Reznichenko Valery
Organization of data and knowledge bases

Lecture 8. Normalization theory of the relational model

National Aviation University
Computer Science Faculty
Department of Software Engineering

CONTENTS

What is the purpose of the normalization theory

The theory of relational model normalization establish :

- **how** initial relational schema may be transformed into other relational schema, which
- **equivalent** initial one in some sense and
- **Is better** it in any sense.

Thus this theory should answer to the following questions:

- What **criteria of equivalence** of relational schemas exist?;
- What **criteria of estimation** of relational schemas **quality** exist?
- What **techniques of equivalent transformations** of relational schemas exist?

Bad DB design (1)

Customers | 9 purchases

CN	NAME	CITY	...	PN1	DATE1	Q-TY1	PN2	DATE2	Q-TY2	...	PN9	DATE9	Q-TY9
1	Ivan	Kiev		1	21.01	20	2	23.01	17				
2	Peter	Odessa		1	26.10	25							
3	John	Kiev		2	29.01	20							

- We have set a limit of 9 purchases. What if a customer has more than 9 purchases?
- What to do if purchases are less 9? Set values to NULL?
  What to do if it is necessary to delete purchase in the middle of the list?
- What we have to do if it is necessary to order cusomers' orders.
- How does search condition look like? For example to find customer that buy parts with No 2?:

(PN1 = 2) OR (PN2 = 2) OR (PN3 = 2) ... OR (PN9 = 2)

Bad DB design (2)

CUSTOMER-PURCHASE

CN	NAME	CITY	…	PN	DATE	Q-TY
1	Ivan	Kiev		1	21.01	20
1	Peter	Kiev		2	23.01	17
2	Peter	Odessa		1	26.10	25
3	John	Kiev		2	29.01	20

- ***Insertion anomaly***: Data cannot be added because some other data is absent .
- *Update anomaly*: Data inconsistency or loss of data integrity can arise from data redundancy/repetition and partial update .
- ***Deletion anomaly***: Data maybe unintentionally lost through the deletion of other data .

Why! **It is possible when one relation contains information about two ore more entities of the application domain**

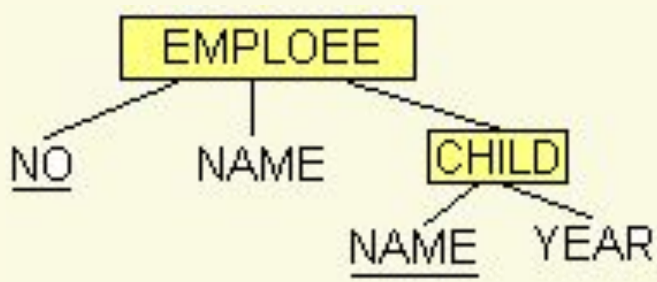
Normalization

Normalization is a step by step reversible process of equivalent transformation of one relational schema into other that has better characteristics. Every step of such transformation is called ***normal form.***

There are the following unwanted properties of relations and normal forms that remove corresponding properties:

- Compound (not atomic) values - 1NF
- Not full (partial) functional dependence - 2NF
- Transitive functional dependence - 3NF
- Multivalued dependence - 4NF
- Join dependence - 5NF

Compound domains and the First Normal Form (1NF)



Relation is in the ***first normal form*** (1NF) if all it's attributes are based on atomic (simple) domains and consequently the values in table cells are simple (atomic).
Relation is called ***normalized*** if it in first normal form .

EMPLOEE

NO	NAME	CHILD	
		NAME	YEAR
1	Walton	Ann	1987
		Adam	1979
2	Jerome	Bernd	1989
		Ann	1991
		Dan	1994
3	Wilde	Clara	1991

EMP-LOEE

NO	NAME
1	Walton
2	Jerome
3	Wilde

CHILD

NO	NAME	YEAR
1	Ann	1987
1	Adam	1979
2	Bernd	1989
2	Ann	1991
2	Dan	1994
3	Clara	1991

$R.A \rightarrow R.B \Leftrightarrow \forall r_1 \in R \ \forall r_2 \in R \ (r_1[A] = r_2[A] \supset r_1[B] = r_2[B])$

Functional dependencies (FD)

Let's relation R with attributes A and B is given. In relation R attribute B ***functionally depends*** on attribute A or A ***functionally determines*** B, if and only if every value of the projection R[A] is linked exactly to one value of the projection R[B]. Such functional dependence is denoted as $R.A \rightarrow R.B$.

