Reznichenko Valery
Organization of data and knowledge bases

Lecture 5. Relational Data Structure

National Aviation University
Computer Science Faculty
Department of Software Engineering

CONTENT

- Relation in mathematics
- Relation definition
- Domains, attributes, schemas and instances of relations in RDM
- Relations and tables
- Keys of relations

Nonformal introduction to relations

Relation is an association between any number of entities.

Like subject			**Is more**				**Supply**		
Who	What	First	Second	Who			Whom	What	Q-ty
John	DBMS	5	3	П1	К7	table	200		
Peter	C	17 5	П3	К14	door	150			
AnnXML		2 1	П18		К9	window	1000		

Form of representation:

- As a table
- By using a condition

Relation definition

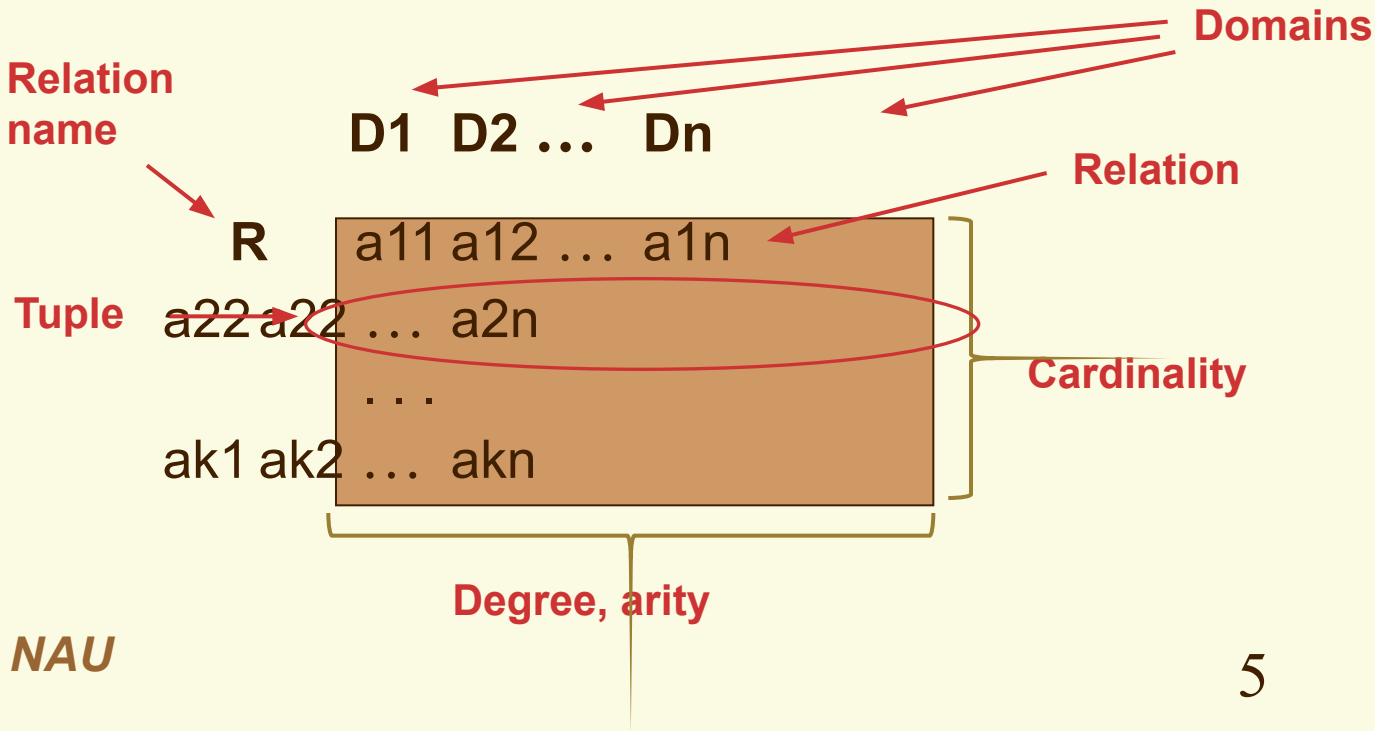
Let's given sets D1, D2,…, Dn (does not obligatory distinct). ***Relation*** R, defined on these sets, is a set of ordered n-tuples (d1, d2,…, dn), such that $d1 \in D1$, …, $dn \in Dn$

Lets given sets D1, D2,…, Dn (does not obligatory distinct). ***Cartesian product*** of these sets, denoted as $D1 \times D2 \times \ldots \times Dn$, is a set of all possible tuples (d1, d2,…, dn), such that $di \in Di$, i = 1,n.
R is a ***relation*** on the sets D1, D2,…, Dn, if:

$$R \subseteq D1 \times D2 \times \ldots \times Dn = \{(d1, d2, \ldots, dn) \mid di \in Di, i = 1,2,\ldots,n\}$$

Additional terms

- Sets D1, D2,…, Dn are called ***domains*** of the relation R .
- n is a ***degree*** of the relation R or its ***arity***.
- Number of tuples in a relation is called ***cardinality***.
- ***Tuple*** is a row of the relation.



