rows) with identical structure (columns). Each column is named with a distinct **attribute**.



EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com

Relational Schema

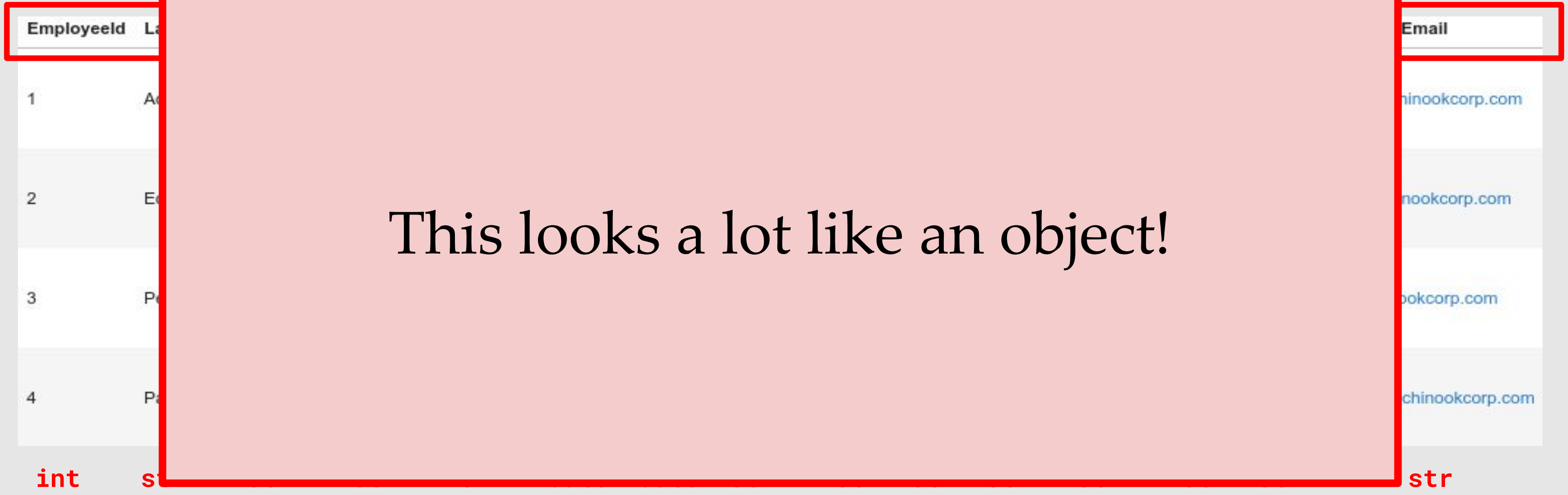
Employees(**EmployeeId**: int, **LastName**: str, **FirstName**: str, **Title**: str, **ReportsTo**: int, **BirthDate**: datetime, **HireDate**: datetime, ...)



EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com
int	str	str	str	int	date	date	str	str	str	str	str	str	str	str

Relational Schema

Employees(**EmployeeId**: int, **LastName**: str, **FirstName**: str, **Title**: str, **ReportsTo**: int, **BirthDate**: datetime, **HireDate**: datetime, ...)

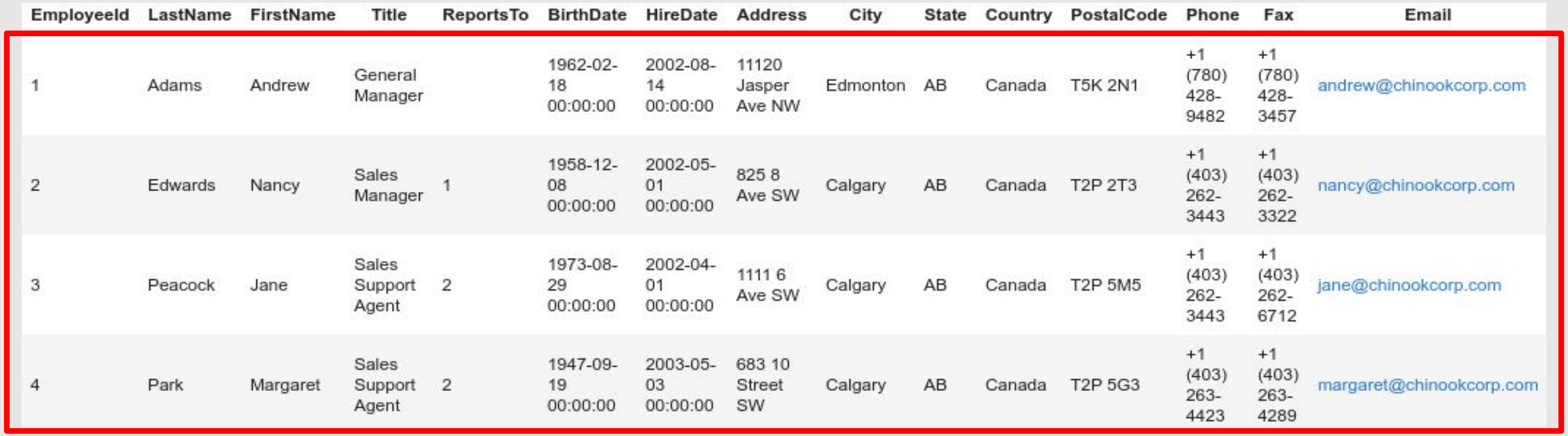


EmployeeId	La	Email
1	Ac	iinookcorp.com
2	E	nookcorp.com
3	P	okcorp.com
4	P	chinookcorp.com
int	st	str

This looks a lot like an object!

Relational Instance of a schema

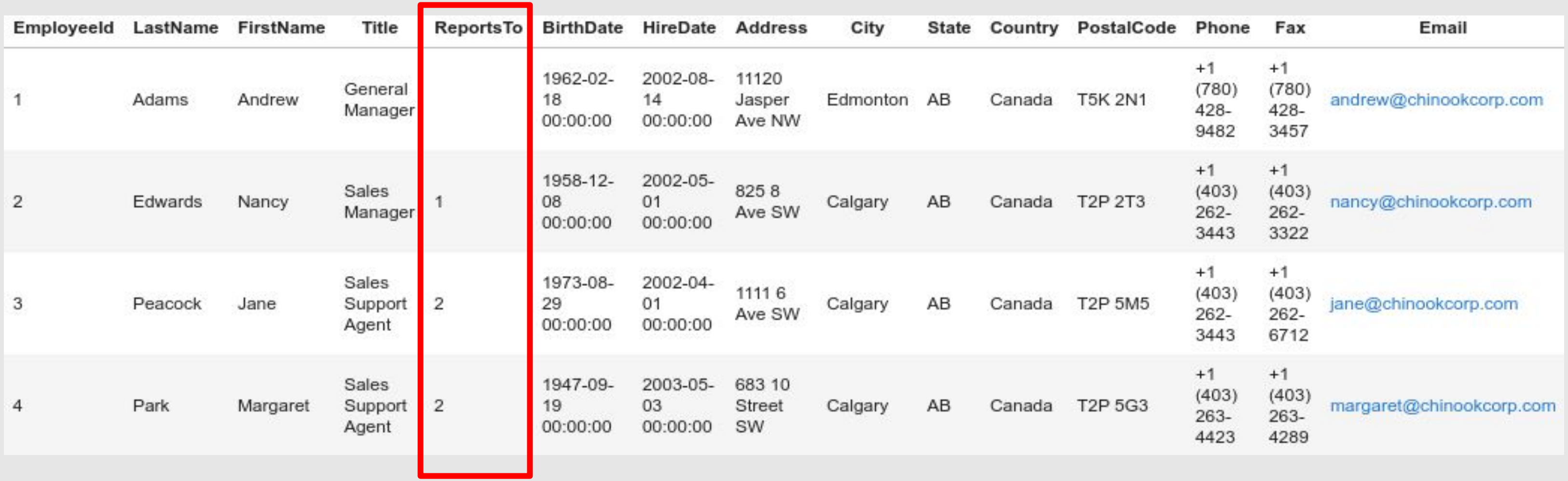
Employees = {{(EmployeeId, 1), (LastName, Adams), (FirstName, Andrew), ...}, {(EmployeeId, 2), (LastName, Edwards), (FirstName, Nancy), ...}, ...}



EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com

Relationships

Relationships between relational instances via **foreign keys**. These can also refer to the same relational instance.



EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com

Data Normalisation

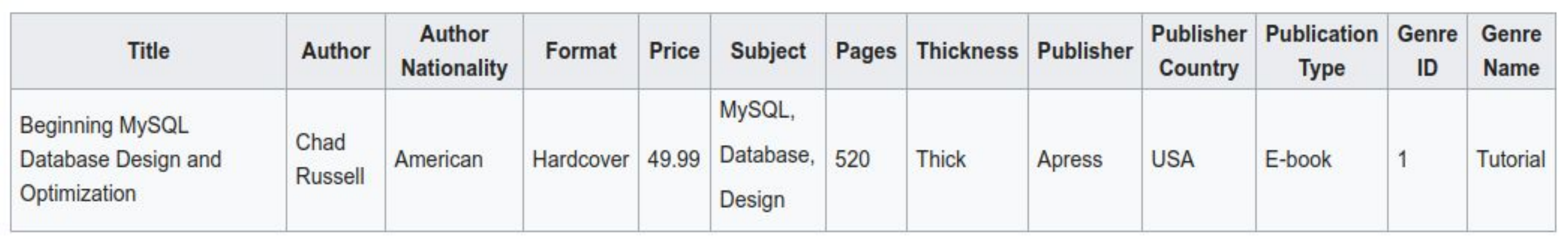
For the relational model to work the data to be stored has to follow certain rules

- All rows should be distinct (no duplicates) and have a **key**
- Values in columns must be *atomic*, e.g. integers, floats, strings, booleans, dates, but not lists, arrays
- No repeating columns
- Values only depend on the key
- No redundancies

Data normalisation is the process of structuring a relational database to follow these rules and is usually carried out as a series of **normal forms**.

Data Normalisation

Consider the following relational instance:



Title	Author	Author Nationality	Format	Price	Subject	Pages	Thickness	Publisher	Publisher Country	Publication Type	Genre ID	Genre Name
Beginning MySQL Database Design and Optimization	Chad Russell	American	Hardcover	49.99	MySQL, Database, Design	520	Thick	Apress	USA	E-book	1	Tutorial

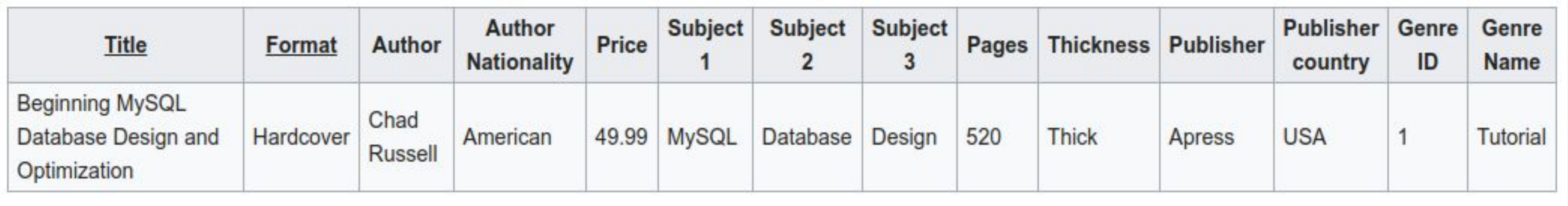
No duplicate tuples ✔

No repeating columns ✔

Atomic columns ✘

Data Normalisation

Satisfying the first normal form (1NF):

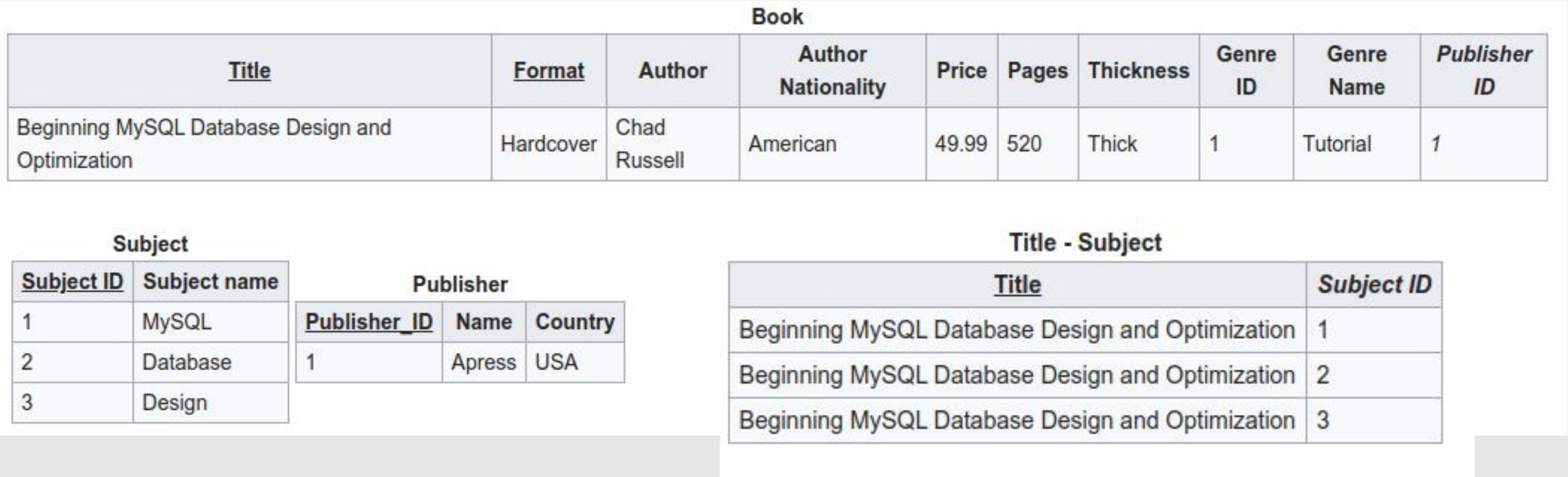


Title	Format	Author	Author Nationality	Price	Subject 1	Subject 2	Subject 3	Pages	Thickness	Publisher	Publisher country	Genre ID	Genre Name
Beginning MySQL Database Design and Optimization	Hardcover	Chad Russell	American	49.99	MySQL	Database	Design	520	Thick	Apress	USA	1	Tutorial

This is perfectly legal, but not particularly elegant.

