3. The relational model

3.1. Relational data structure 3
3.2. Relational data integrity 6
3.2.1. Primary key 6
3.2.2. Foreign key 8
3.3. Relational algebra 10
Bibliography 17

1970 - E.F.Codd: A relational data model for large shared data banks. CACM 13, No.6, 1970.

Contributions of the paper:

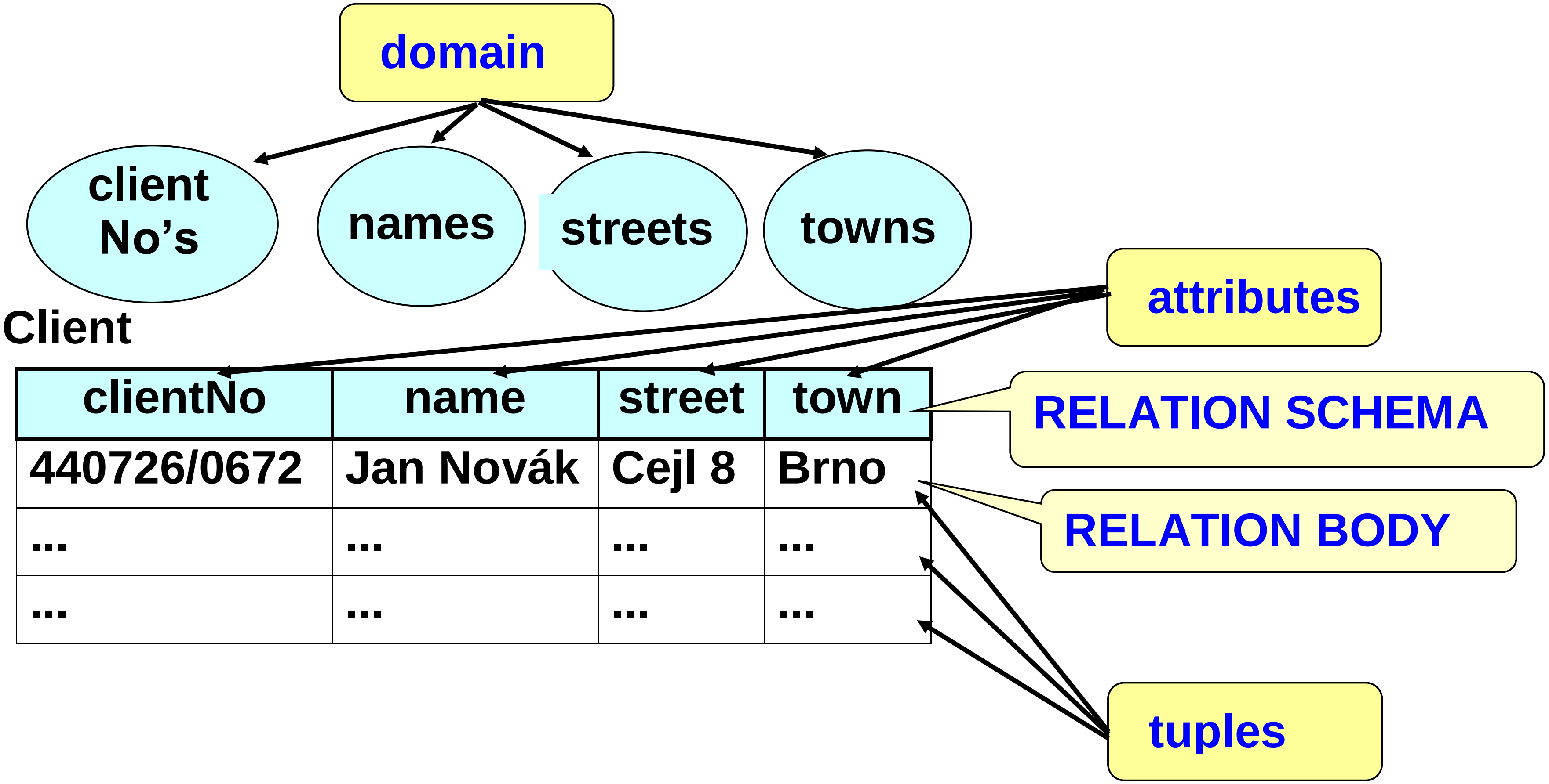
- **Separation of logical structure of data from implementation.**
- **Mathematical support for data manipulation.**
- **Mathematical support for avoiding redundancy in database logical schema design.**

Components of the relational model

- **Relational data structure.**
- **General integrity constraints in relational databases.**
- **Data manipulation in relational databases.**

3.1. Relational data structure

RELATION



Client

clientNo	name	street	town
440726/0672	Jan Novák	Cejl 8	Brno
...	...	...	...
...	...	...	...