Representation of binary relations

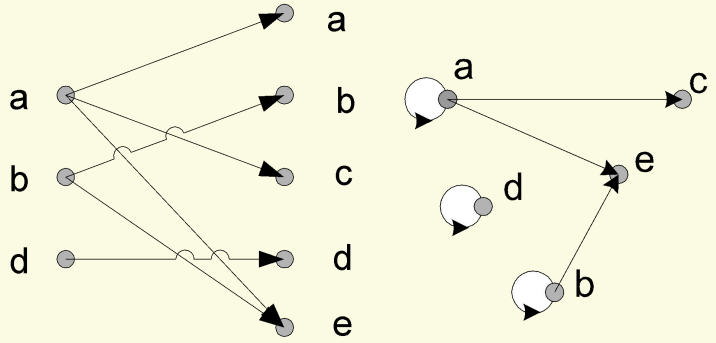
As a matrix

$$R = \begin{matrix} a & a \\ a & c \\ a & e \\ b & b \\ b & e \\ d & d \end{matrix}$$

As a table

R	a	b	c	d	e
a	*		*		*
b		*			*
d				*	

Graphical



As a logical condition: $R(x,y,...,z) = \{(x,y,...,z) \mid \varphi(x,y,...,z)\}$

Basic operations

Union: $R \cup S = \{t \mid t \in R \vee t \in S\}$

Intersection: $R \cap S = \{t \mid t \in R \,\&\, t \in S\}$

Negation: $\neg R = \{t \mid t \notin R\}$

Cartesian product: $R \times S = \{(r,s) \mid r \in R \,\&\, s \in S\}$

Property of binary relations

Reflexivity: Relation R is reflexive if:

$\forall a\ R(a, a)$.

Symmetry: Relation R is symmetric if

$\forall a \forall b\ (R(a, b) \Rightarrow R(b, a))$

Transitivity: Relation R transitive if:

$\forall a \forall b \forall c\ (R(a, b)\ \&\ R(b, c) \Rightarrow R(a, c))$.

Antisymmetry: Relation R is antisymmetric if:

$\forall a \forall b\ \ (R(a, b)\ \&\ R(b, a) \Rightarrow a = b)$.

Examples of binary relations

Relation **Look-Like(x,y)** is reflexive (any person looks like himself), symmetric (if b looks like d, then d looks like b), but not transitive (if have chain of pairs of similar persons it does not mean that persons at the ends of this chain are similar).

Relation **Is-Higher(x,y)** is transitive but not reflexive and symmetric.

Relation **Is-Equal (=)** is reflexive, symmetric and transitive.

Relation **Teach(x,y)** is not reflexive, symmetric and transitive.

Schema of a relation

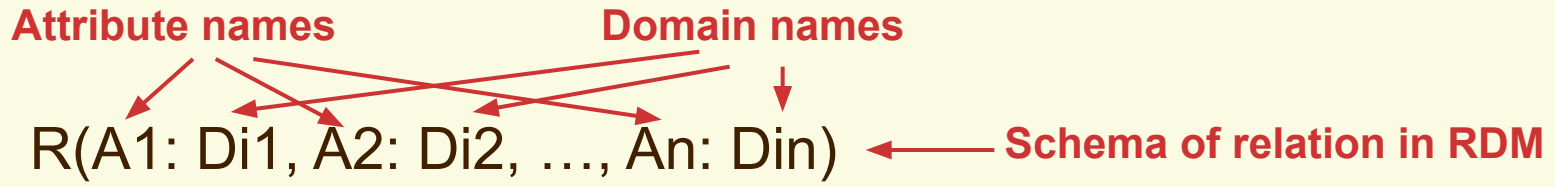
In mathematics order of columns is essential.

Is	More
5	3
17	5

	Is More
3	~~5~~
5	~~17~~

In relational data structure order of “columns” is not essential. It is achieved at the expense of introducing of the concept “attribute”.

Attribute – is semantically sensible names of the relation columns.