Data Normalisation

Satisfying the first normal form (1NF):



Book

Title	Format	Author	Author Nationality	Price	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Hardcover	Chad Russell	American	49.99	520	Thick	1	Tutorial	*1*

Subject

Subject ID	Subject name
1	MySQL
2	Database
3	Design

Publisher

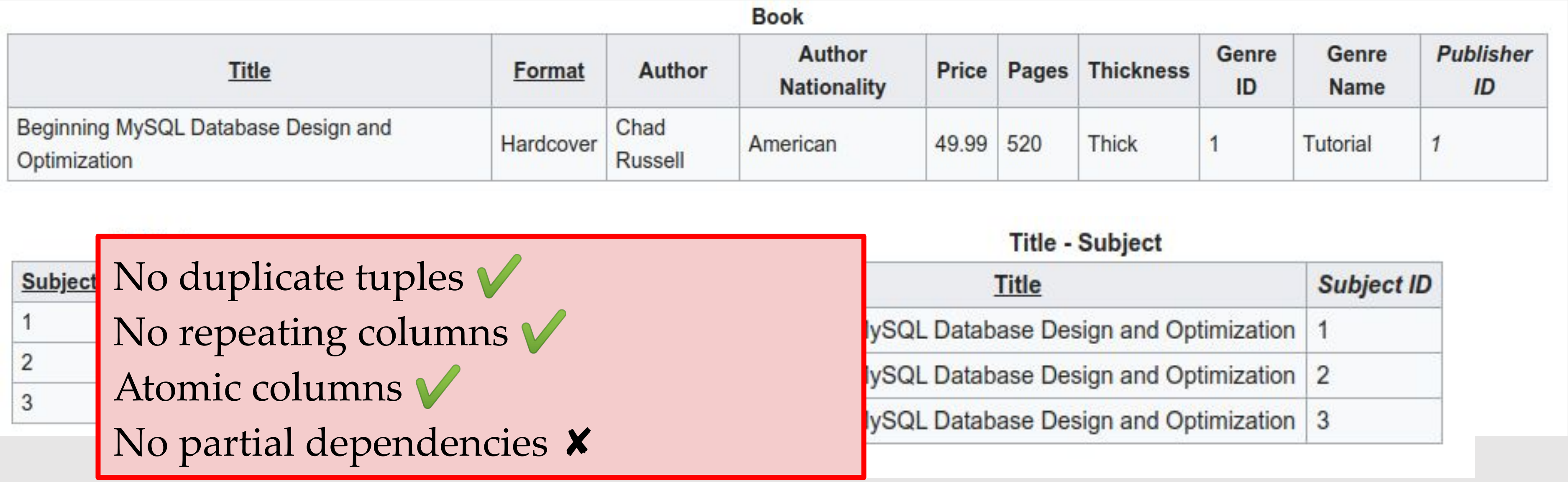
Publisher_ID	Name	Country
1	Apress	USA

Title - Subject

Title	*Subject ID*
Beginning MySQL Database Design and Optimization	1
Beginning MySQL Database Design and Optimization	2
Beginning MySQL Database Design and Optimization	3

Data Normalisation

Satisfying the second normal form (2NF):



Book

Title	Format	Author	Author Nationality	Price	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Hardcover	Chad Russell	American	49.99	520	Thick	1	Tutorial	*1*

Subject
1
2
3

Title - Subject

Title	*Subject ID*
ySQL Database Design and Optimization	1
ySQL Database Design and Optimization	2
ySQL Database Design and Optimization	3

No duplicate tuples ✓

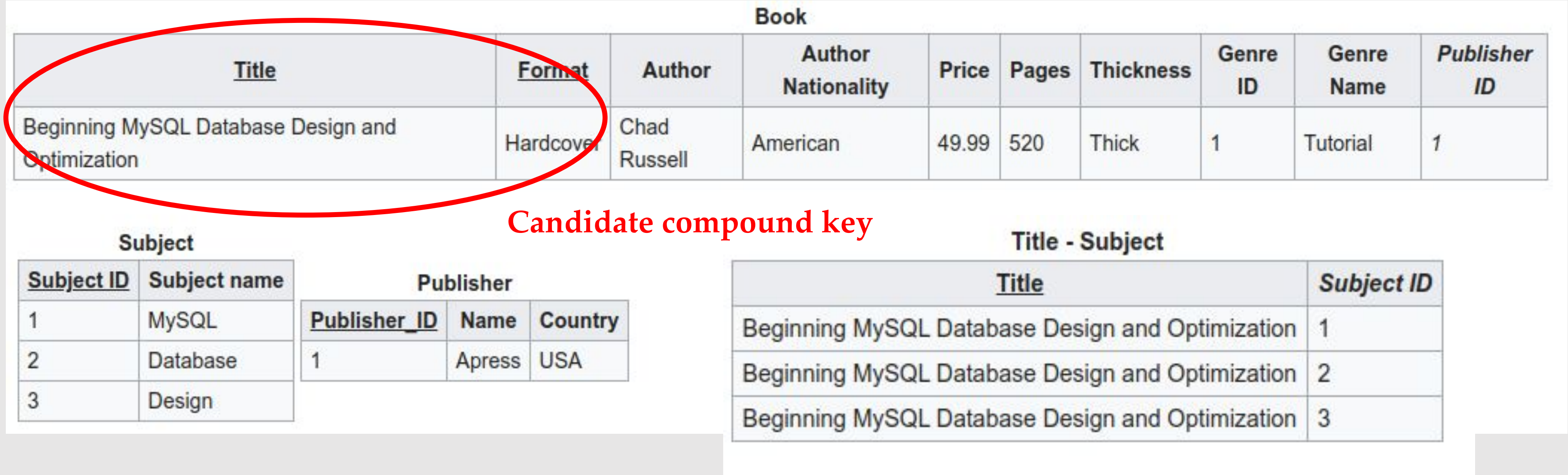
No repeating columns ✓

Atomic columns ✓

No partial dependencies ✘

Data Normalisation

Satisfying the second normal form (2NF):



Book

Title	Format	Author	Author Nationality	Price	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Hardcover	Chad Russell	American	49.99	520	Thick	1	Tutorial	*1*

