Databases

Content

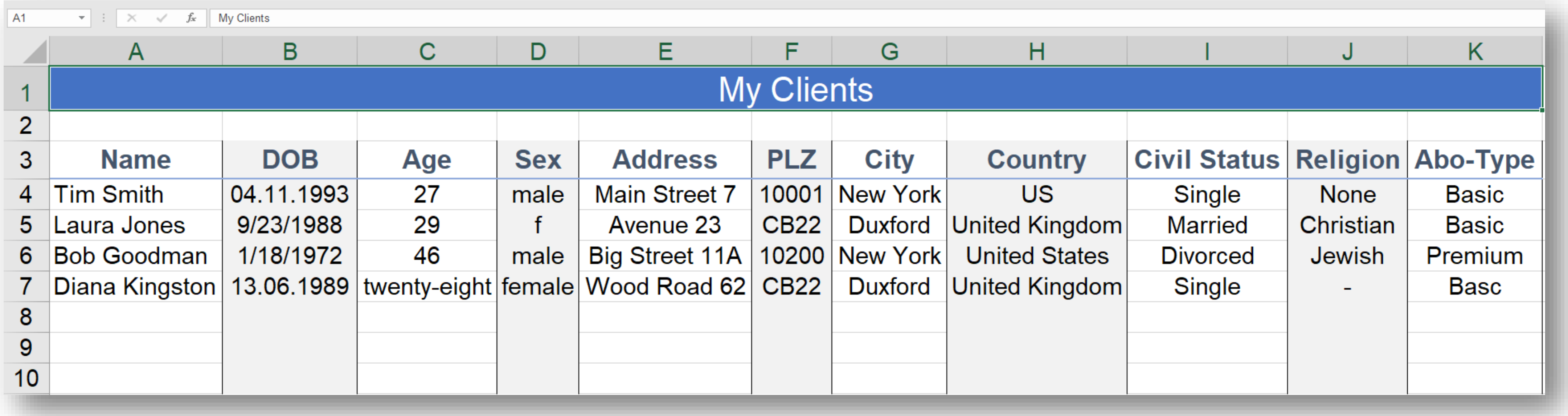
- Why?
- Qualities
- Types

Content

- Why?
- Qualities
- Types

Why?

A little hands on…

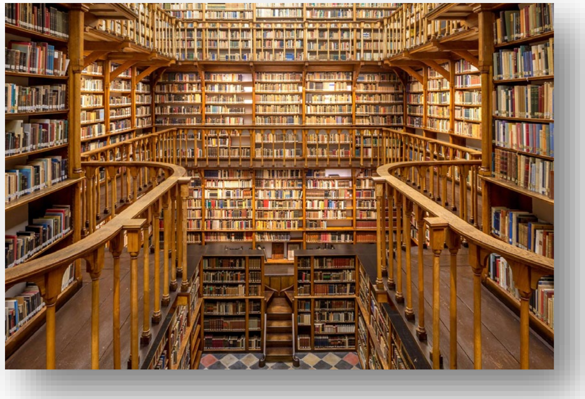


	A	B	C	D	E	F	G	H	I	J	K
1	My Clients										
2											
3	Name	DOB	Age	Sex	Address	PLZ	City	Country	Civil Status	Religion	Abo-Type
4	Tim Smith	04.11.1993	27	male	Main Street 7	10001	New York	US	Single	None	Basic
5	Laura Jones	9/23/1988	29	f	Avenue 23	CB22	Duxford	United Kingdom	Married	Christian	Basic
6	Bob Goodman	1/18/1972	46	male	Big Street 11A	10200	New York	United States	Divorced	Jewish	Premium
7	Diana Kingston	13.06.1989	twenty-eight	female	Wood Road 62	CB22	Duxford	United Kingdom	Single	-	Basc
8											
9											
10											

Why?

Spreadsheet-driven

Library with 1000 books and 50 customers



➞ Go digital

- Book-Management
- Client-Data
- Lending-Information
- Dunning, Debt-Collection
- …

Clear overview?

Consistent?

Multi-user?

Queries?

Spreadsheet Application

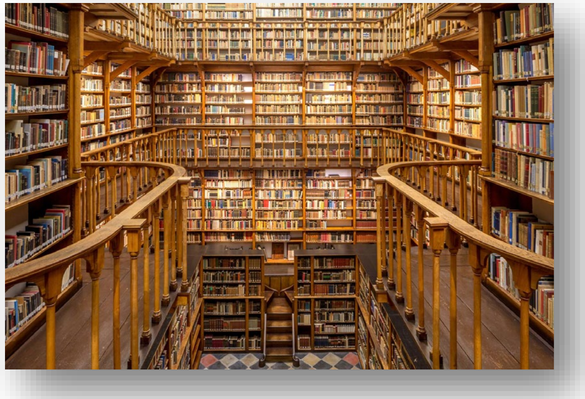
Performance?

What tables?

Why?

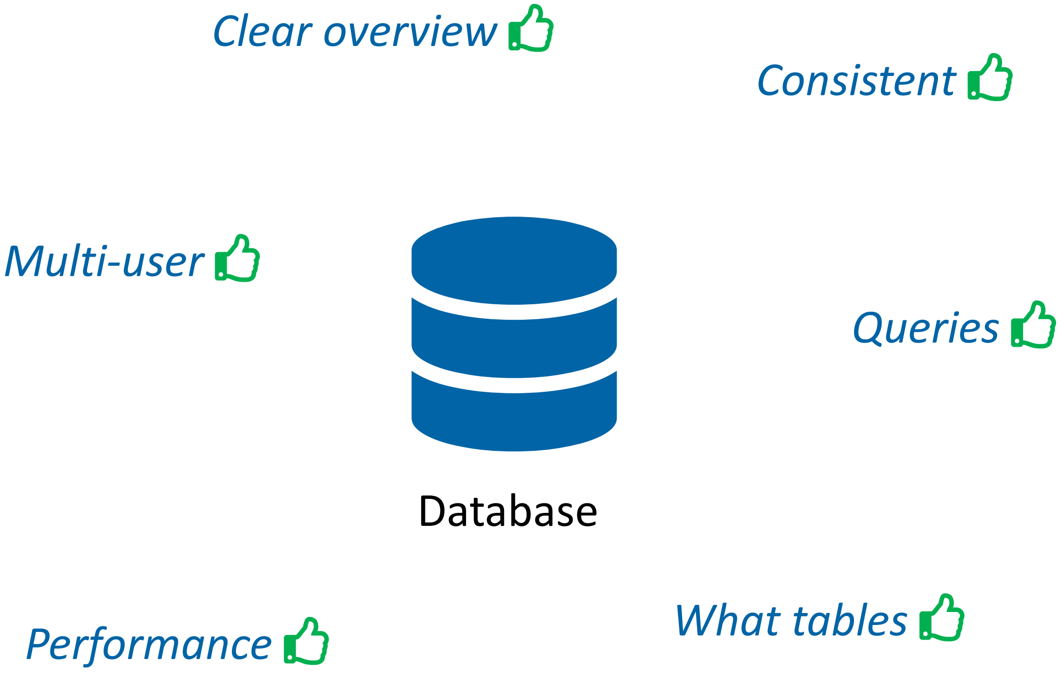
Database-driven

Library with 1000 books and 50 customers



→ Go digital

- Book-Management
- Client-Data
- Lending-Information
- Dunning, Debt-Collection
- …