The set of attributes A is called ***determinant*** for the set of attributes B.

Formally FD is defined in such a way :

$$R.A \rightarrow R.B \Leftrightarrow \forall r_1 \in R \ \forall r_2 \in R \ (r_1[A] = r_2[A] \supset r_1[B] = r_2[B])$$

The presence of FD is property of the relational schema, but not of instance of relational schema, and reflects semantics of the AD.

The set of FD can be viewed as a set of integrity constraints on the relation scheme; it should be preserved under decomposition.

Keys

Set of attributes K in relation R is called ***superkey*** of the relation R if each attribute of the relation R functionally depends on K.

Set of attributes K in relation R is ***candidate key*** of the relation R if :

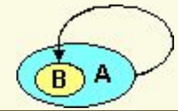
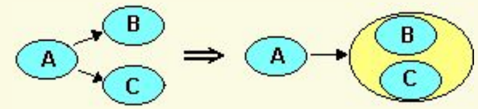
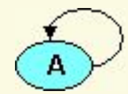
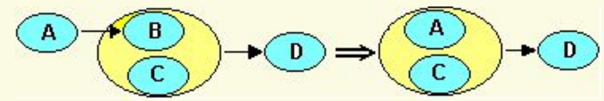
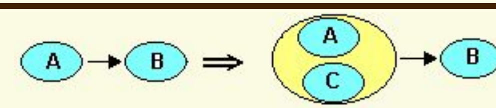
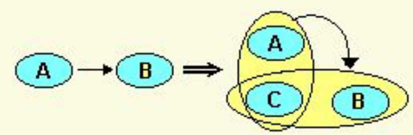
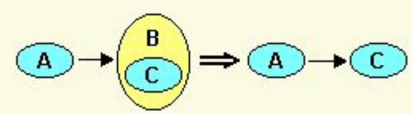
- each attribute of the relation R functionally depends on K;
- any attribute in K cannot be removed from K without violation of property (a).

It is possible to give the following formal definition of a candidate key. Let M - complete set of attributes of the relation R. The subset of attributes K of the relation R is candidate key, if

- $\forall A \subseteq M \quad R.K \rightarrow R.A$
- $\forall K' \subset K \quad \exists B \subseteq M \quad R.K' \not\rightarrow R.B$

Assertion: Any relation has candidate key.

Properties of functional dependencies

1)Transitivity : If A → B and B → C, then A → C.	
2) Projectivity: If B ⊆ A, then A → B	
3) Additivity: If A→ B and A→ C, then A→ (B,C)	
4) Reflexivity: A → A.	
5) Pseudotransitivity: If A→ B and (B,C)→ D, then (A,C) →D	
6) Continuation: If A → B, then (A, C) → B for any attribute C.	
7) Augmentation: If A → B, then (A, C) → (B, C) for any attribute C.	
8) Decomposition: If A → B and C ⊆ B, then A → C	

Properties 1), 2), 3) are *Armstrong* axioms

Armstrong's axioms are a *sound* and *complete* set of inference rules

Logical inference of functional dependencies

Let's R have set of functional dependencies F and the dependence A → C that is not in F. Dependence A → C is ***logically implied*** by (or ***logically deduced*** from) F if it may be inferred from F with the help of functional dependencies axioms.

For example, if we have the relation R(A, B, C) and F contains the dependence A → C, then the following dependences are logically implied from F:

(A, C) → B - continuation property is applied;
(A, C) → (B, C) - augmentation property is applied **.**

Closure, completeness, equivalence and minimal cover of FD

Let's relation R have set of functional dependencies F. The set of all functional dependencies logically implied by F is called ***(logical) closure*** of F. It is notified as F^+. It is obvious, that $F \subseteq F^+$ и $F^+ = F^{++}$.