Candidate compound key

Subject

Subject ID	Subject name
1	MySQL
2	Database
3	Design

Publisher

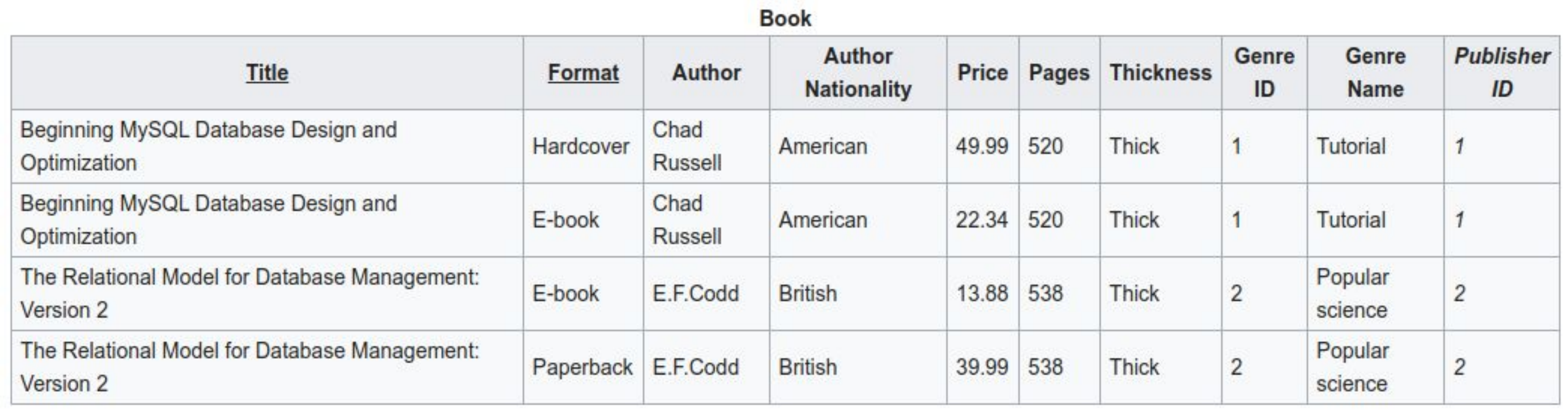
Publisher_ID	Name	Country
1	Apress	USA

Title - Subject

Title	*Subject ID*
Beginning MySQL Database Design and Optimization	1
Beginning MySQL Database Design and Optimization	2
Beginning MySQL Database Design and Optimization	3

Data Normalisation

Satisfying the second normal form (2NF): Consider the table fragment



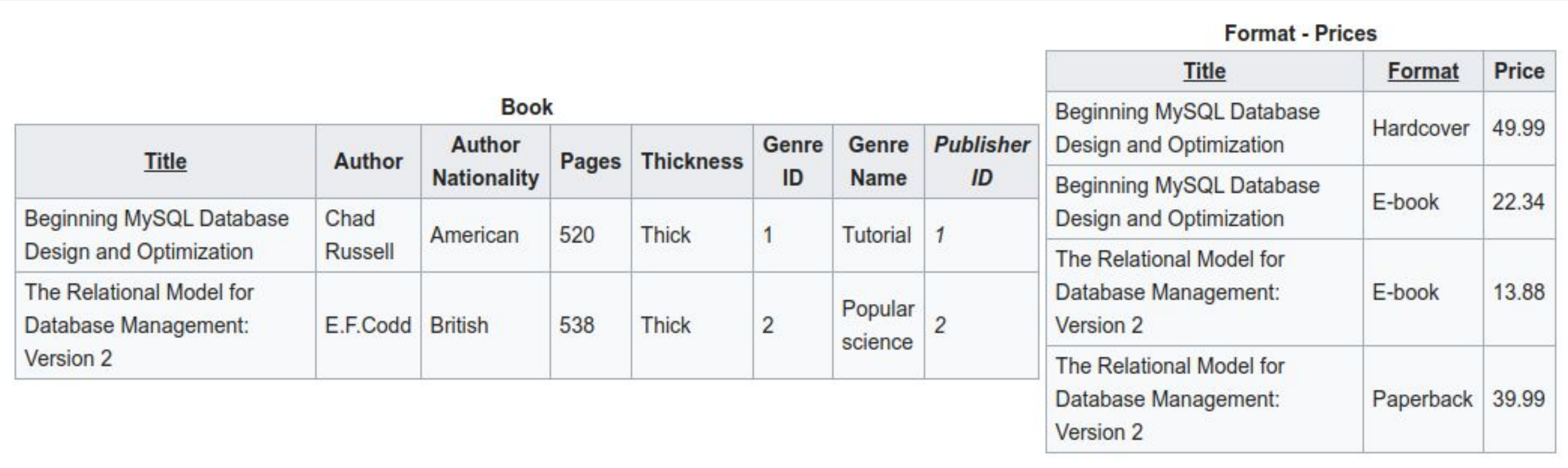
Book

Title	Format	Author	Author Nationality	Price	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Hardcover	Chad Russell	American	49.99	520	Thick	1	Tutorial	*1*
Beginning MySQL Database Design and Optimization	E-book	Chad Russell	American	22.34	520	Thick	1	Tutorial	*1*
The Relational Model for Database Management: Version 2	E-book	E.F.Codd	British	13.88	538	Thick	2	Popular science	2
The Relational Model for Database Management: Version 2	Paperback	E.F.Codd	British	39.99	538	Thick	2	Popular science	2

{Title, Format} looks like a reasonable candidate key, but only “Price” depends on Format.

Data Normalisation

Satisfying the second normal form (2NF):



Book

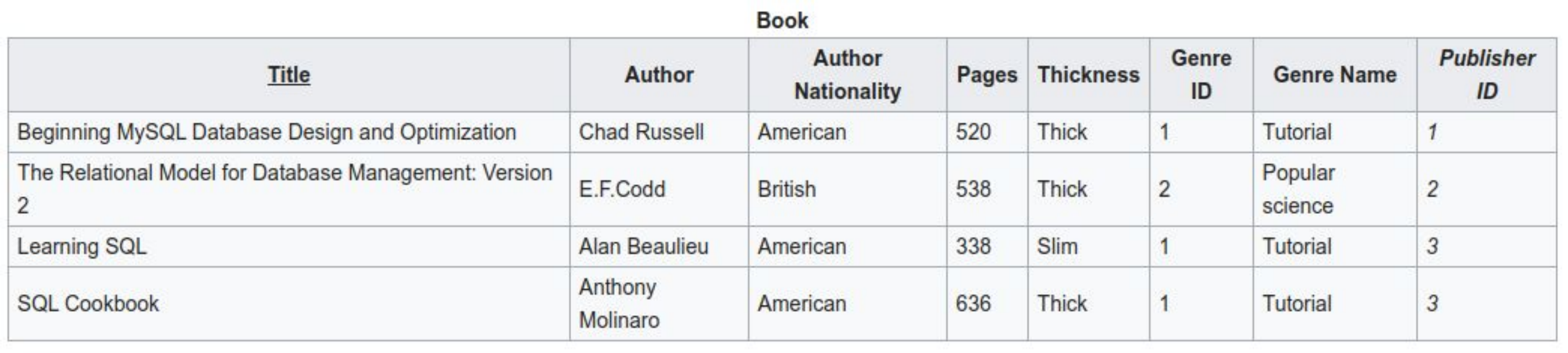
Title	Author	Author Nationality	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Chad Russell	American	520	Thick	1	Tutorial	*1*
The Relational Model for Database Management: Version 2	E.F.Codd	British	538	Thick	2	Popular science	2

