Relational Database Model

Sławek Staworko

Univ. Lille 3

2018

How to model a database?

Relational model

- introduced by E. Codd in 1969
- using structure and language consistent with first-order logic
- lays foundation to relational databases and SQL

Declarative method

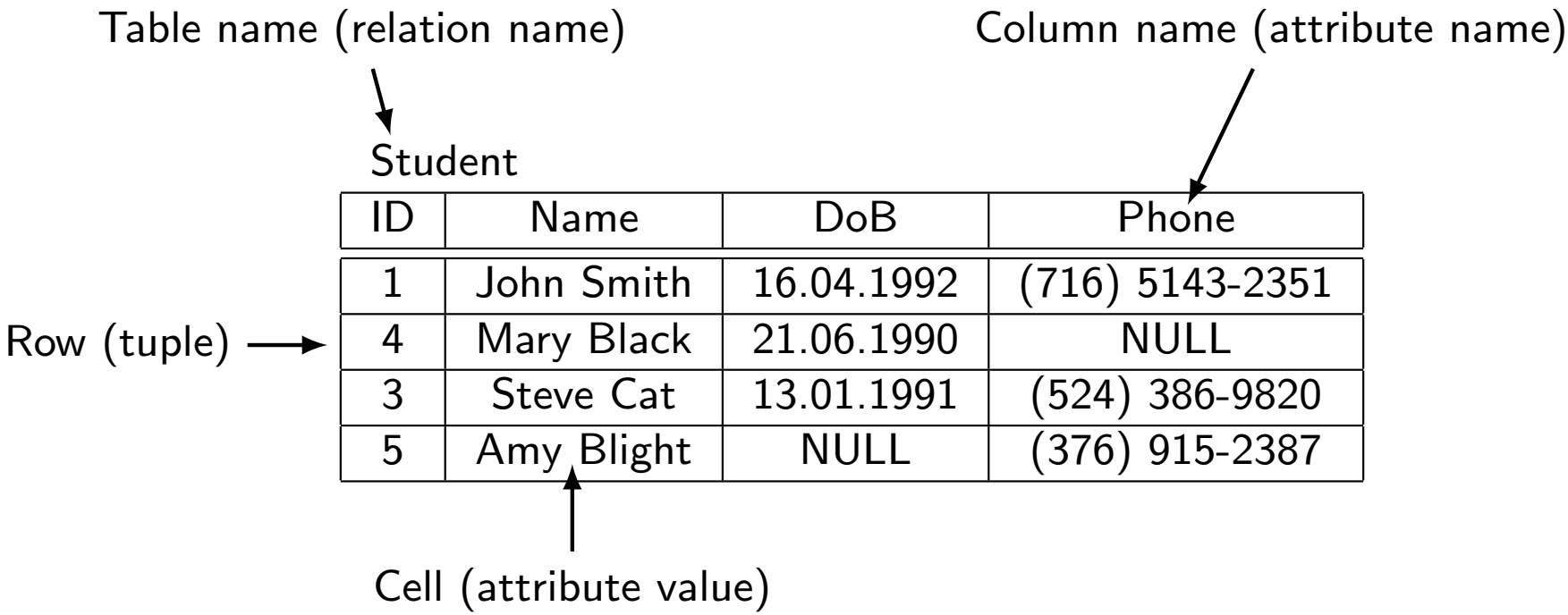
- high-level abstraction
- you express what you know (facts)
- you express what you want to find out (queries)
- the system takes care of implementation details