Property of attributes and schemas

- **Properties of the relation attributes:**
  - Each attribute of a relation has a name.
  - The set of allowed values for each attribute is called the domain of the attribute.
  - Different attributes may have the same domain.
  - Attribute values are required to be atomic, that is, indivisible.

Properties of the relation schema:

- Every schema has a name.
- Attribute names in schema must be unique.
- Order of attributes in schema is not fixed

Relation instance

Relation instance corresponds to the relation in mathematics with the only difference – the order of columns in the relation is not important.

Properties of a relation instance:

- Order of attributes is arbitrary, but it is defined by a relation schema.
- Order of tuples is arbitrary
  (tuples may be located in an arbitrary order)
- Tuples must be unique within the instance

Relational data structure

As ***relational structure*** the set of the *relational schema* and its *instance* (state).

The ***relational schema*** is a set of the schemas of relations:

$R_1(A_1,\ldots,A_n)$

$R_2(B_1,\ldots,B_k)$

…

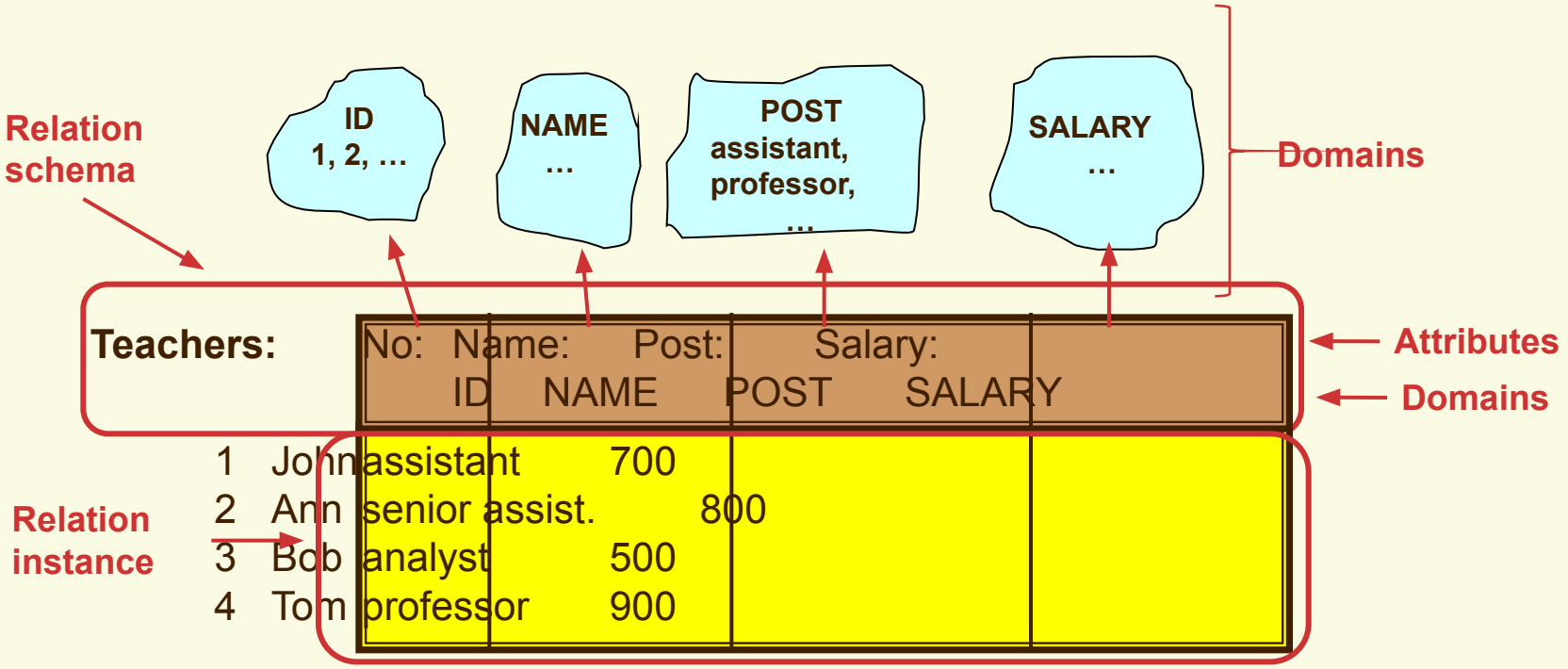
$R_n(K_1,\ldots,K_m)$

Instance of relational schema is a set of instances of the relations in relational schema .