Format - Prices

Title	Format	Price
Beginning MySQL Database Design and Optimization	Hardcover	49.99
Beginning MySQL Database Design and Optimization	E-book	22.34
The Relational Model for Database Management: Version 2	E-book	13.88
The Relational Model for Database Management: Version 2	Paperback	39.99

Data Normalisation

Satisfying the second normal form (3NF): Consider the table fragment

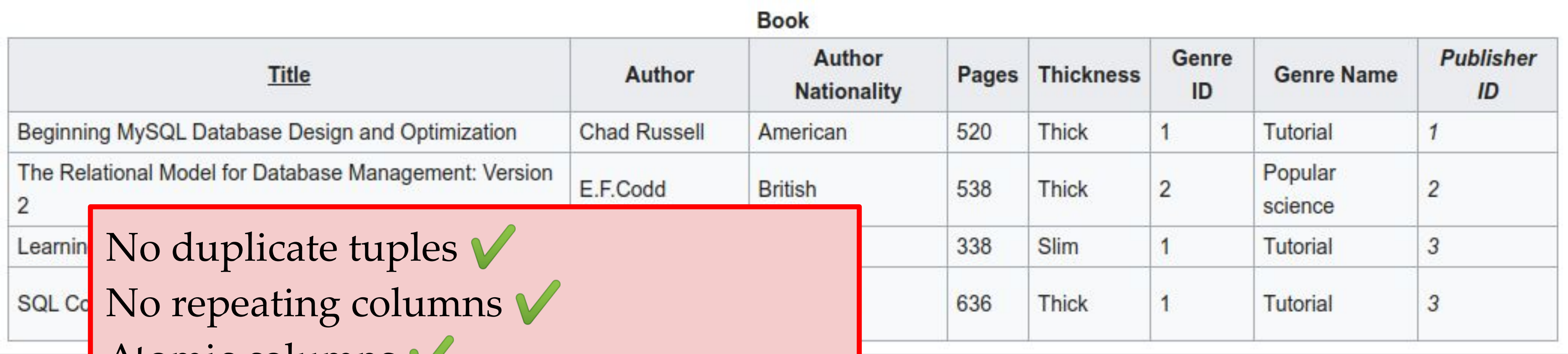


Book

<u>Title</u>	Author	Author Nationality	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Chad Russell	American	520	Thick	1	Tutorial	*1*
The Relational Model for Database Management: Version 2	E.F.Codd	British	538	Thick	2	Popular science	2
Learning SQL	Alan Beaulieu	American	338	Slim	1	Tutorial	*3*
SQL Cookbook	Anthony Molinaro	American	636	Thick	1	Tutorial	*3*

Data Normalisation

Satisfying the second normal form (3NF): Consider the table fragment



Book

Title	Author	Author Nationality	Pages	Thickness	Genre ID	Genre Name	*Publisher ID*
Beginning MySQL Database Design and Optimization	Chad Russell	American	520	Thick	1	Tutorial	*1*
The Relational Model for Database Management: Version 2	E.F.Codd	British	538	Thick	2	Popular science	*2*
Learnin			338	Slim	1	Tutorial	*3*
SQL Co			636	Thick	1	Tutorial	*3*

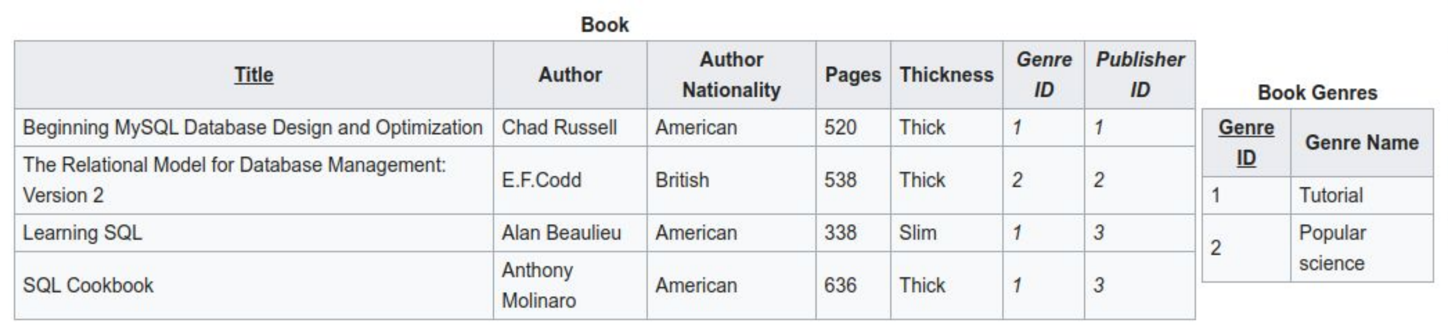
No duplicate tuples ✓
No repeating columns ✓
Atomic columns ✓
No partial dependencies ✓
No transitive dependencies ✘

simula consulting

simula

Data Normalisation

Satisfying the second normal form (3NF):

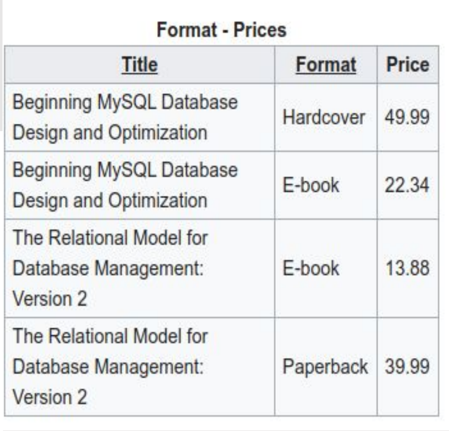


Book

Title	Author	Author Nationality	Pages	Thickness	*Genre ID*	*Publisher ID*
Beginning MySQL Database Design and Optimization	Chad Russell	American	520	Thick	*1*	*1*
The Relational Model for Database Management: Version 2	E.F.Codd	British	538	Thick	2	2
Learning SQL	Alan Beaulieu	American	338	Slim	*1*	*3*
SQL Cookbook	Anthony Molinaro	American	636	Thick	*1*	*3*

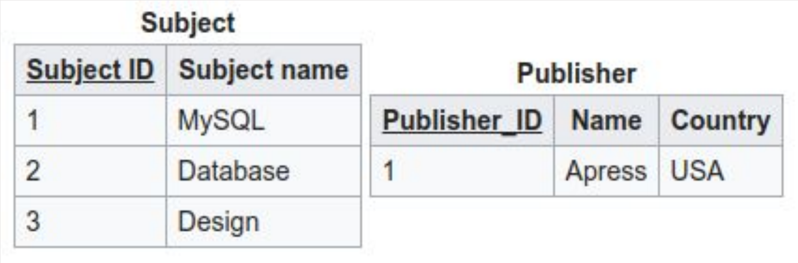
Book Genres

Genre ID	Genre Name
1	Tutorial
2	Popular science



Format - Prices

Title	Format	Price
Beginning MySQL Database Design and Optimization	Hardcover	49.99
Beginning MySQL Database Design and Optimization	E-book	22.34
The Relational Model for Database Management: Version 2	E-book	13.88
The Relational Model for Database Management: Version 2	Paperback	39.99