Domain **- a named set of <u>scalar values.</u> all of the same type.**

Ex) Domain of all possible names of towns

Scalar value **- the smallest semantic unit of data (atomic, cannot be decomposed without losing its meaning, no internal structure).**

Ex) Josef Novák – provided that we do not distinguish first and last names.

Composite domain **– a domain composed from other domains.**

Ex) (Josef, Novák) - provided that we distinguish first and last names.

Relation

Relation **on domains $D_1, D_2, \ldots, D_n$ is a structure $\mathbf{R} = (R, R^*)$, where $R = R(A_1{:}D_1, A_2{:}D_2, \ldots A_n{:}D_n)$ is a *relation schema* and $R^* \subseteq D_1 \times D_2 \times \ldots \times D_n$ is a relation body. We often write the schema only as $R(A_1, A_2, \ldots, A_n)$. The number of attributes n of a relation is referred to as the *degree of the relation*, cardinality of its body $m = |R^*|$ is called *cardinality of the relation*.**

Relationship of the notion "relation" and the notion "table"

relation	**fundamental concept of the relational model**
table	**the way how to represent (and visualize) relation**

Attribute ~ *a column of a table*, a tuple ~ *a row of a table*

Comment: The name ″relational model″ and ″relational database″ results from the fact that relation is fundamental abstract concept of the model and it is the only structure of the database recognized at a logical level.

Properties of the relation (in the theory of the relational model):

- **There are no duplicate tuples in a relation.**
- **Tuples are unordered, top to bottom.**
- **All attribute values are atomic (a relation is said to be normalized (in 1NF)). It is then simpler and simpler operations can be used**

Summary of fundamental properties of the relational data structure

- **Relational database is perceived by the user (or by application) as a collection of time-varying normalized relations.**
- **All data in the database is represented by explicit value, logical connections within and across relations are represented by explicit values; there are no user-visible pointers.**

3.2. Relational data integrity

- Constraints resulting from the real world represented in the database, they are specified by ***integrity rules***.

Types of integrity constraints:

- ***Database - specific*** **– specific for a particular database (e.g. weight cannot be negative).**
- ***General*** **- relevant to the integrity of every database based on the same model of data.**

- **General integrity constraints in the relational model concern primary and foreign keys.**

3.2.1. Primary key

A *candidate key* for a relation R is a subset of the set of attributes of with the following properties:

1. **uniqueness**
2. **irreducibility.**

- **There is at least one candidate key for each relation (in the theory).**

- **An attribute that is a component of a candidate key is called a *key attribute*.**
- **Candidate keys provide the basic tuple-level addressing mechanism.**

A primary key - one of candidate keys, the other candidate keys are sometimes called alternate keys.

- **The choice of the primary key from candidate keys is essentially arbitrary (normal basis is simplicity).**
- **The primary key is used to address tuples in the relation.**

Ex)

```
SELECT *
FROM Client
WHERE clientNo='440726/0672';
```

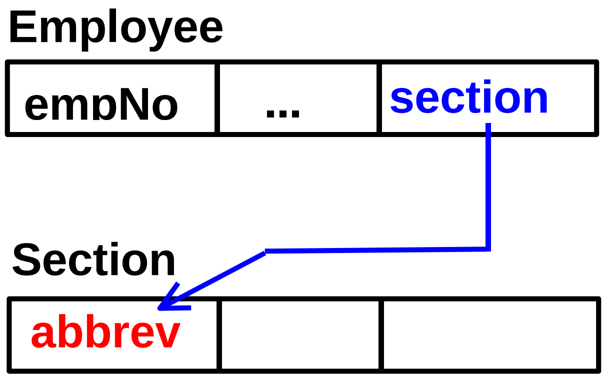
```
SELECT *
FROM Client
WHERE name = 'Jan Novak';
```