The relational schema and its instance have the following properties:

- Name of the relations in relational schema must be unique

Relations and tables



Term correspondence

Formal term	Nonformal equivalent
domain	allowed values
attribute	column, field
relation	table
tuple	row, record
cardinality	number of rows
degree, arity,	number of columns
key	unique identifier

Keys

The ***key*** is a set of relation attributes that uniquely identify the tuples of the relation.

Assertion. Any relation has a key.

Example: In the relation **STUDENT(No, Name, Course)** set of attributes (**No, Name, Course**) are key because tuples of any relation are unique.

Assertion. Any relation may has many keys.

Simple and compound keys

The key is ***simple***, if it consists of one attribute.

The key ***compound*** if it consists of several attributes.

Example. In the relation:

STUDENT(ID-No, Name, Pasp-ser, Pasp-No, Course)

ID-No is a simple key and pair of attributes
(Pasp-ser, Pasp-No) is a compound key.

Redundant and minimal keys

Compound key is ***redundant*** (*not minimal*) if there is a subset of this key that is also a key. Redundant key is also called ***superkey.***

Example. In the relation:

STUDENT(ID-No, Name, Pasp-ser, Pasp-No, Course)
key (**Pasp-ser, Pasp-No, Course**) is redundant key because its subset (**Pasp-ser, Pasp-No, Course**) is also a key

Nonredundant key is called ***minimal***.

Primary key

A relation may have many minimal keys. All of them are called ***candidate keys.***

Example. Relation:

STUDENT (ID-No, Name, Pasp-ser, Pasp-No, Course)

has two minimal (that is why candidate) keys:

- **ID-No**
- **Pasp-ser, Pasp-No**

Among set of all candidate (minimal) keys only one is selected as a ***primary key.***

Properties of a primary key

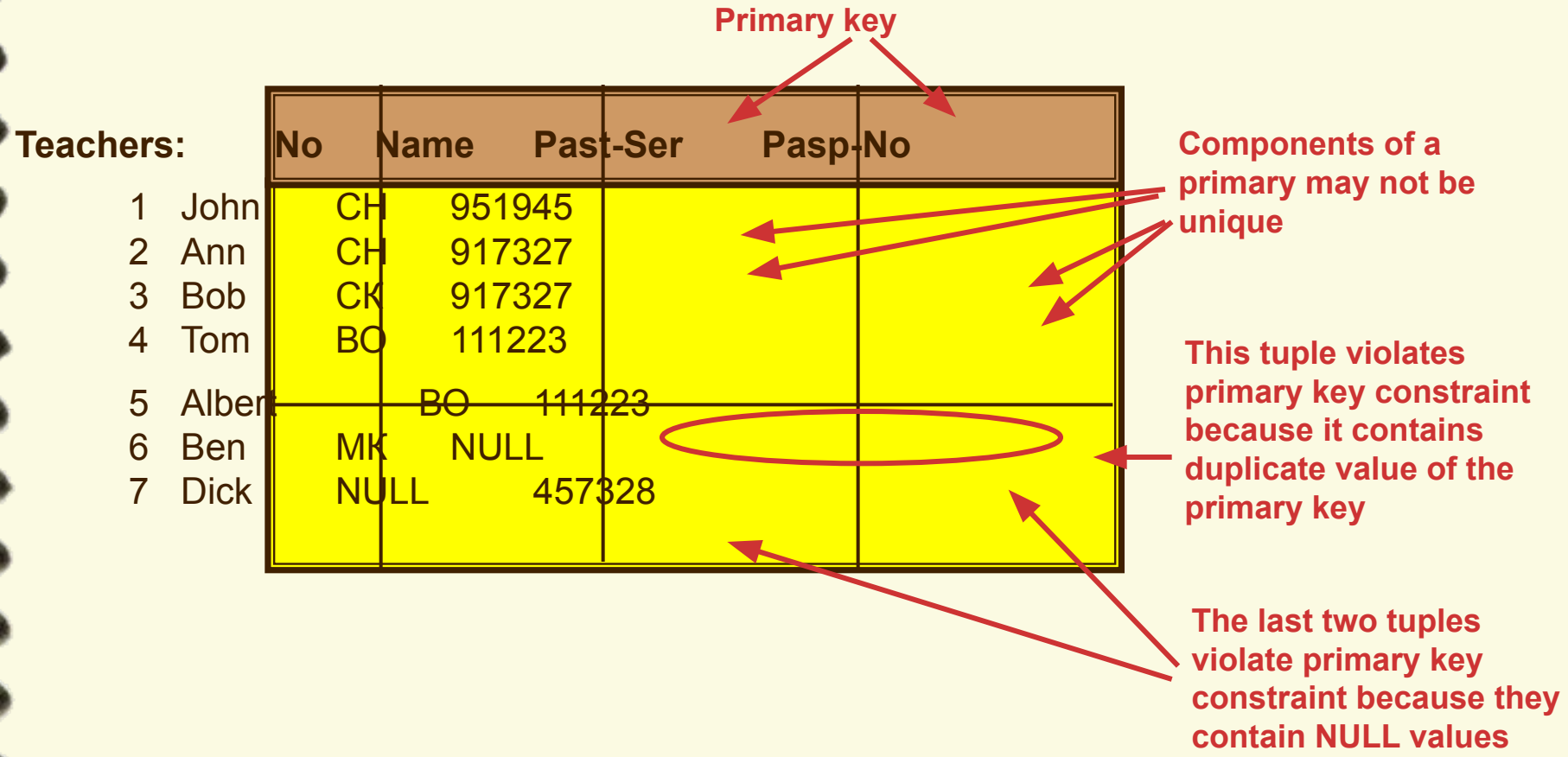
Main properties (integrity constraints):