Clear overview 👍

Consistent 👍

Multi-user 👍

Database

Queries 👍

Performance 👍

What tables 👍

Why? | Pros and Cons

Advantages

- Reduced data redundancy
- Reduced updating errors and increased consistency
- Greater data integrity and independence from applications programs
- Improved data access to users through use of host and query languages
- Improved data security
- Reduced data entry, storage, and retrieval costs
- Facilitated development of new applications program

Disadvantages

- Database systems are complex, difficult, and time-consuming to design
- Substantial hardware and software start-up costs
- Damage to database affects virtually all applications programs
- Extensive conversion costs in moving form a file-based system to a database system
- Initial training required for all programmers and users

Content

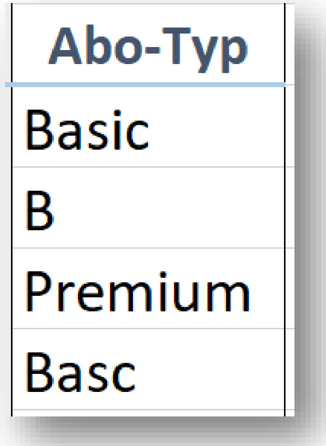
- Why?
- **Qualities**
- Types

Qualities

- Consistency
- Performance
- Independence
- Abstraction

Qualities | Consistency

- Any given database transaction must change affected data only in allowed ways.
- Any data written to the database must be valid according to all defined rules, including constraints, cascades, triggers, and any combination thereof.



Abo-Typ
Basic
B
Premium
Basc

- Humans are rather inconsistent
- Databases were designed to significantly increase data consistency

Qualities | Consistency

ACID

Atomicity
Requires that each transaction be "all or nothing“

Consistency
Ensures that any transaction will bring the database from one valid state to another.

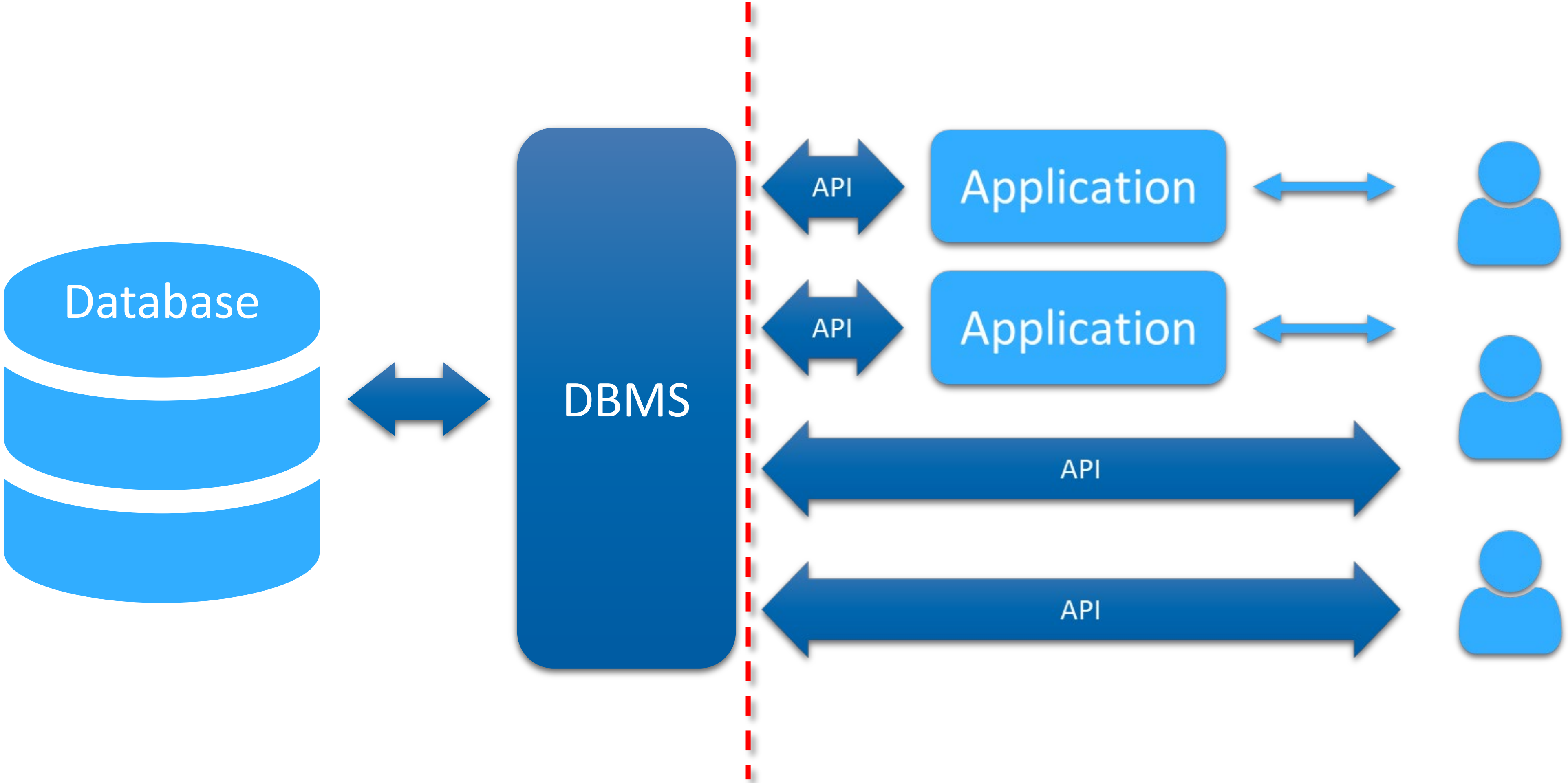
Isolation
Ensures that the concurrent execution of transactions results in a system state that would be obtained if transactions were executed sequentially.

Durability
Ensures that once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or errors.