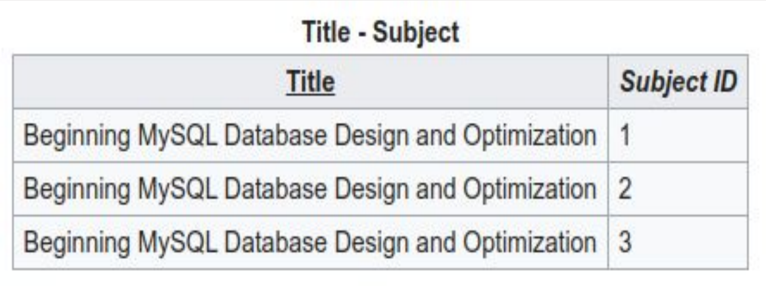


Subject

Subject ID	Subject name
1	MySQL
2	Database
3	Design

Publisher

Publisher_ID	Name	Country
1	Apress	USA



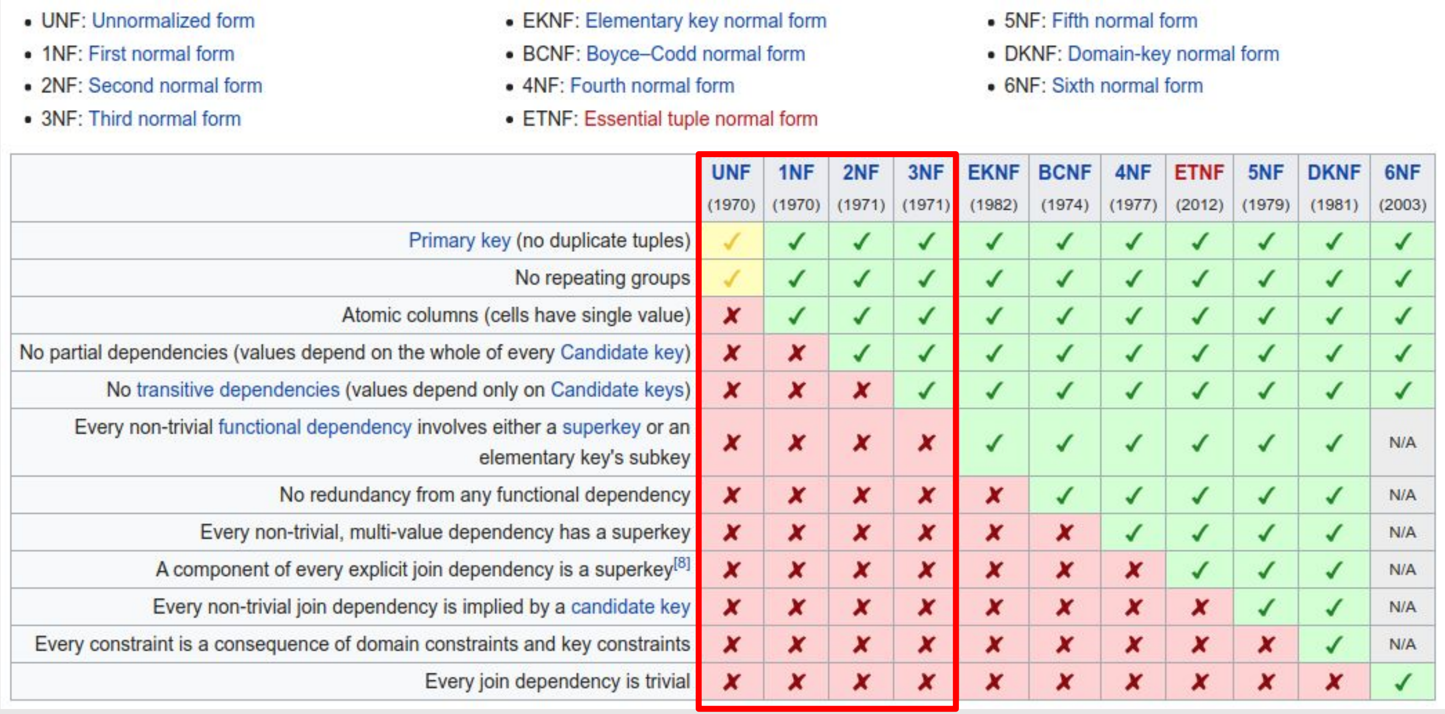
Title - Subject

Title	*Subject ID*
Beginning MySQL Database Design and Optimization	1
Beginning MySQL Database Design and Optimization	2
Beginning MySQL Database Design and Optimization	3

simula consulting

simula

Data Normalisation



- UNF: Unnormalized form
- 1NF: First normal form
- 2NF: Second normal form
- 3NF: Third normal form
- EKNF: Elementary key normal form
- BCNF: Boyce–Codd normal form
- 4NF: Fourth normal form
- ETNF: Essential tuple normal form
- 5NF: Fifth normal form
- DKNF: Domain-key normal form
- 6NF: Sixth normal form

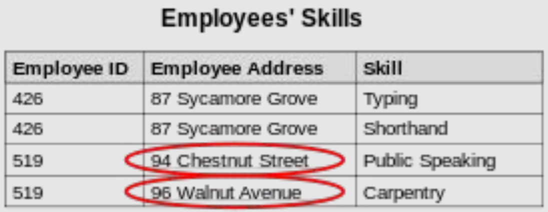
	UNF (1970)	1NF (1970)	2NF (1971)	3NF (1971)	EKNF (1982)	BCNF (1974)	4NF (1977)	ETNF (2012)	5NF (1979)	DKNF (1981)	6NF (2003)
Primary key (no duplicate tuples)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
No repeating groups	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Atomic columns (cells have single value)	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
No partial dependencies (values depend on the whole of every Candidate key)	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓
No transitive dependencies (values depend only on Candidate keys)	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓
Every non-trivial functional dependency involves either a superkey or an elementary key's subkey	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	N/A
No redundancy from any functional dependency	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓	N/A
Every non-trivial, multi-value dependency has a superkey	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	N/A
A component of every explicit join dependency is a superkey[8]	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	N/A
Every non-trivial join dependency is implied by a candidate key	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	N/A
Every constraint is a consequence of domain constraints and key constraints	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	N/A
Every join dependency is trivial	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

"good enough"

Data Normalisation

Data normalisation additionally prevents undesired side-effects induced by update, insertion, or deletion operations:

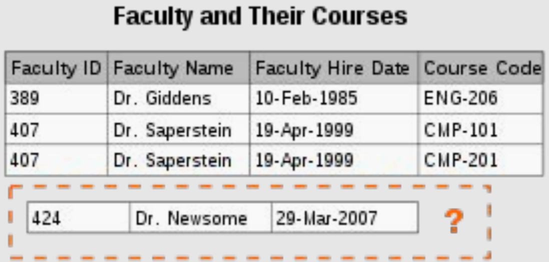
Update anomaly



Employees' Skills

Employee ID	Employee Address	Skill
426	87 Sycamore Grove	Typing
426	87 Sycamore Grove	Shorthand
519	94 Chestnut Street	Public Speaking
519	96 Walnut Avenue	Carpentry