Set of functional dependencies F is ***complete*** if $F = F^+$.

Two sets of dependencies F and G are ***(logically) equivalent*** if $F^+ = G^+$.

Lets given sets of functional dependencies F and G such that $G \subset F$. G is a ***cover*** of F if $G^+ = F^+$. If G is minimal then G is called ***basis of dependencies*** of F or ***minimal cover.***
NOTE: Minimal cover isn't necessarily unique.

FD и сущности предметной области

Thesis. If application domain contains functional dependence A → B there exists class of the entities that consist of attributes (A, B). More over in this class set of attributes A is an unique identifier of entities of this class (key) and B are properties of these entities .

If A → B1, A → B2, …, A → Bn, the exists class of entities with attributes(A, B1,…,Bn), where A – unique identifier and B1,…, Bn – are ordinary attributes.

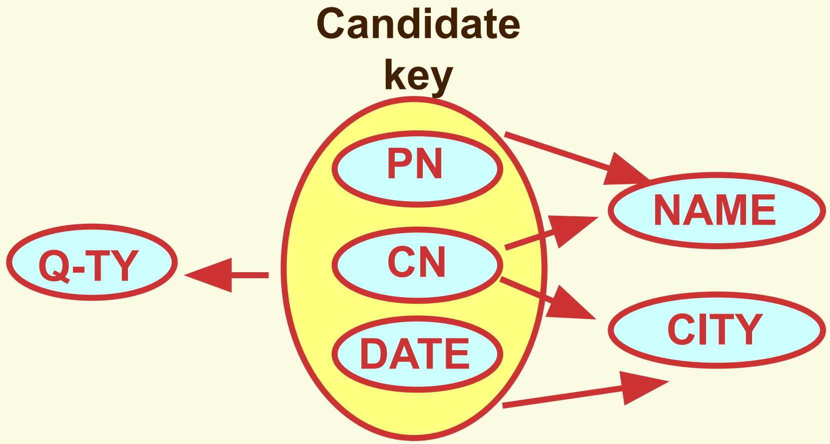
This thesis gives formal basis for identifying entities in AD.

Not full (partial) functional dependencies and second normal form (2NF)

Let's given relation with schema R(A, B, C). Functional dependence R.A → R.B is ***full*** if B does not depends functionally from any C ⊂ A that does not contained in B.

CUSTOMER-PURCHASE

CN	NAME	CITY	…	PN	DATE	Q-TY
1	Ivan	Kiev		1	21.01	20
1	Ivan	Kiev		2	23.01	17
2	Peter	Odessa		1	26.10	25
3	John	Kiev		2	29.01	20



- Attribute Q-TY depends fully from (PN, CN, DATE)
- Attributes NAME and CITY depends fully from CN
- Attributes NAME and CITY depends not fully (partially) from (PN, CN, DATE)

Anomalies of insertion, deleting and updating when not full FD exist

CUSTOMER-PURCHASE

CN	NAME	CITY	…	PN	DATE	Q-TY
1	Ivan	Kiev		1	21.01	20
1	Ivan	Kiev		2	23.01	17
2	Peter	Odessa		1	26.10	25
3	John	Kiev		2	29.01	20
4	Ann	Kherson		?	?	?

- **Update anomalies**. When changing customer city it is necessary to remember that information about customers may be duplicated.
- **Inset anomalies**. When it is necessary to enter information about new customer (Ann) we may do it only when it will do purchase.
- **Delete anomalies**. On deleting information about purchase of Ann we have to delete information about this customer.

The second normal form (2NF)

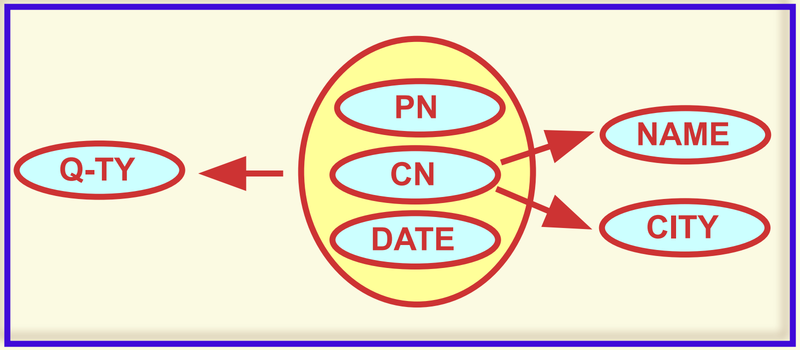
Relation is in the ***second normal form*** (2NF) if it is in the first normal form and all its nonprimary attributes are depend fully from candidate key .