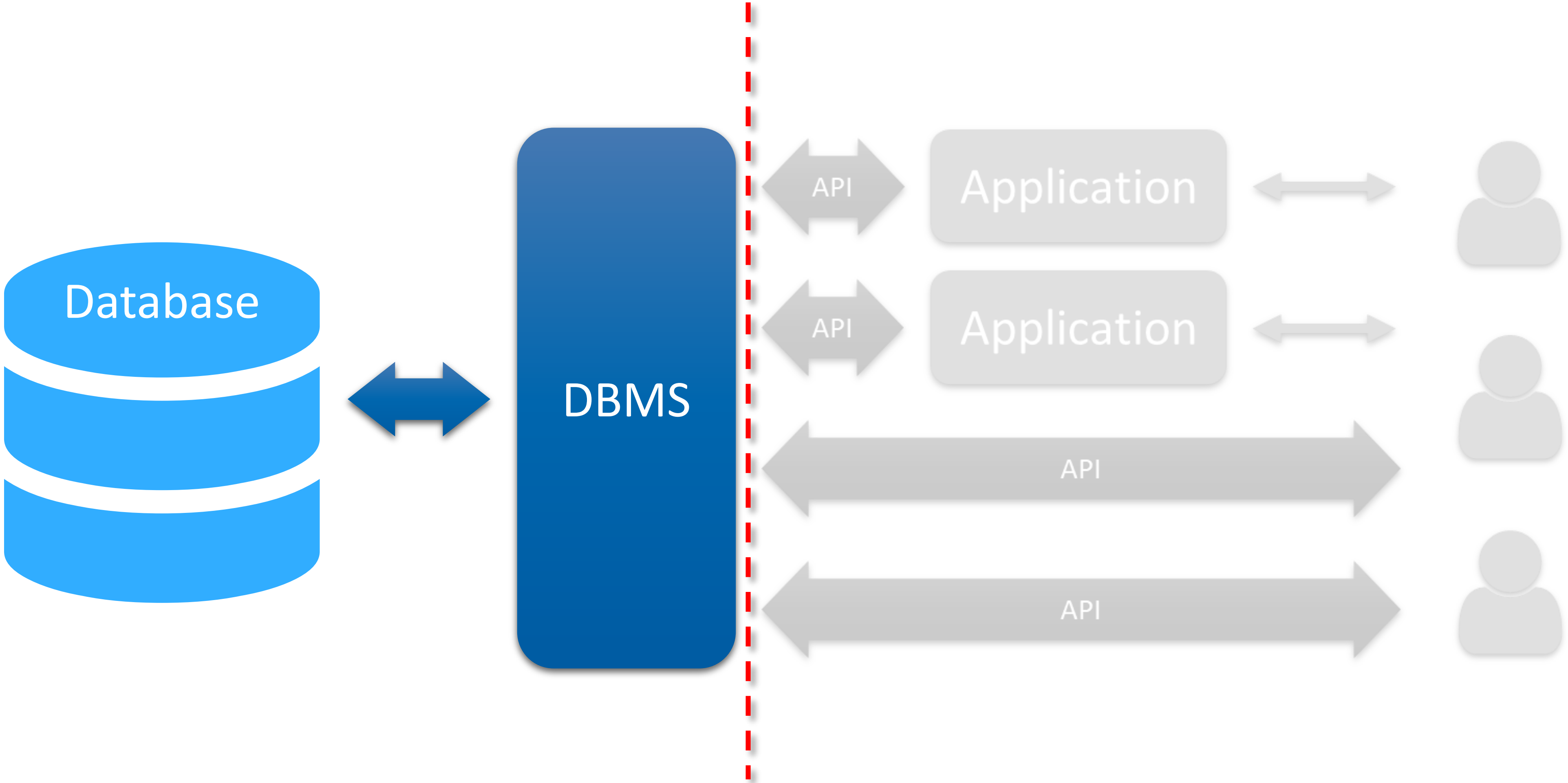
Qualities | Performance

- Performance is not a high priority purpose of a databases
- Database APIs usually come already with the best performance to retrieve, filter, and combine data
- High performance is only achieved if the database concept is fully understood and applied

Qualities | Independence

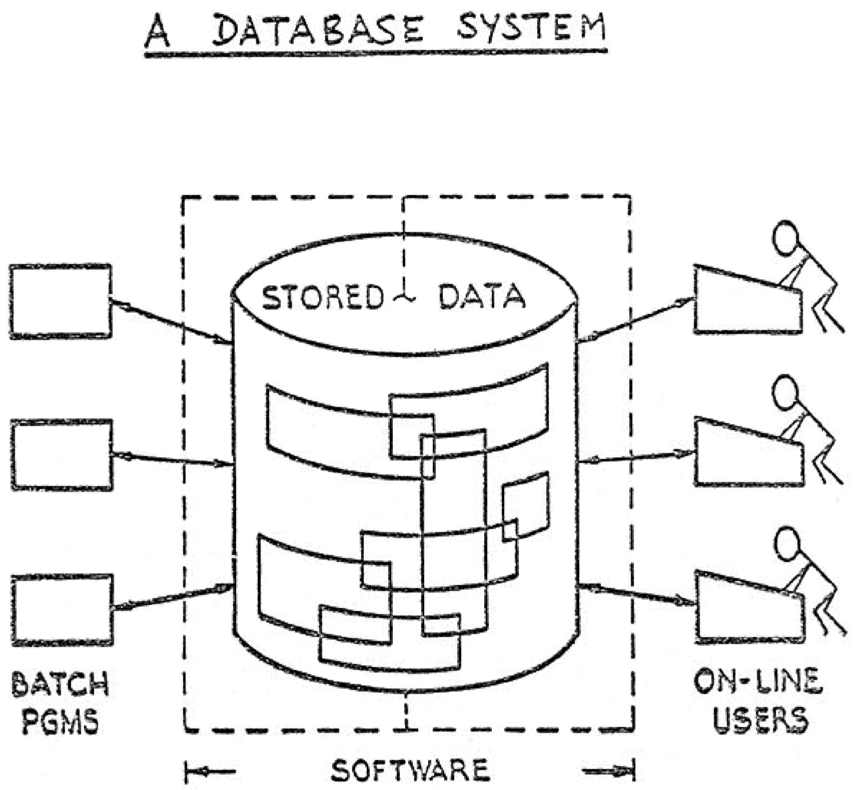


Qualities | Independence



Qualities | Abstraction

In 1970, Edgar F. Codd, an Oxford-educated mathematician working at the IBM San Jose Research Lab, published a paper showing how information stored in large databases could be accessed ***without knowing how the information was structured or where it resided in the database***.

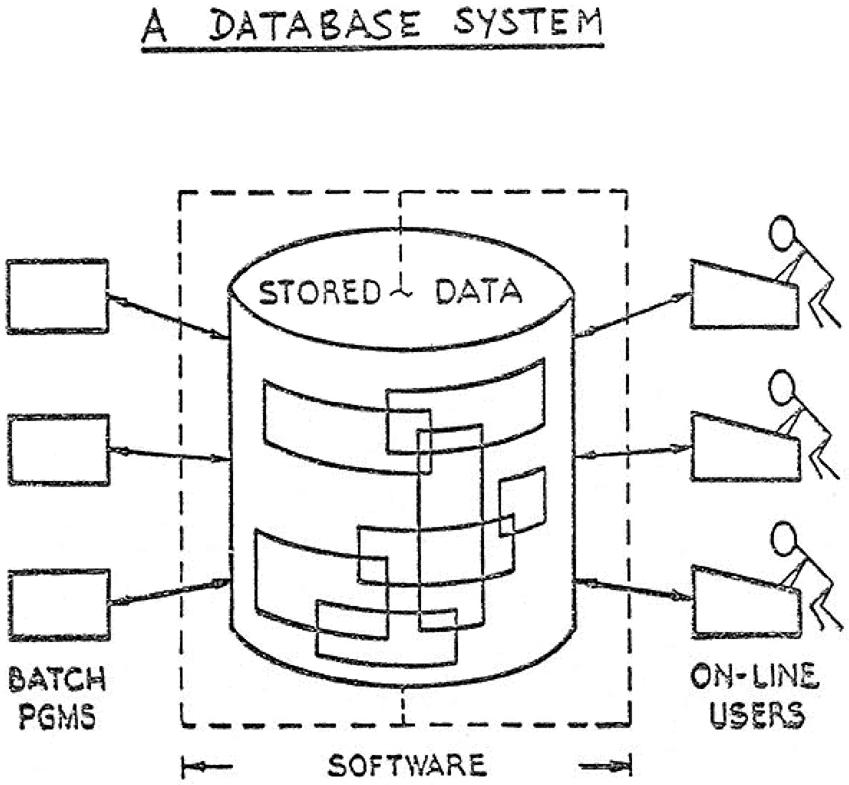


Qualities | Abstraction

Data in a database is still stored in files. However, the user does not have to know how and where the data is stored, or how the data is organized in the files.