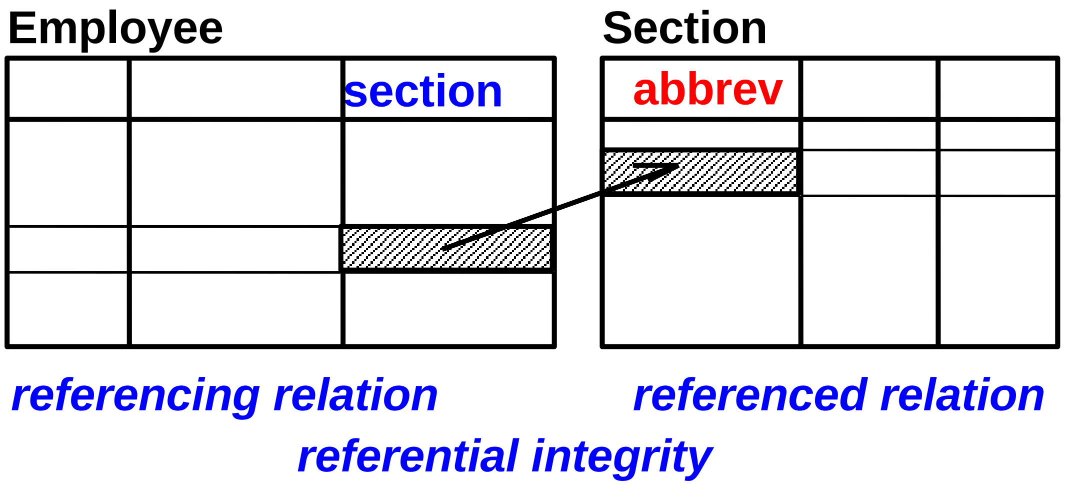
Entity integrity:

No component of the primary key of a base relation is allowed to accept nulls. (must not be NULL).

3.2.2. Foreign key

Ex) Consider the column Employee.section - Is a value ‘DEV’ valid?





A *foreign key* in a base relation R2 is an attribute of R2, say *FK*, such that:

1. **Each value of *FK* fully specified or fully null.**
2. **There exists a base relation R1 (not necessarily distinct of R2) with a candidate key *CK* such that each specified value of *FK* is identical to the value of *CK* in some tuple of R1.**

- **Foreign - to - candidate key matches represent certain relationships between tuples (it is a ″glue″ holding the database together).**

Referential integrity:

The database must not contain any unmatched foreign key values.

- **The user (database designer) should be able to specify what to do if the rule is not satisfied (it is out of the relational model).**
  - See referential actions in the CREATE TABLE statement of SQL.

3.3. Relational algebra

The relational algebra **is a structure *RA = ($\mathcal{R}$, O)*, where $\mathcal{R}$ a set of relations (a carrier of the algebra) and *O* is a set of operations that includes (as defined by Codd) two groups of operations:**

- The traditional set operations union, intersection, difference and Cartesian product.

- The special relational operations projection, selection (restriction), join and division.

Traditional set operations

- **The operations have the usual meaning, but their definitions are modified somewhat to take account of the fact that operands are relations, not arbitrary sets.**

Union **of relations R1 = (*R, R1*)* and R2 = (*R*, *R2**) with a schema *R* is a relation R1 *union* R2 = *(R, R1** $\cup$ *R2*).***

Similarly for *intersection* (R1 *intersect* R2*) and difference (*R1 *minus* R2*).*

Cartesian product **of relations R1 = (*R1*, *R1**) and R2 = (*R2*, *R2**) is a relationR1 *times* R2 = ((*R1*,*R2*), *R1* x R2**).**

Ex)

R1

A	B	C
0	a	d
1	a	e
2	b	f

R2

A	B	C
2	c	d
0	a	d
0	a	e

R1 *union* R2

A	B	C
0	a	d
1	a	e
2	b	f
2	c	d
0	a	e

R1 *intersect* R2

A	B	C
0	a	d

R1 *minus* R2

A	B	C
1	a	e
2	b	f

R1 *times* R2

AR1	BR1	CR1	AR2	BR2	CR2
0	a	d	2	c	d
0	a	d	0	a	d
0	a	d	0	a	e
1	a	e	2	c	d
...	...	...	...	...	...