- Primary key values must never be duplicated. That is they are unique within a relation. But there may be duplicates in parts of compound primary key.
- Primary key cannot have NULL values.

Additional properties:

- Every relation must have one and only one primary key.
- Primary key do not influence attribute order.
- Primary keys do not influence tuple order.

Example of a primary key



Foreign key

In a relational model relationships between relations are defined “by values”.

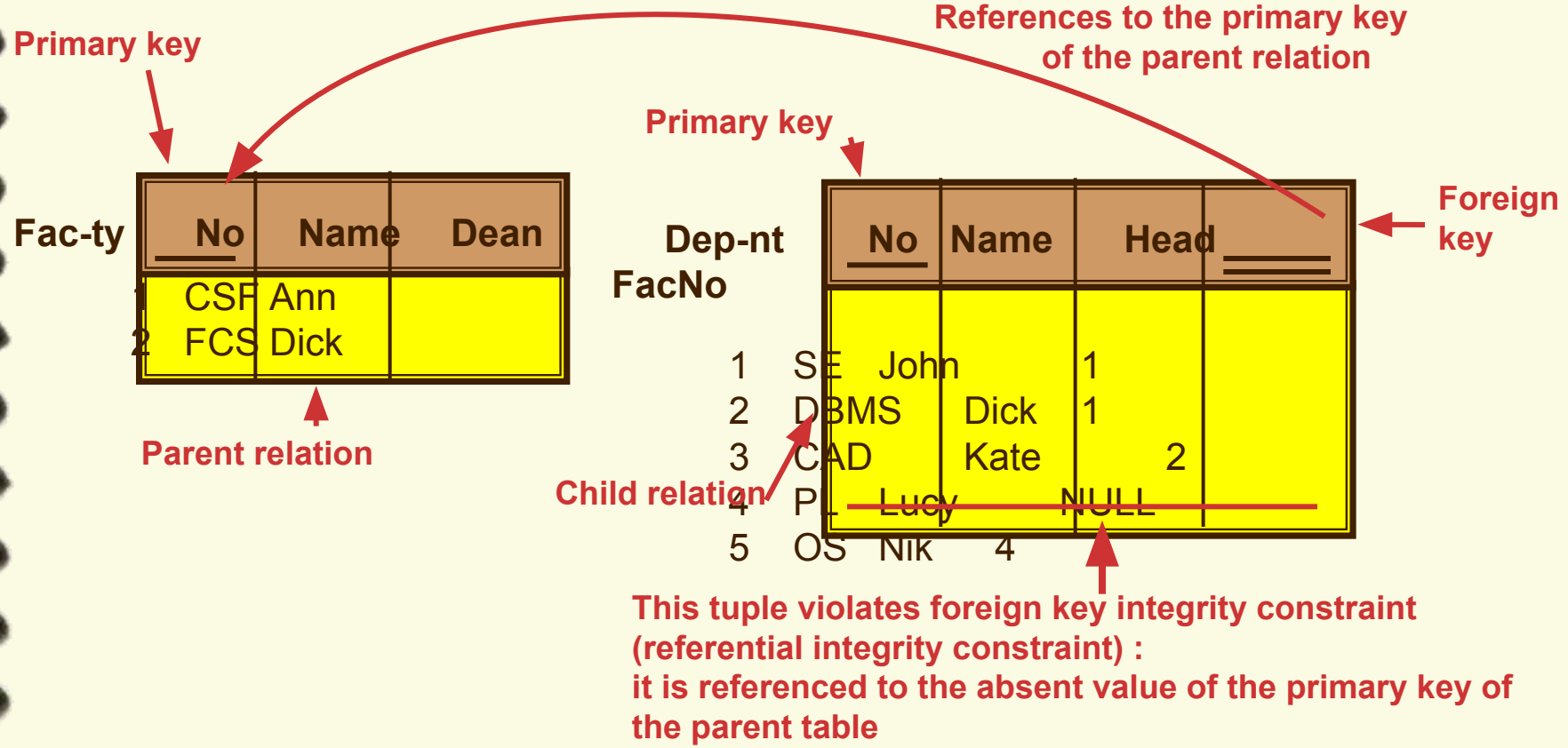
Foreign key is one or more attributes of a relation that are used to reference to tuples of other relation.