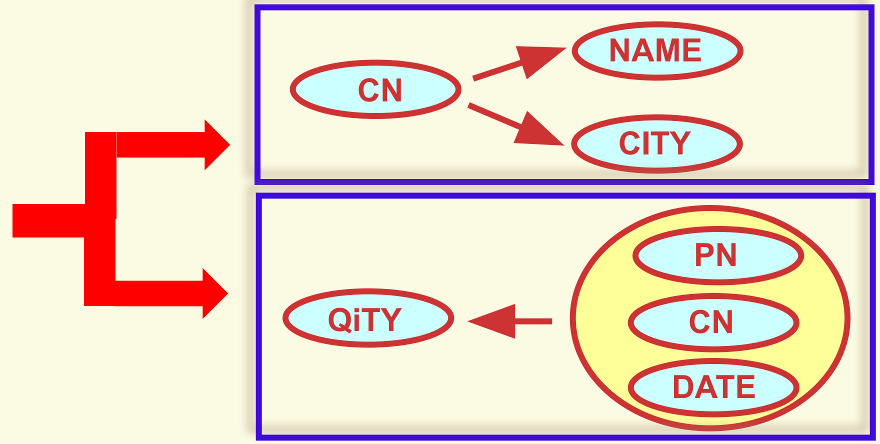
Теорема (Хита). The relation R with attributes A, B, C , where R.A → R.B, is equal to natural join of the projections R[A, B] and R[A, C].

Such splitting is called ***binary decomposition***.

Algorithm of reduction to 2NF. Given relation R with set of attributes M. If in R there is not full functional dependence R.A → R.B of non primary attribute B from a candidate key A, the relation R is decomposed into the following two relations: R [A, B] and R [M - B]. If the resulting relations are still not in the second normal form, the mentioned algorithm is applied to these relations again.

Example of reduction to the 2NF





CUSTOMER-PURCHASE

CN	NAME	CITY	...	PN	DATE	Q-TY
1	Ivan	Kiev		1	21.01	20
1	Ivan	Kiev		2	23.01	17
2	Peter	Odessa		1	26.10	25
3	John	Kiev		2	29.01	20

CUSTOMER

CN	NAME	CITY	...
1	Ivan	Kiev	
2	Peter	Odessa	
3	John	Kiev	

PURCHASE

CN	PN	DATE	Q-TY
1	1	21.01	20
1	2	23.01	17
2	1	26.10	25
3	2	29.01	20

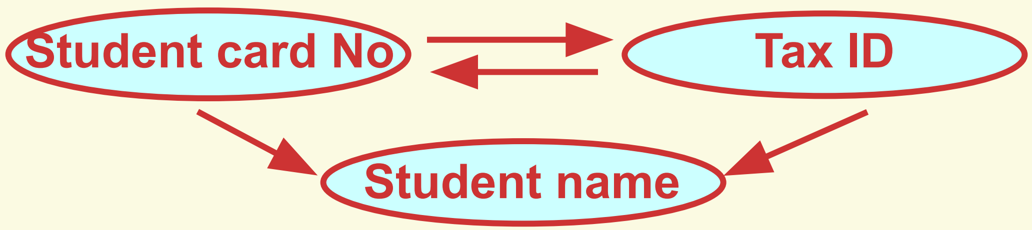
Example of reduction to the 2NF - Summary

- Source relation contains information from 2 entities, every resulting relations contain information about one entity each.
- Resulting relations do not contain anomalies of deletion, inserting and updating.
- Source relation can be restored from resulting relations with the help of natural join.
- Such decomposition do not lose functional dependencies. They may be restored from decomposed relations.