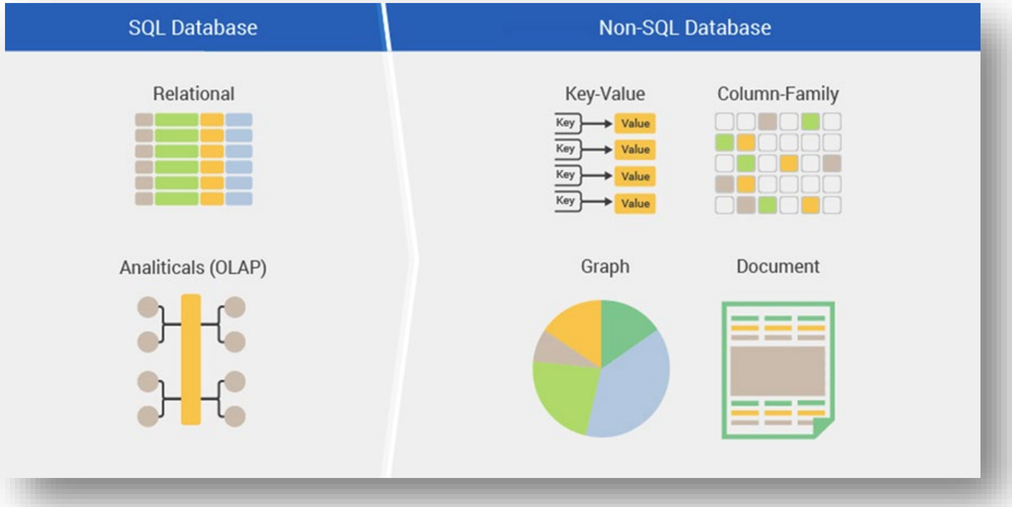
The database-management systems creates an abstraction layer to access data.