Relation that is references to other relation is called ***child relation***.

Relation that is referenced by other relation is called ***parent relation.***

Child relation may references only to primary key (or unique key) of the parent relation.

Example of a foreign key



Property of foreign key

Main property (integrity constraint):

- Value of a foreign key cannot reference to absent values of the primary key of the parent relation.
  It is so called ***referential*** integrity constraint of the foreign key.

Additional properties:

- Foreign key can contain duplicate values.
- Foreign key can contain NULL values.

Supporting referential constraint

Manipulating by tuples of a child relation:

- When a tuple is added or updated referential constraint is tested and if it is violated the corresponding action (adding/updating) is rejected.
- When a tuple is deleted referential constraint cannot be violated.

Manipulating by tuples of a parent relation :

- When a tuple is added referential constraint cannot be violated.
- When deleting of updating:
  - Tuple deleting/updating is not done if there are references from child relation, that violate referential integrity constraint.
  - Deleting/updating tuple of parent relation causes deleting/updating tuples of a child relation that references the tuple of the parent relation.
  - Deleting/updating tuple of parent relation causes setting NULL values to corresponding tuples of a child relation .
  - Deleting/updating tuple of parent relation causes setting DEFAULT values to corresponding tuples of a child relation.

Supporting referential integrity constraints in SQL

Supporting referential integrity constraints in standard SQL:

```
ON DELETE {RESTRICT | CASCADE | SET NULL | SET DEFAULT}
ON UPDATE {RESTRICT | CASCADE | SET NULL | SET DEFAULT}
```

Supporting referential integrity constraints in SQL Oracle:

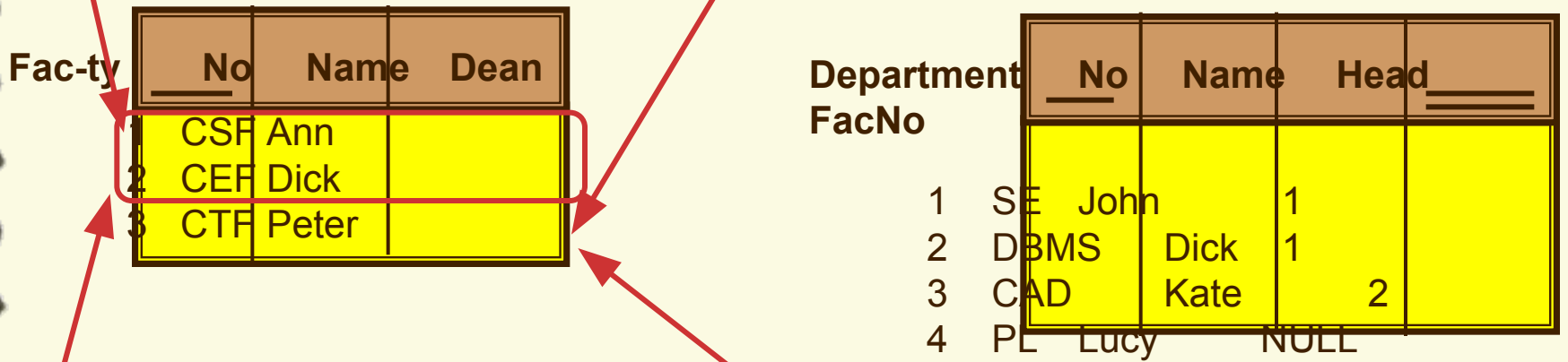
```
[ON DELETE {CASCADE | SET NULL}]
```

Supporting referential constraint – RESTRICT

ON DELETE RESTRICT

This tuple cannot be deleted because it is referenced by tuples of a child relation