Special relational operations

- **projection, selection (restriction) - unary; join, division – binary**

➢ **Projection**

The *projection* of relation R = (*R, R) on attributes *X, Y, ..., Z* is a relation R[*X, Y, ..., Z*] with a schema (*X, Y, ..., Z)* and a body containing all tuples *t* = *(x, y, …, z)* such that there is a tuple *t'* in *R** with the value of the attribute *X* equal *x, Y* equal *y*, ... *Z* equal *z*.**

Ex)

R

A	B	C	D	E
0	a	x	3	1
1	b	y	6	2
0	b	y	4	2

R [B,C,E]

B	C	E
a	x	1
b	y	2

➢ Selection (restriction)

Let θ stands for a scalar comparison operator (<, >, <>, =, etc.).
θ selection (restriction) of relation **R** = (*R*, *R**) on attributes *X* and *Y* is a relation

R *where X* θ *Y*,

with the same schema as **R** and a body consisting of all tuples $t \in R^*$ such that $x \; \theta \; y$ where *x* is the value of an attribute *X* and *y* is the value of an attribute *Y* in the tupple *t*.

- A scalar literal can be instead one of operands in the comparison.

Ex)

R

A	B	C	D	E
0	a	x	3	1
1	b	y	6	2
0	b	y	4	2

R *where A*=0

A	B	C	D	E
0	a	x	3	1
0	b	y	4	2

- A restriction condition can be extended to include Boolean combination of several comparisons:

R1 *where* c1 *and* c2 ≡ (**R1** *where* c1) *intersect* (**R1** *where* c2)

- **Similarly *or*, *not*.**
  - **Join**

Let R1 be a relation with a schema R1(*X1,X2,..,Xm,Y1,Y2,...,Yn)* and R2 be a relation with a schema (*Y1,Y2, ...,Yn, Z1,Z2,...,Zk)*.
Consider composite attributes *X*=(*X1,X2,...,Xm*), *Y*=(*Y1,Y2,...,Yn*) and *Z*=(*Z1,Z2,...,Zk*). Then a *natural join* of relations R1 and R2 is a relation

R1 join R2

with a schema (*X, Y, Z*) and a body consisting of all tuples *t = (x, y, z)* such that there is a tuple *t'* in R1* with the value *x* of the attribute *X* and with the value *y* of the attribute *Y,* and there is a tuple *t''* in *R2 with the value *y* of the attribute *Y* and the value *z* of the attribute Z.**

Ex)

R1

A	B	C
0	a	d
1	a	e
2	b	f

R2

C	D	E
e	1	0
d	1	1
d	0	1

R1 *join* R2

A	B	C	D	E
0	a	d	1	1
0	a	d	0	1
1	a	e	1	0

Comment: You can find the following symbols denoting operations of the relational algebra in literature:

σ_θ **(R)**	**R** *where* θ
$\Pi_{X,\,Y}$ **(R)**	**R** [*X*, *Y*]
R $\bowtie$ **S**	**R** *join* **S**
R $\div$ **S**	**R** *divideby* **S**

Minimum set of operations (primitive)

- **Union, difference, Cartesian product, projection, selection.**

Extended relational algebra

- **Other operations have been proposed later- e.g. (relational assignment, RENAME, aggregate functions,...)**

The importance of the relational algebra

- **It is an appropriate base for query processing optimization (the language of its expressions is a procedural query language)**
- **It is a reference for assessment and comparison of relational database languages.**

A database language is *relationally complete* if it is at least as powerful as the relational algebra.

Bibliography

1. Silberschatz, A., Korth H.F, Sudarshan, S.:Database System Concepts. Fourth Edition. McGRAW-HILL. 2001, str. 79 – 131.
2. Date C.J.: An Introduction to Database Systems. Sixth edition. Addison-Wesley, 1995, str. 79 – 218.