Relational databases

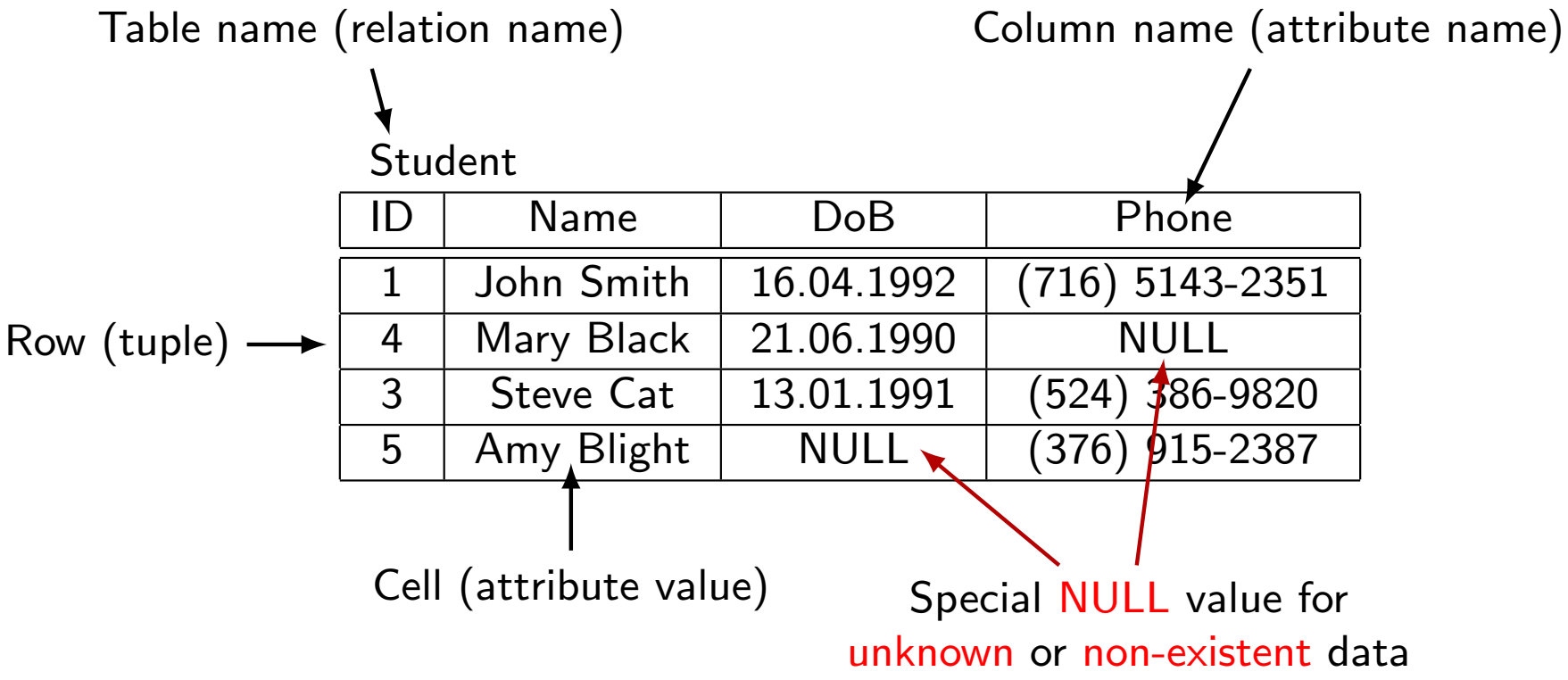
Student

ID	Name	DoB	Phone
1	John Smith	16.04.1992	(716) 5143-2351
4	Mary Black	21.06.1990	NULL
3	Steve Cat	13.01.1991	(524) 386-9820
5	Amy Blight	NULL	(376) 915-2387

Relational databases



Relational databases



Relational databases

Relation Schema

Student

ID	Name	DoB	Phone
1	John Smith	16.04.1992	(716) 5143-2351
4	Mary Black	21.06.1990	NULL
3	Steve Cat	13.01.1991	(524) 386-9820
5	Amy Blight	NULL	(376) 915-2387

Relation Instance

Relational databases

Student(ID, Name, DoB, Phone)

Relation Schema

Student

ID	Name	DoB	Phone
1	John Smith	16.04.1992	(716) 5143-2351
4	Mary Black	21.06.1990	NULL
3	Steve Cat	13.01.1991	(524) 386-9820
5	Amy Blight	NULL	(376) 915-2387

Relation Instance

Integrity Constraints

Constraints are restrictions on database instances

They restrict the values that can be used

- attribute/column types
- whether NULL values can be used
- whether values can repeat in a table (key)
- whether values have to be referenced from another table (foreign key)