Insertion anomaly

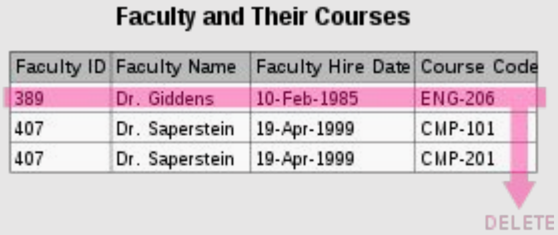


Faculty and Their Courses

Faculty ID	Faculty Name	Faculty Hire Date	Course Code
389	Dr. Giddens	10-Feb-1985	ENG-206
407	Dr. Saperstein	19-Apr-1999	CMP-101
407	Dr. Saperstein	19-Apr-1999	CMP-201

424	Dr. Newsome	29-Mar-2007	?

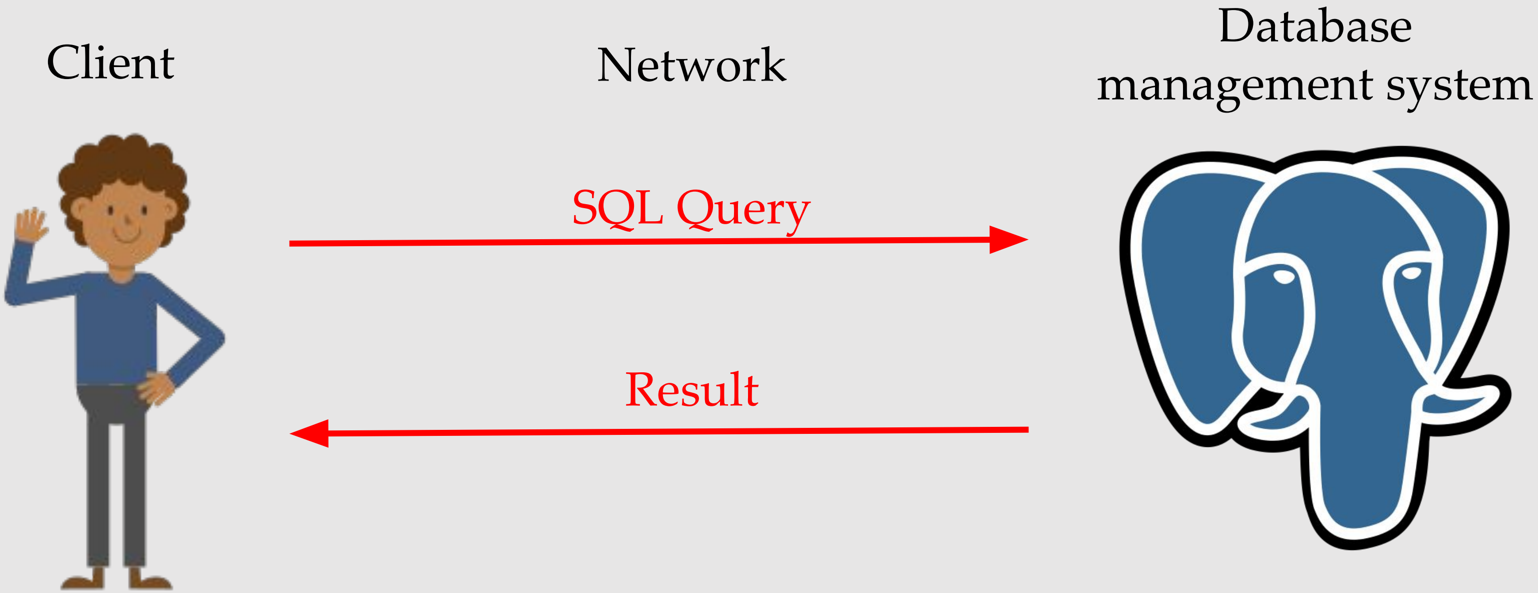
Deletion anomaly



Faculty and Their Courses

Faculty ID	Faculty Name	Faculty Hire Date	Course Code
389	Dr. Giddens	10-Feb-1985	ENG-206
407	Dr. Saperstein	19-Apr-1999	CMP-101
407	Dr. Saperstein	19-Apr-1999	CMP-201

Interacting with databases



Client

Network

Database management system

SQL Query

Result

SQL

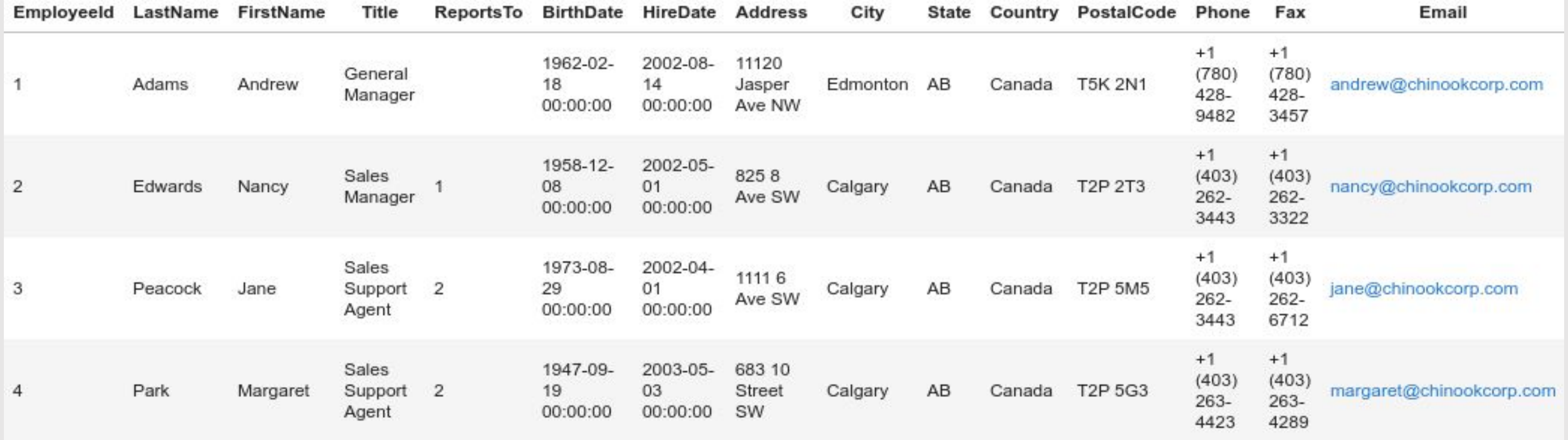
“S-Q-L” or “Sequel” is the language for management, manipulation, and querying of relational databases. It incorporates both a

- **Data Manipulation Language:** notation for expressing management, manipulation, and queries.
- **Data Definition Language:** notation for expressing a database or table structure

SQL: Creating a database

```sql
CREATE TABLE employees
(
  EmployeeId INT,
  LastName VARCHAR(20),
  FirstName VARCHAR(20),
  Title VARCHAR(30),
  ReportsTo INT,
  BirthDate DATE,
  HireDate DATE,
  Address VARCHAR(70),
  City VARCHAR(40),
  ...
);
```