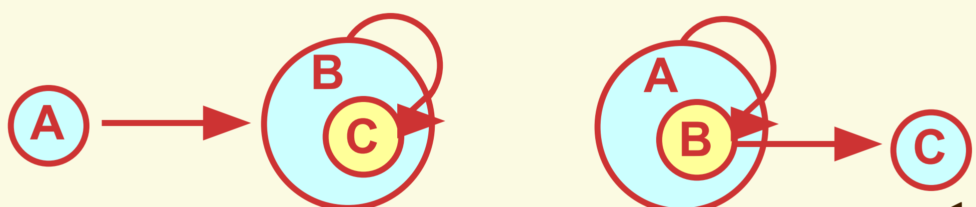
Transitive dependencies and the Third Normal Form (3NF)

Relation R has ***transitive dependence*** if it has such attributes (set of attributes) A, B, C, C ⊄ B, B ⊄ A that:
$A \to B$, $B \to C$; at the same time $B \nrightarrow A$

1) Condition $B \nrightarrow A$ is necessary in order to exclude trivial transitive dependence like this:



2) Conditions C ⊄ B, B ⊄ A are necessary to exclude the following trivial transitive dependencies:



Anomalies of insertion, deleting and updating when transitive FD exist

DEPARTMENT-FACULTY

DepNo	DName	DHead	FacNo	FName	FDean
1	DBMS	John	1	CSF	Ann
2	CAD	Peter	1	CSF	Ann
3	SE	Sam	1	CSF	Ann
4	CAM	Dick	2	CTF	Dave
5	OS	PL	2	CTF	Dave

DEPARTMENT entities **FACULTY entities**

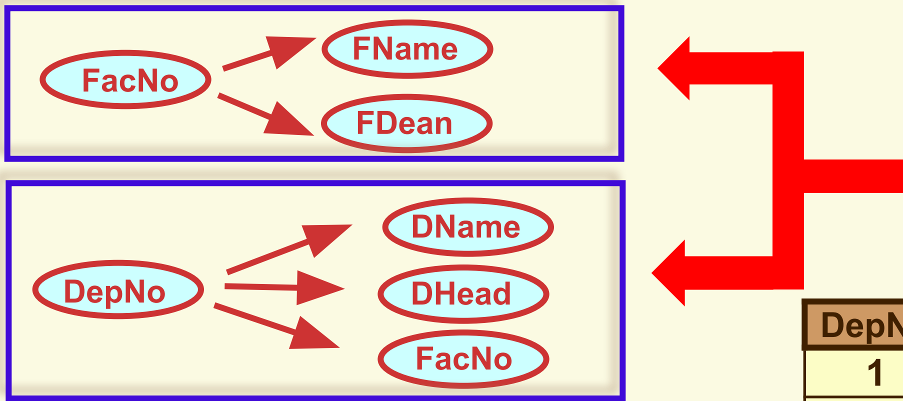
- Availability transitive dependencies in a relation means the relation contains information from more that one entity.
- As a result such relations imply anomalies of insertion, deletion, updating.

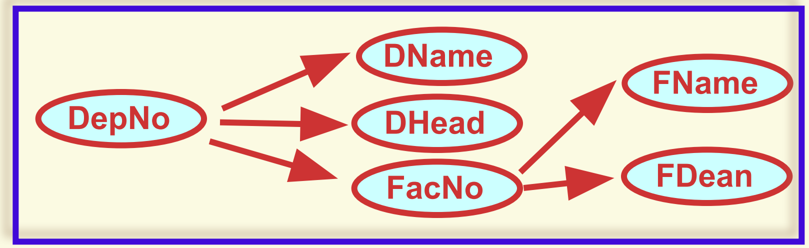
The Third Normal Form (3NF)

The relation is in the third normal form (3NF) if it is in the second normal form and does not contain transitive dependencies of nonprimary attributes from candidate keys .
Other words all nonprimary attributes must functionally depend ONLY from candidate keys.