This tuple may be deleted because it is not referenced



This tuple may be changed including "No" attribute

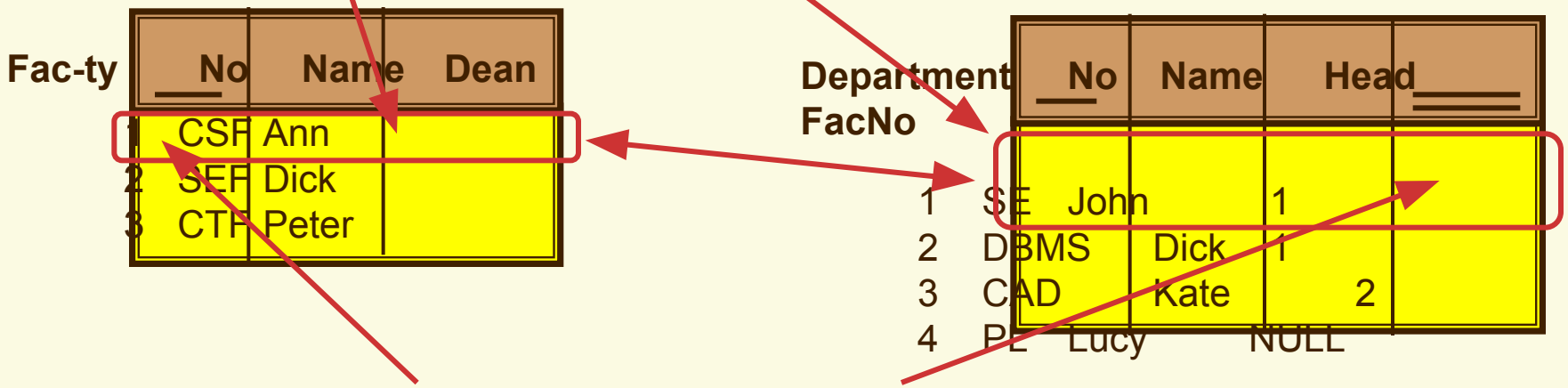
The "No" attribute of these tuples cannot be updated because they are referenced by foreign keys of a child relation

ON UPDATE RESTRICT

Supporting referential constraint – CASCADE

ON DELETE CASCADE

When deleting this tuple the following tuples of parent relation are also deleted.



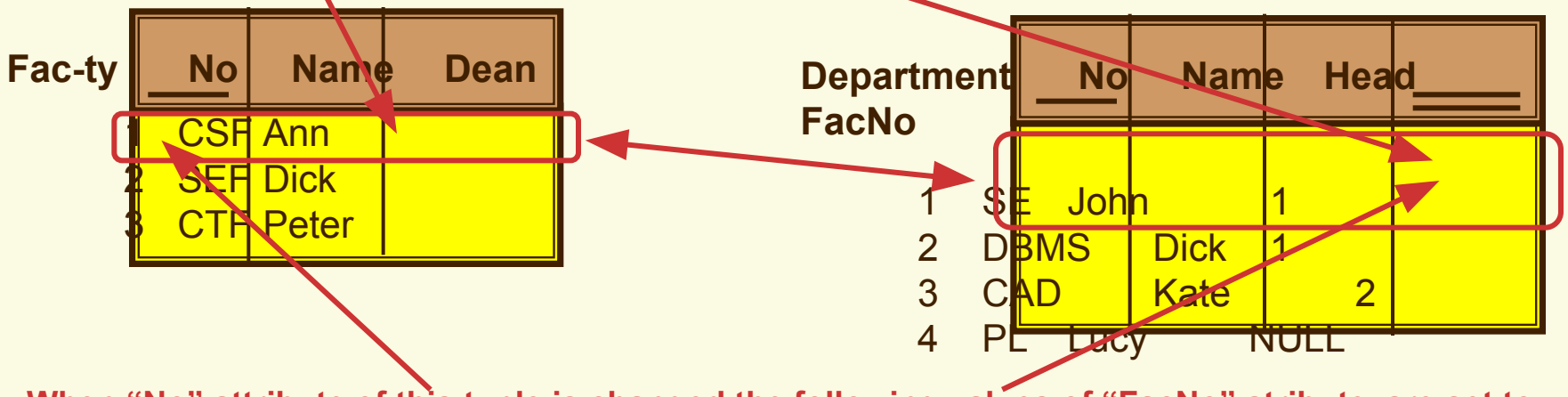
When "No" attribute of this tuple is changed the following values of "FacNo" atribute are also changed.

ON UPDATE CASCADE

Supporting referential constraint – SET NULL

ON DELETE SET NULL

On deleting this tuple the following values of "FacNo" attribute are set to NULL.