Just give me all the data that fulfills the required conditions



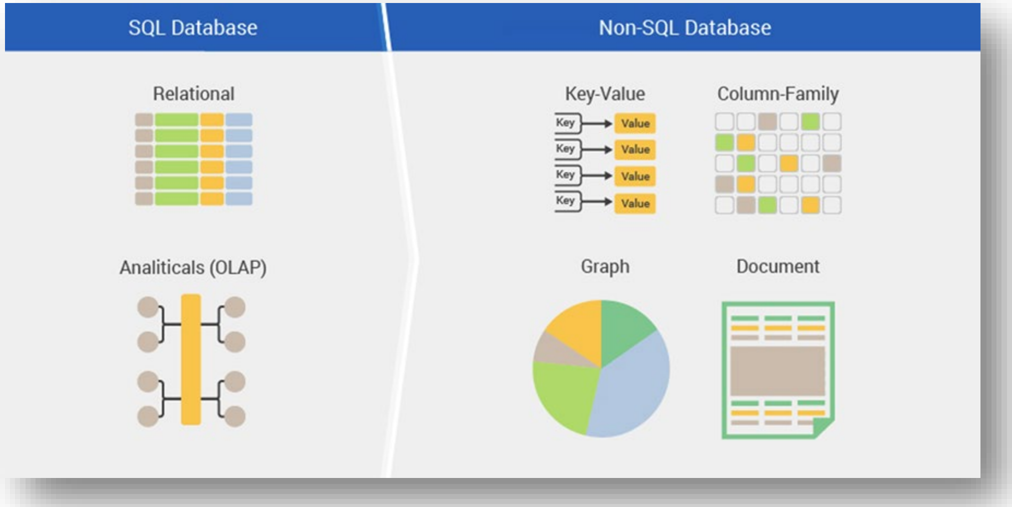
Content

- Why?
- Qualities
- Types

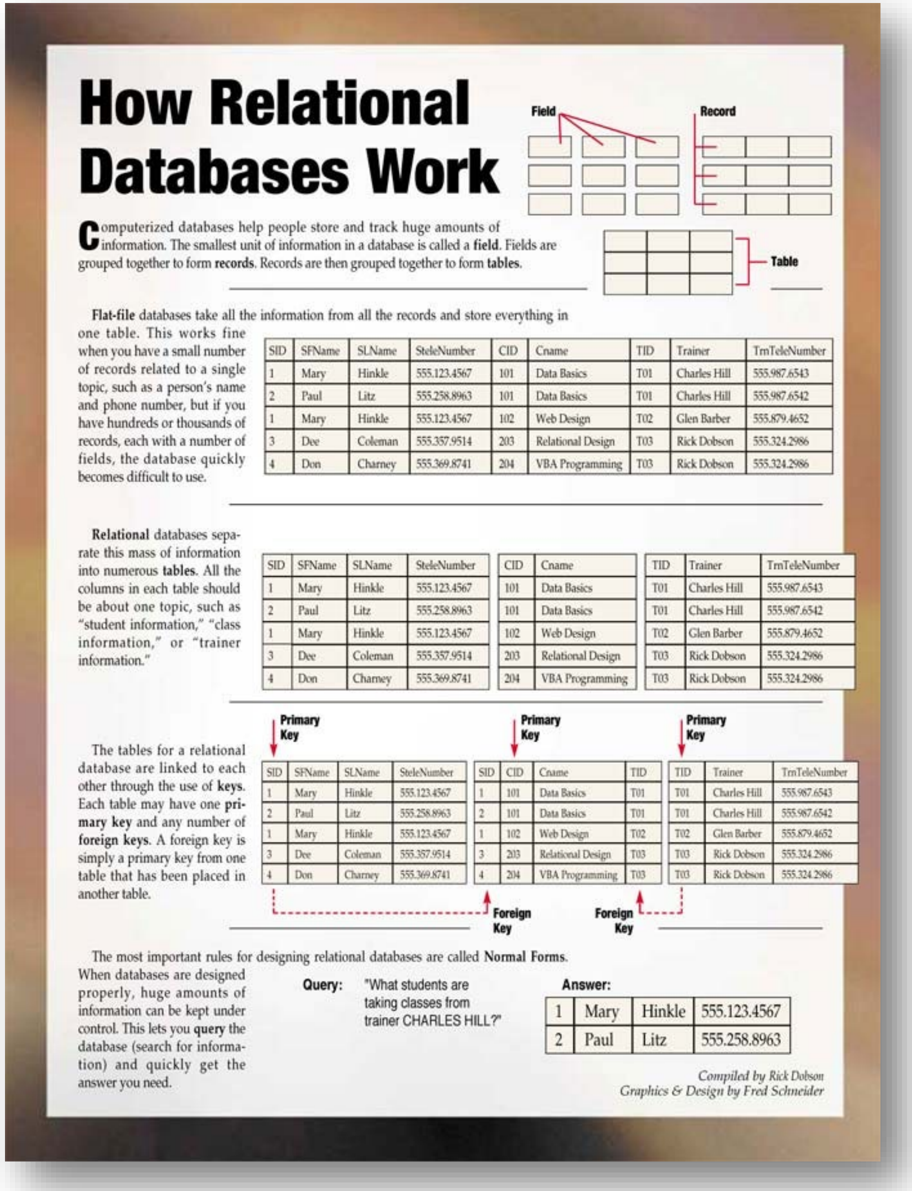


Types

- Relational
- Analytical
- Key-Value
- Column-Family
- Graph
- Document



Types | Relational



Field
Record
Table
Primary Key
Primary Key
Primary Key
Foreign Key
Foreign Key

How Relational Databases Work

Computerized databases help people store and track huge amounts of information. The smallest unit of information in a database is called a **field**. Fields are grouped together to form **records**. Records are then grouped together to form **tables**.

Flat-file databases take all the information from all the records and store everything in one table. This works fine when you have a small number of records related to a single topic, such as a person's name and phone number, but if you have hundreds or thousands of records, each with a number of fields, the database quickly becomes difficult to use.

SID	SFName	SLName	SteleNumber	CID	Cname	TID	Trainer	TrnTeleNumber
1	Mary	Hinkle	555.123.4567	101	Data Basics	T01	Charles Hill	555.987.6543
2	Paul	Litz	555.258.8963	101	Data Basics	T01	Charles Hill	555.987.6542
1	Mary	Hinkle	555.123.4567	102	Web Design	T02	Glen Barber	555.879.4652
3	Dee	Coleman	555.357.9514	203	Relational Design	T03	Rick Dobson	555.324.2986
4	Don	Charney	555.369.8741	204	VBA Programming	T03	Rick Dobson	555.324.2986

Relational databases separate this mass of information into numerous **tables**. All the columns in each table should be about one topic, such as "student information," "class information," or "trainer information."

SID	SFName	SLName	SteleNumber
1	Mary	Hinkle	555.123.4567
2	Paul	Litz	555.258.8963
1	Mary	Hinkle	555.123.4567
3	Dee	Coleman	555.357.9514
4	Don	Charney	555.369.8741

CID	Cname
101	Data Basics
101	Data Basics
102	Web Design
203	Relational Design
204	VBA Programming

TID	Trainer	TrnTeleNumber
T01	Charles Hill	555.987.6543
T01	Charles Hill	555.987.6542
T02	Glen Barber	555.879.4652
T03	Rick Dobson	555.324.2986
T03	Rick Dobson	555.324.2986

The tables for a relational database are linked to each other through the use of **keys**. Each table may have one **primary key** and any number of **foreign keys**. A foreign key is simply a primary key from one table that has been placed in another table.

SID	SFName	SLName	SteleNumber
1	Mary	Hinkle	555.123.4567
2	Paul	Litz	555.258.8963
1	Mary	Hinkle	555.123.4567
3	Dee	Coleman	555.357.9514
4	Don	Charney	555.369.8741

SID	CID	Cname	TID
1	101	Data Basics	T01
2	101	Data Basics	T01
1	102	Web Design	T02
3	203	Relational Design	T03
4	204	VBA Programming	T03

TID	Trainer	TrnTeleNumber
T01	Charles Hill	555.987.6543
T01	Charles Hill	555.987.6542
T02	Glen Barber	555.879.4652
T03	Rick Dobson	555.324.2986
T03	Rick Dobson	555.324.2986

The most important rules for designing relational databases are called **Normal Forms**. When databases are designed properly, huge amounts of information can be kept under control. This lets you **query** the database (search for information) and quickly get the answer you need.

Query: "What students are taking classes from trainer CHARLES HILL?"

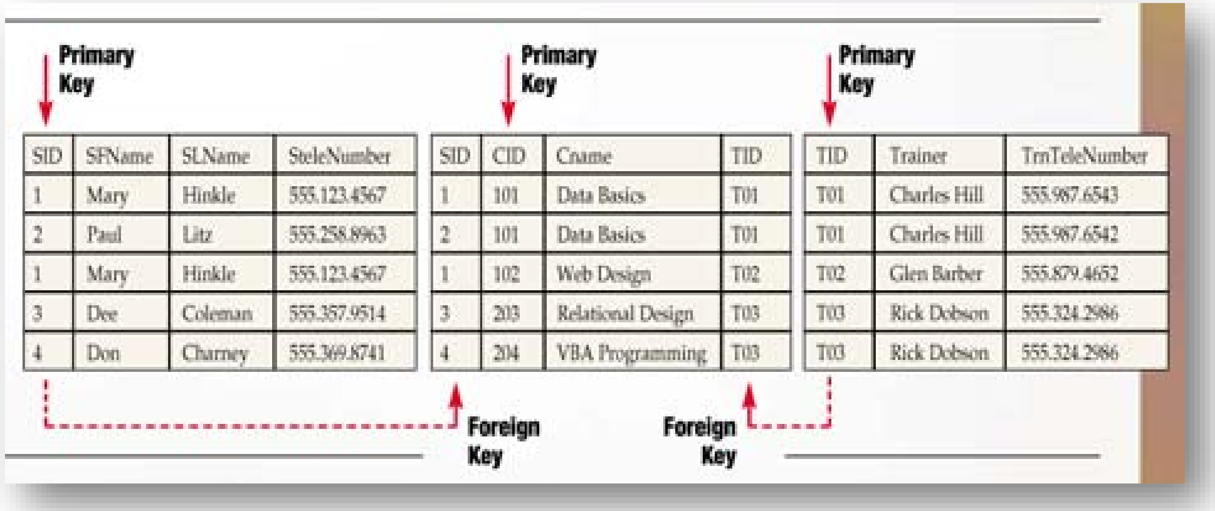
Answer:

1	Mary	Hinkle	555.123.4567
2	Paul	Litz	555.258.8963

Compiled by Rick Dobson
Graphics & Design by Fred Schneider

Types | Relational

- Data is stored in *records*
- Records contain one or many *attributes* (single value)
- Each record contains a unique *key*
- Records are linked to each other by using the keys