Algorithm of the relation reduction to 3NF. Let's given the relation R with attributes A, B, C and there are functional dependencies R.A → R.B and R.B → R.C. The relation R decomposed into following two relations: R[A, B] and R[B, C]. If the resulting relations are still not in the third normal form, the mentioned algorithm is applied to these relations again.

Example of reduction to the 3NF





DepNo	DName	DHead	FacNo	FName	FDean
1	DBMS	John	1	CSF	Ann
2	CAD	Peter	1	CSF	Ann
3	SE	Sam	1	CSF	Ann
4	CAM	Dick	2	CTF	Dave
5	OS	PL	2	CTF	Dave

DEPARTMENT

DepNO	DName	DHead	FacNo
1	DBMS	John	1
2	CAD	Peter	1
3	SE	Sam	1
4	CAM	Dick	2
5	OS	PL	2

FACULTY

FacNo	FName	FDean
1	CSF	Ann
2	CTF	Dave

Example of reduction to the 3NF - Summary

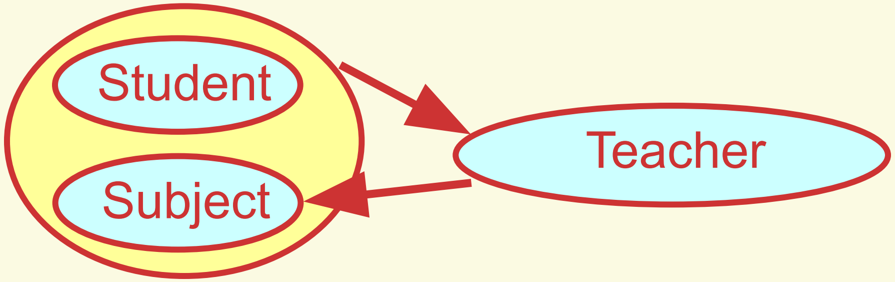
Results the same as in reduction to the 2NF:

- Source relation contains information from 2 entities, every resulting relations contain information about one entity each.
- Resulting relations do not contain anomalies of deletion, inserting and updating.
- Source relation can be restored from resulting relations with the help of natural join.
- Such decomposition do not lose functional dependencies. They may be restored from decomposed relations.

Strong 3NF (S3NF)

Note that 3NF requires absence of transitive dependence of *nonprimary attributes* but not all attributes of the relation . Strong 3NF requires absence of transitive dependence of ALL attributes of a relation

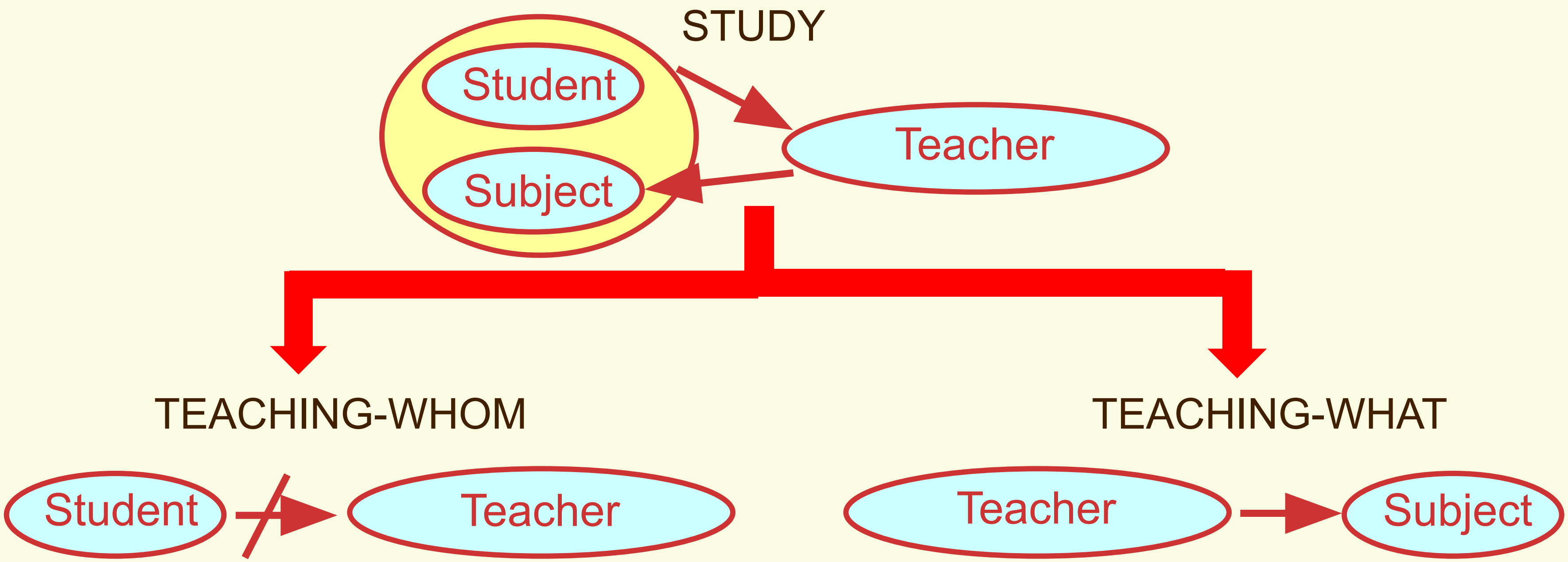
This relation is in the 3NF, but still contains information about two entities. So it hold anomalies.