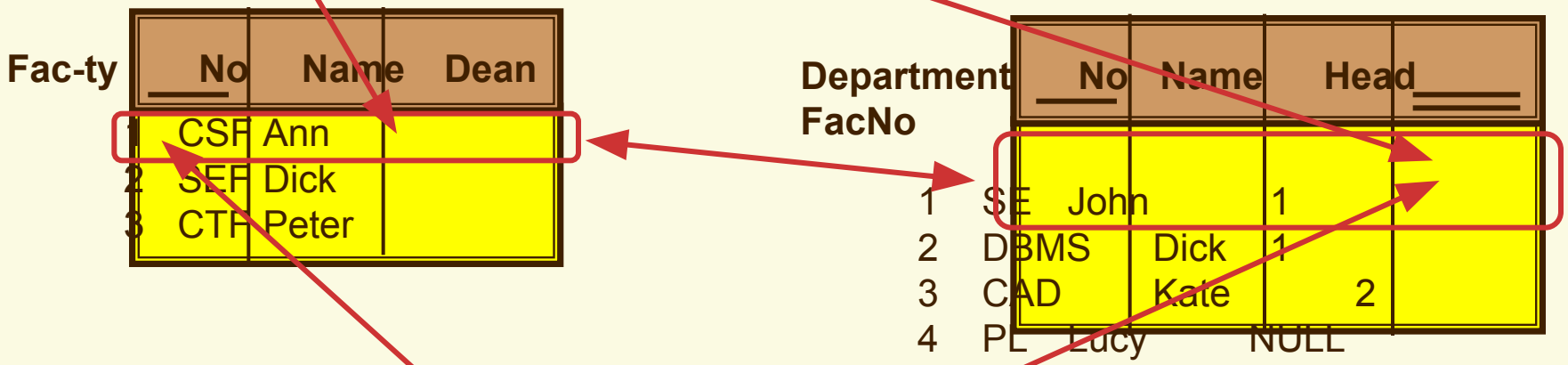
When "No" attribute of this tuple is changed the following values of "FacNo" atribute are set to NULL.

ON UPDATE SET NULL

Supporting referential constraint – SET DEFAULT

ON DELETE SET DEFAULT

On deleting this tuple the following values of "FacNo" attribute are set to default value.



When "No" attribute of this tuple is changed the following values of "FacNo" atribute are set to default value.

ON UPDATE SET DEFAULT

Recursive foreign key

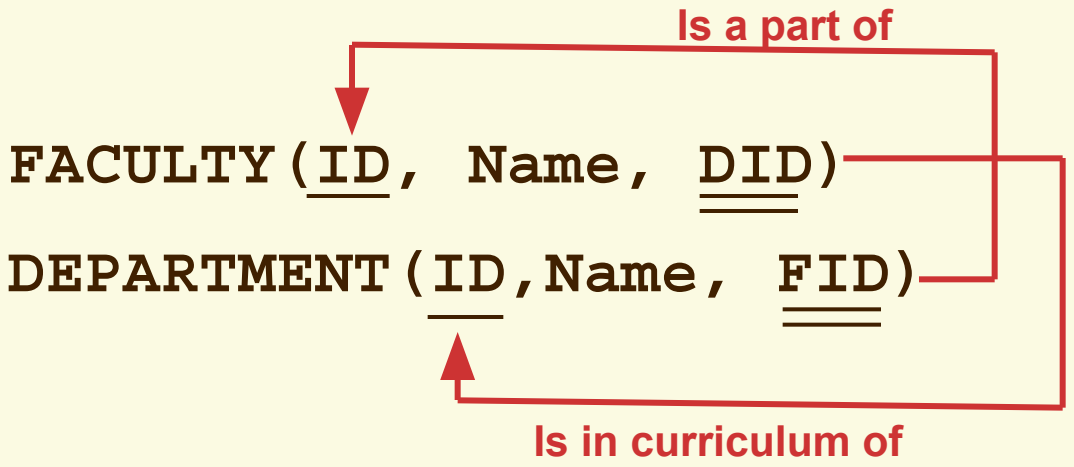
Foreign key is ***recursive*** if it references to the primary key of the relation where it is defined*.*

Teacher

No	Name	Pasp-Ser	Pasp-No	ChiefNo
1	Ann	CH	951945	4
2	Dick	CH	917327	4
3	Peter	CK	917327	7
4	John	BO	111223	7
5	Kate	BO	111224	4
6	Lucy	MK	NULL	7
7	Mary	NULL	457328	NULL

It allows to model hierarchy structure that is defined on one relation.

Cross-reference foreign keys



Foreign key as a part of primary key