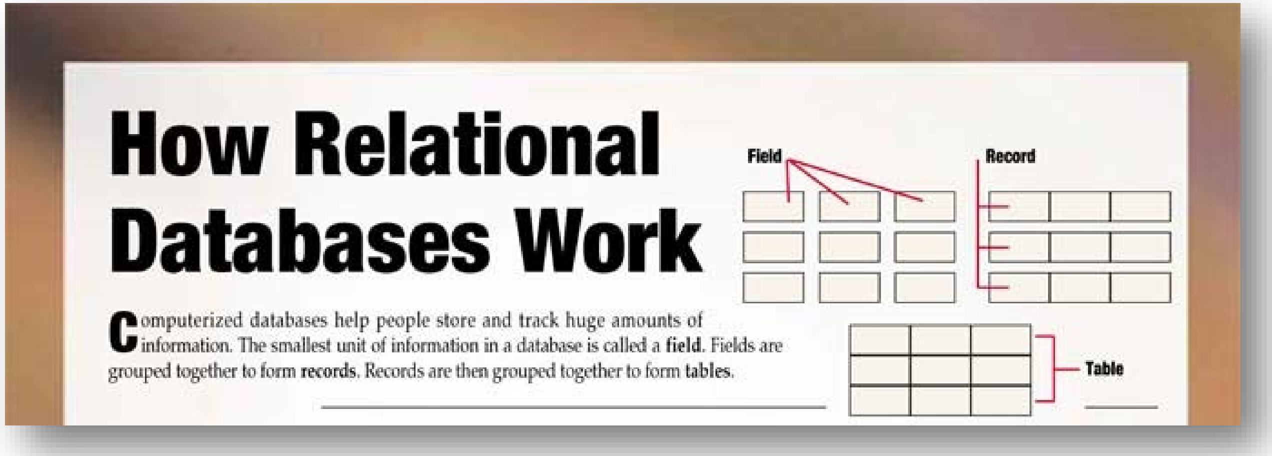
SID	SFName	SLName	SteleNumber
1	Mary	Hinkle	555.123.4567
2	Paul	Litz	555.258.8963
1	Mary	Hinkle	555.123.4567
3	Dee	Coleman	555.357.9514
4	Don	Charney	555.369.8741

SID	CID	Cname	TID
1	101	Data Basics	T01
2	101	Data Basics	T01
1	102	Web Design	T02
3	203	Relational Design	T03
4	204	VBA Programming	T03

TID	Trainer	TrnTeleNumber
T01	Charles Hill	555.987.6543
T01	Charles Hill	555.987.6542
T02	Glen Barber	555.879.4652
T03	Rick Dobson	555.324.2986
T03	Rick Dobson	555.324.2986

Types | Relational

- The most popular database type
- Records are stored in tables
- Allows many-to-many relationships
- Use the standardized query language SQL



How Relational Databases Work

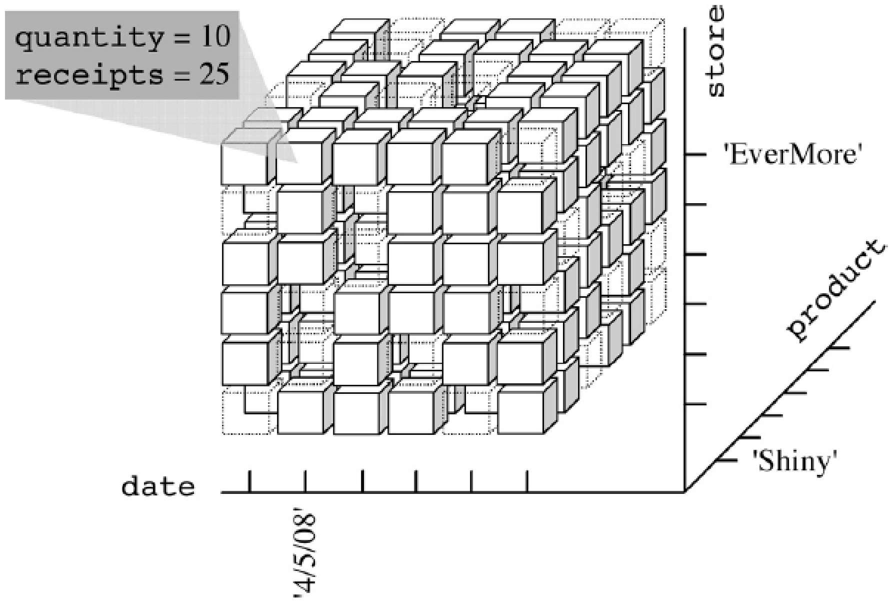
Computerized databases help people store and track huge amounts of information. The smallest unit of information in a database is called a **field**. Fields are grouped together to form **records**. Records are then grouped together to form **tables**.

Field
Record
Table

Types | Analytical

Also relational databases but…

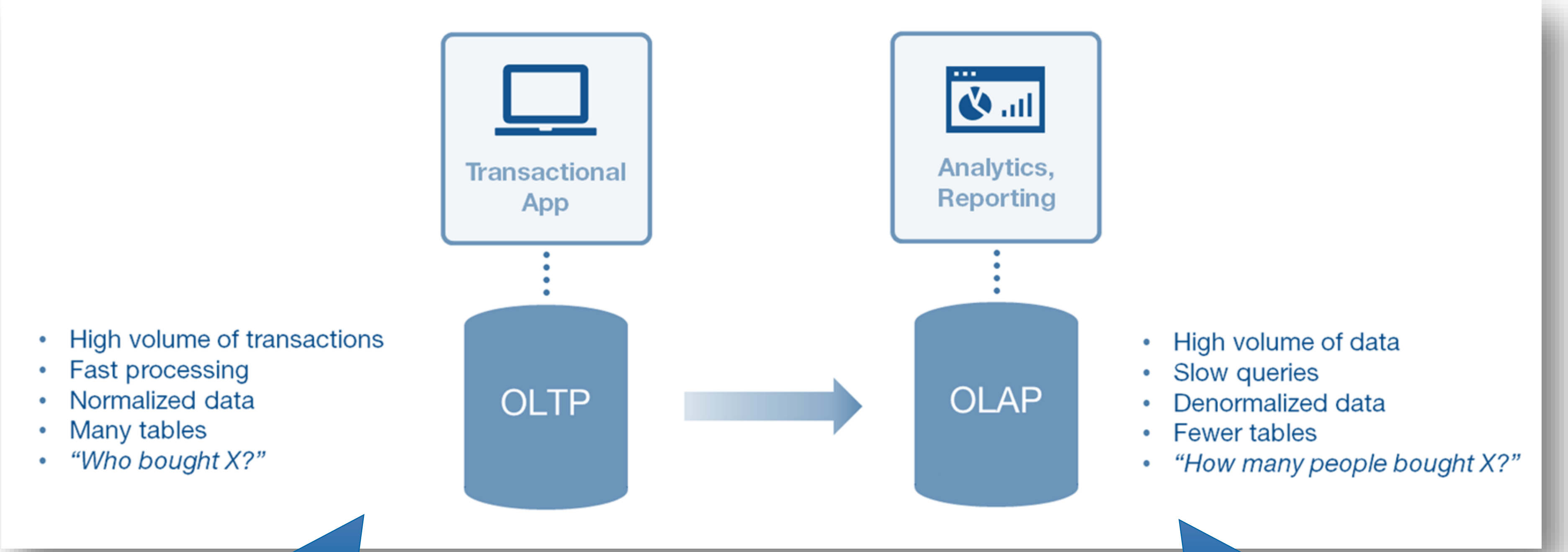
- Model is question-driven (i.e. optimized for a predefined analysis)
- Data is stored in tables representing multidimensional data cubes
- Database is not normalized
- Used in a classical data warehouses
- Used for OLAP (see next slide)