Relation is in ***strong 3NF***, if it is in 3NF and does not contain transitive dependencies of ALL attributes from candidate keys .

Reduction to the S3NF

Algorithm of reduction to the S3NF is the same as for 3NF



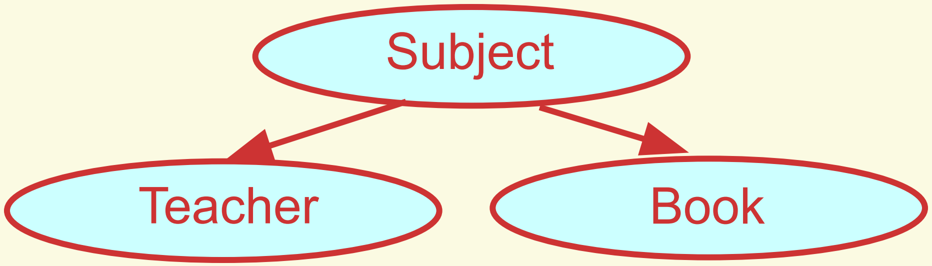
NOTE. One of the functional dependence is lost!!!

Boyce-Codd normal form (BCNF)

Relation R is in ***Boyce-Codd normal***, if every its determinants is a superkey.

Assertion. S3NF and BCNF are equivalent

Multivalued dependencies and the Fourth Normal Form (4NF)



TEACHING

ПРЕДМЕТ	ПРЕПОД	УЧЕБНИК
DB&KB	Smith	DB Foundations
DB&KB	Smith	Introduction to DB
DB&KB	Smith	DB Theory
DB&KB	John	DB Foundations
DB&KB	John	Introduction to DB
DB&KB	John	DB Theory

Thesis: If in an application domain there is no direct relationship between attributes A and B, and it is necessary to fix such relationship in one relation, the only correct decision is to determine, that all values of attribute A are related to all values of attribute B, and vise versa.

Definition of the multivalued dependency (MVD)

Given relation R with attributes (set of attributes) A, B, C. There exists ***multivalued dependency*** B of A (or A determines B multivalued), denoted as A →→ B, if for given set of values attributes from A there exist set of related values attributes of B and this set of B-values does not depends from values of attributes C.

Example: In the relation TEACHING the are the following MVD:

Subject →→ Teacher
Subject →→ Bood

Let's there is relation R(A,B). The MVDs A →→ ∅ and A →→ B are called ***trivial*** because they exist in any relations.

MVD axioms

Given relation R with attributes (set of attributes) A, B, C.

Multivalued dependences have the following ***axioms*** :

1) Complementation axiom

If A →→ B, then A →→ C

2) Augmentation axiom

If A →→ B and V ⊆ W, then (A, W) →→ (B, V)

3) Transitivity axiom

If A →→ B and B →→ C, then A →→ C – B

Axioms that relates FD и MVD

The following two axioms relates functional and multivalued dependencies .