SQL: Querying

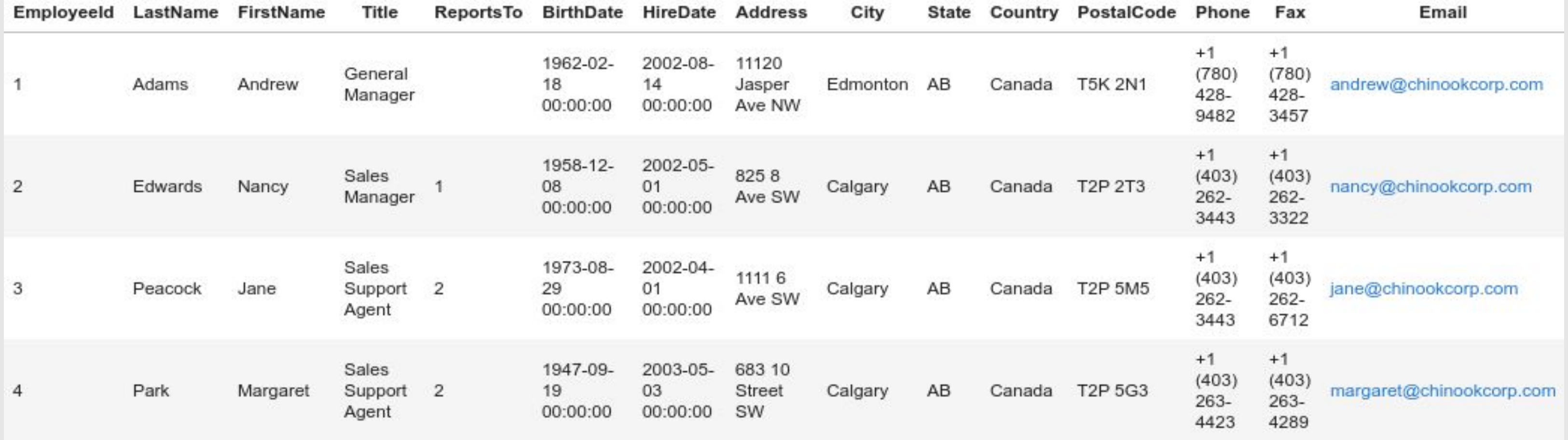


EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com

```
SELECT *
FROM employees;
```

```
EmployeeId|LastName|FirstName|Title|ReportsTo|BirthDate|Hire
Date|Address|City|State|Country|PostalCode|Phone|Fax|Email
1|Adams|Andrew|General Manager||1962-02-18
00:00:00|2002-08-14 00:00:00|11120 Jasper Ave
NW|Edmonton|AB|Canada|T5K 2N1|+1 (780) 428-9482|+1 (780)
428-3457|andrew@chinookcorp.com
```

SQL: Querying

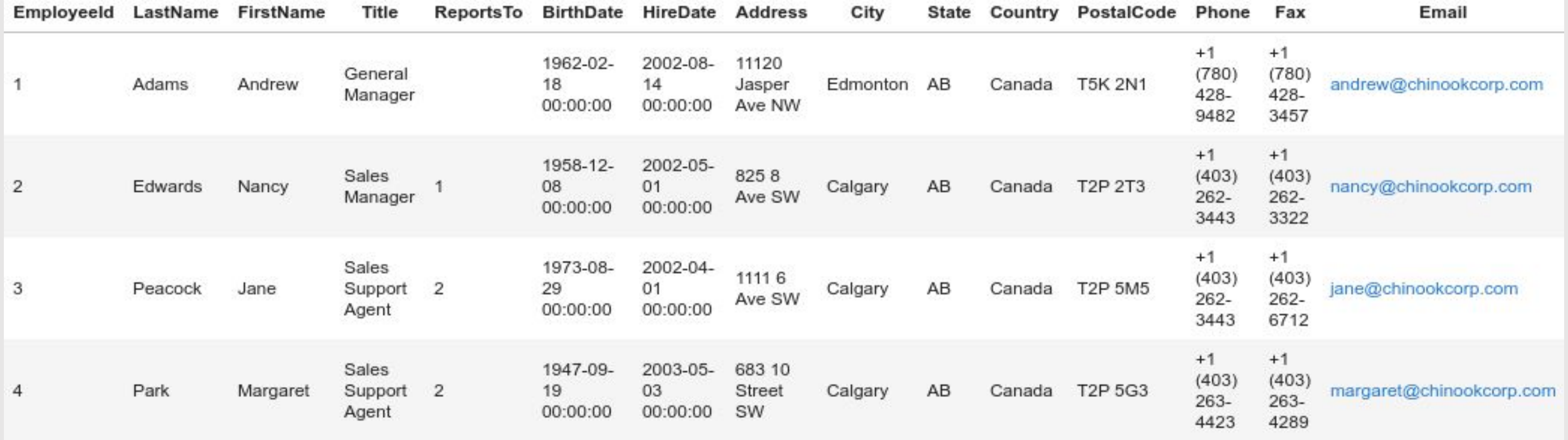


EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com

```
SELECT FirstName, LastName
FROM employees;
```

```
FirstName|LastName
Andrew|Adams
Nancy|Edwards
Jane|Peacock
Margaret|Park
```

SQL: Querying



EmployeeId	LastName	FirstName	Title	ReportsTo	BirthDate	HireDate	Address	City	State	Country	PostalCode	Phone	Fax	Email
1	Adams	Andrew	General Manager		1962-02-18 00:00:00	2002-08-14 00:00:00	11120 Jasper Ave NW	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1 (780) 428-3457	andrew@chinookcorp.com
2	Edwards	Nancy	Sales Manager	1	1958-12-08 00:00:00	2002-05-01 00:00:00	825 8 Ave SW	Calgary	AB	Canada	T2P 2T3	+1 (403) 262-3443	+1 (403) 262-3322	nancy@chinookcorp.com
3	Peacock	Jane	Sales Support Agent	2	1973-08-29 00:00:00	2002-04-01 00:00:00	1111 6 Ave SW	Calgary	AB	Canada	T2P 5M5	+1 (403) 262-3443	+1 (403) 262-6712	jane@chinookcorp.com
4	Park	Margaret	Sales Support Agent	2	1947-09-19 00:00:00	2003-05-03 00:00:00	683 10 Street SW	Calgary	AB	Canada	T2P 5G3	+1 (403) 263-4423	+1 (403) 263-4289	margaret@chinookcorp.com

```
SELECT FirstName, LastName
FROM employees
WHERE City==‘Calgary’;
```

```
FirstName|LastName
Nancy|Edwards
Jane|Peacock
Margaret|Park
```

PostgreSQL

- a.k.a "Postgres"
- in their own words (https://www.postgresql.org/about/): "PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads."

Pro	Con
- Data integrity through advanced currency control - Open source, community-driven - Integration with many other tools and programming languages - Efficiently answers complex queries by leveraging multiple CPUs - User extensible	- Comparatively slow - Overkill for simple DBs - Replication is not its most mature feature