Functional dependencies

Functional dependency $R : X \rightarrow Y$

- R is a relation name, X and Y are sets of attributes of R
- there cannot be two tuples that have the same values on X but different values on Y

$$\text{Student} : \text{Post Code} \rightarrow \text{City}$$

Student

ID	Name	Post Code	City
1	John Smith	59000	Lille
4	Mary Black	59800	Lille
3	Steve Cat	59000	Lille
5	Amy Blight	62000	Arras

Functional dependencies

Functional dependency $R : X \rightarrow Y$

- R is a relation name, X and Y are sets of attributes of R
- there cannot be two tuples that have the same values on X but different values on Y

Student : Post Code $\rightarrow$ City

Student

ID	Name	Post Code	City
1	John Smith	59000	Lille
4	Mary Black	59800	Lille
3	Steve Cat	59000	Lille
5	Amy Blight	62000	Arras
6	Pam Anders	59000	Wazemmes

Two tuples with the same Post Code but different City

Keys

Key is a specific type of functional dependency

- let R be a relation with attributes U and $K \subseteq U$
- K is a key of R if $R : K \rightarrow U$
- a valid instance cannot have two tuples with the same value on K

$$\text{Student} : \text{ID} \rightarrow \text{ID Name DoB Phone}$$

Student

<u>ID</u>	Name	DoB	Phone
1	John Smith	16.04.1992	(716) 5143-2351
4	Mary Black	21.06.1990	NULL
3	Steve Cat	13.01.1991	(524) 386-9820
5	Amy Blight	NULL	(376) 915-2387

Primary Key

- one selected key of a relation (all others are called secondary)
- <u>underlined</u> in the schema

Inclusion dependencies

Inclusion dependency $R[X] \subseteq P[Y]$

- R and P are relation names, X is a set of attributes of R, Y is a set of attributes of P (X and Y have the same number of elements)
- the values on X in any tuple of R must be used on Y in some tuple of P

$$\text{Course[Prof\_ID]} \subseteq \text{Prof[ID]}$$

Course

Name	Room	Prof_ID
AI	A2.416	2
BDD	B3.245	3
R	A1.425	3

Prof

ID	Name	Office
1	Marc	C.40
2	Fabien	C.41
3	Sławek	C.42

Foreign key constraint

$R[X] \subseteq P[Y]$, where Y is the primary key of P

Inclusion dependencies

Inclusion dependency $R[X] \subseteq P[Y]$

- R and P are relation names, X is a set of attributes of R, Y is a set of attributes of P (X and Y have the same number of elements)
- the values on X in any tuple of R must be used on Y in some tuple of P

$$\text{Course[Prof\_ID]} \subseteq \text{Prof[ID]}$$

Course

Name	Room	Prof_ID
AI	A2.416	2
BDD	B3.245	3
R	A1.425	3
HTML	A1.425	13

Prof

ID	Name	Office
1	Marc	C.40
2	Fabien	C.41
3	Sławek	C.42

There is not professor with ID=13 in our database

Foreign key constraint

$R[X] \subseteq P[Y]$, where Y is the primary key of P

Constraints in SQL

```
CREATE TABLE VILLE (
    NOM TEXT,
    CP TEXT,
    TEL_MAIRIE TEXT NOT NULL,

    PRIMARY KEY (NOM,CP)
);

CREATE TABLE ETUDIANT (
    ID INTEGER PRIMARY KEY AUTOINCREMENT,
    NOM TEXT NOT NULL,
    NSS TEXT UNIQUE,
    SEXE TEXT CHECK(SEXE IN ('F','M')),
    TEL_PORT TEXT,
    TEL_FIXE TEXT,
    M_POTE INT REFERENCES ETUDIANT(ID),
    VILLE TEXT,
    CP TEXT,

    CHECK (TEL_PORT IS NOT NULL OR TEL_FIXE IS NOT NULL),
    FOREIGN KEY (VILLE,CP) REFERENCES VILLE(NOM,CP)
);
```