Types | Analytical

On-**Line** **Transaction** **Processing**

On-**Line** **Analytical** **Processing**



- High volume of transactions
- Fast processing
- Normalized data
- Many tables
- *“Who bought X?”*

- High volume of data
- Slow queries
- Denormalized data
- Fewer tables
- *“How many people bought X?”*

Ask me what’s happening

Ask me what will happen

Types | Key-Value

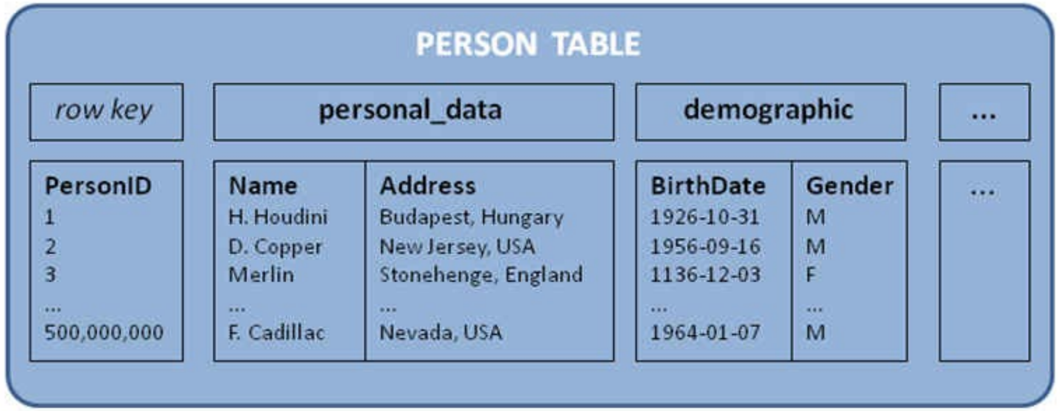
- Data is stored in a key-value pair
- Similar to a dictionary (as in Python)
- Keys must be unique, single identifiers
- Values can be anything (single value, object, record, file, etc.)
- The value’s datatype is not known to the database
- There are no database managed relationships

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

Types | Column-Family

- Data is stored in tables
- Records are modeled as rows
- Attributes are stored in columns

like in a relational database, but...

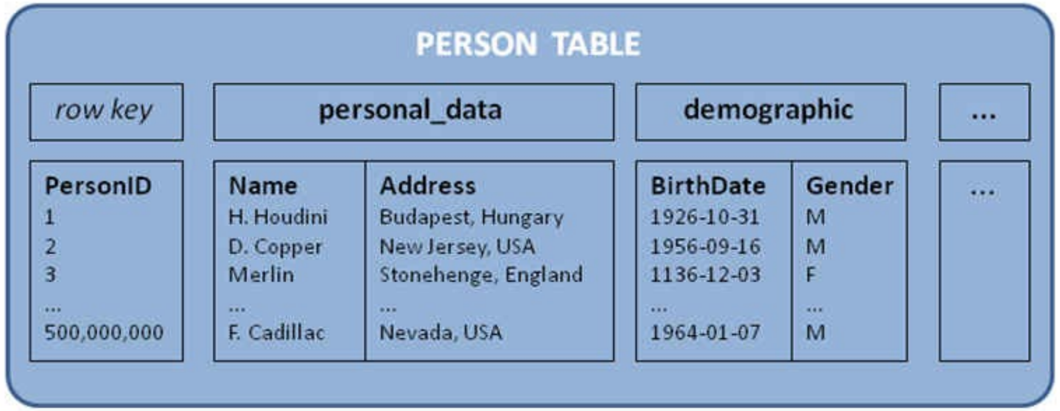


PERSON TABLE

row key	personal_data		demographic		...
PersonID	**Name**	**Address**	**BirthDate**	**Gender**	...
1	H. Houdini	Budapest, Hungary	1926-10-31	M	
2	D. Copper	New Jersey, USA	1956-09-16	M	
3	Merlin	Stonehenge, England	1136-12-03	F	
...	...	...	...	...	
500,000,000	F. Cadillac	Nevada, USA	1964-01-07	M	

Types | Column-Family

- Columns are grouped in column families
- Only column families have to be defined on creation
- New columns can be added on-the-fly on a record-basis
- Data is stored column-wise
- There are no database managed relationships

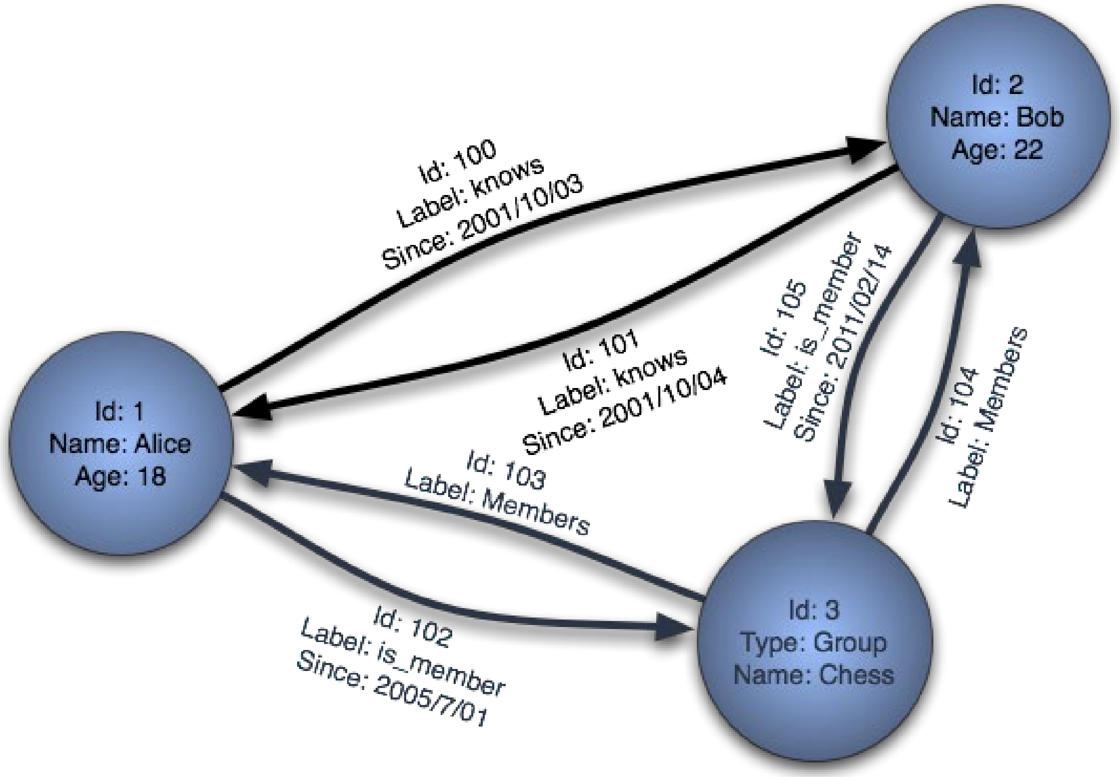


PERSON TABLE

row key	personal_data		demographic		...
PersonID	**Name**	**Address**	**BirthDate**	**Gender**	...
1	H. Houdini	Budapest, Hungary	1926-10-31	M	
2	D. Copper	New Jersey, USA	1956-09-16	M	
3	Merlin	Stonehenge, England	1136-12-03	F	
...	...	...	...	...	
500,000,000	F. Cadillac	Nevada, USA	1964-01-07	M	

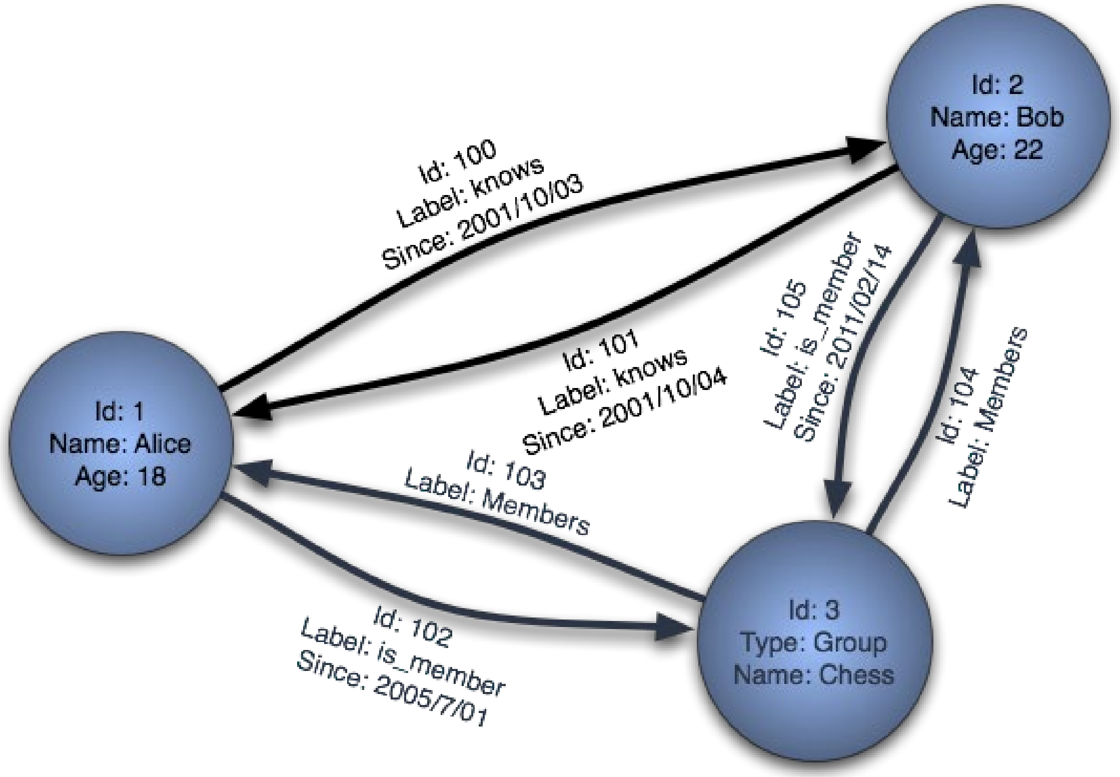
Types | Graph

- Data is stored in nodes
- Nodes contain one or many properties (key-value-pairs)
- Relationships are modeled as directed edges
- Edges contain one or many properties (key-value-pairs)
- Nodes and edges have a label and an ID



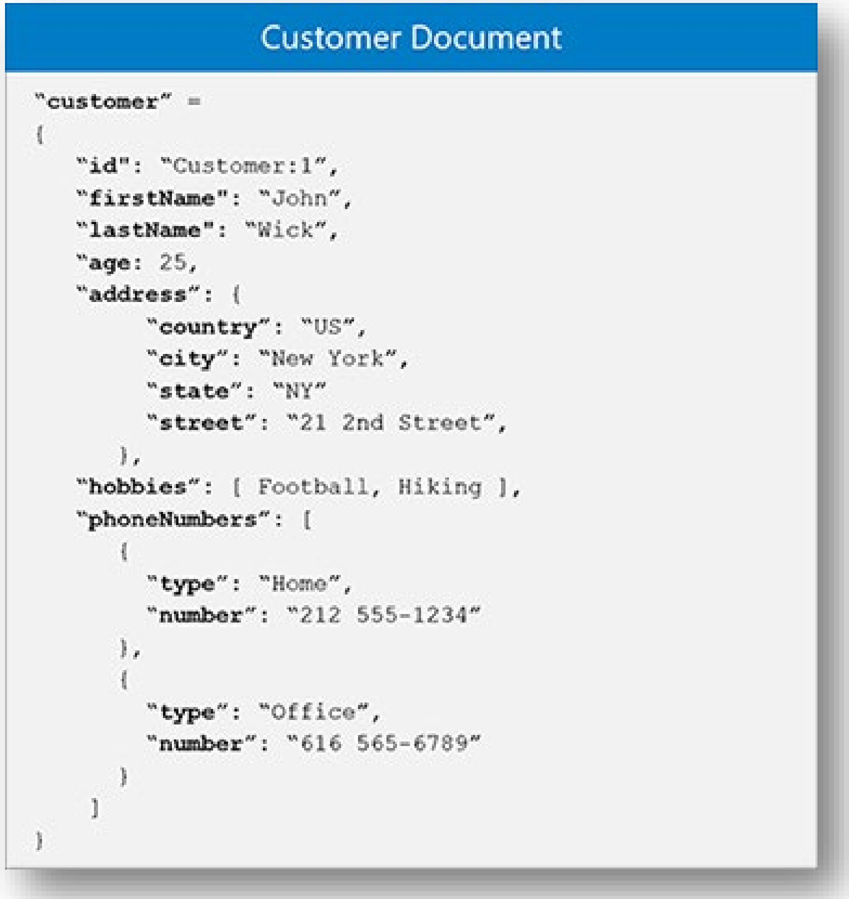
Types | Graph

- Allow simple and fast retrieval of complex structures
- Semantic queries
- Needs special query language (e.g. SPARQL)
- Popularity is growing



Types | Document

- Data is stored in documents with a given data format
- Indexing based on document properties (filenames are irrelevant)
- Document structure is not fixed
- Allows defining rules based on the content
- There are no database managed relationships



Customer Document

```
“customer” =
{
   “id": “Customer:1”,
   “firstName": “John”,
   “lastName": “Wick”,
   “age: 25,
   “address”: {
        “country”: “US”,
        “city”: “New York”,
        “state”: “NY”
        “street”: “21 2nd Street”,
      },
   “hobbies”: [ Football, Hiking ],
   “phoneNumbers”: [
      {
        “type”: “Home”,
        “number”: “212 555-1234”
      },
      {
        “type”: “Office”,
        “number”: “616 565-6789”
      }
    ]
}
```