1) Replication axiom

If A → B, then A →→ B

2) Coalescence axiom

If A →→ B and Z ⊆ B, and for some W, that is not intersect with B we have W → Z, then A → Z

Some additional properties of MVD

1) Union

If A $\rightarrow\rightarrow$ B and A $\rightarrow\rightarrow$ C, then A $\rightarrow\rightarrow$ (B, C)

2) Pseudo-transitivity

If A $\rightarrow\rightarrow$ B and (W, B) $\rightarrow\rightarrow$ Z ,
then (W, A) $\rightarrow\rightarrow$ Z – (W, B)

3) Mixed pseudo-tranisitivity

If A $\rightarrow\rightarrow$ B and (A,B) $\rightarrow\rightarrow$ C, then A $\rightarrow\rightarrow$ (C - B)

4) Intersection and difference

If A $\rightarrow\rightarrow$ B and A $\rightarrow\rightarrow$ C,
then A $\rightarrow\rightarrow$ B $\cap$ C, A $\rightarrow\rightarrow$ B – C, A $\rightarrow\rightarrow$ C – B

The fourth nornal form (4NF)

The relation R is in ***fourth normal form*** (4NF), if from existence of nontrivial multivalued dependence X →→ Y (where Y ⊄ X) it is follows that X is a superkey of the relation R.

Assertion. Lets relation R consists of attributes (set of attributes) A, B, C. Dependence A →→ B exist in R if and only if R = R[A, B] * R[A, C].

Reduction to the 4NF and embedded MVD

Algorithm reduction to the 4NF. Lets given relation R with attributes (set of attributes) A, B, C, and given multivalued dependence R.A →→ R.B. Relation R decomposed into the following two relations: R[A, B] и R[B, C].

If resulting relations are not in 4NF the algorithm is applied once more to these relations .

Multivalued dependency is ***embeded*** if it is absent in the relation but exists in the projection of the relation by some attributes.

$R=\pi_{A_1}(R)*\pi_{A_2}(R)*...*\pi_{A_n}(R)$

Join dependency (JD) and the Fifth Normal Form (5NF)

Lets R is a relation with attributes (set of attributes) A1, A2,,.., An. Relation R have ***join dependency*** with respect of A1, A2,,.., An, that is denoted as *(A1, A2,…, An), if relation R is equal to natural joins of all of its projections over A1, A2,…, An:

$$R = \pi_{A1}(R)*\pi_{A2}(R)*\ldots*\pi_{An}(R) \Leftrightarrow R = R[A1] * R[A2]*\ldots*R[An]$$

JD is ***trivial*** if one of Ai is equal to the list of all attributes of the relation R.

A join dependency is ***implied by the candidate keys of R*** if each of Ai $(1 \le i \le n)$ are superkeys of R.

Relationships between JD and MVD

Every JD of the form *(A, B) in relation with schema R(A,B), where A and B - set of attributes, is equivalent to the MVDs $A \cap B \rightarrow\rightarrow A$ and $A \cap B \rightarrow\rightarrow B$. (Any MVD is JD, but not wise versa!!!)

But there exist JD that are not equivalent any MVD. An example of such JD in relation R(A, B, C) is the dependency *((A,B), (B, C), (A,C)). It is not equivalent to any MVD. Example:

A	B	C
a1	b1	c2
a2	b1	c1
a1	b2	c1
a1	b1	c1

In the example to the left relation contains JD *((A,B), (B, C), (A,C)). It may be verified by calculating:

$\pi_{A1}(R) * \pi_{A2}(R) * ... * \pi_{An}(R)$.

But it does not contain any nontrivial MVD.

It may be convinced by testing that no one of the following multi-valued dependency exist:

$A \rightarrow\rightarrow B$, $A \rightarrow\rightarrow C$, $B \rightarrow\rightarrow A$, $B \rightarrow\rightarrow C$, $C \rightarrow\rightarrow A$, $C \rightarrow\rightarrow B$.

The Fifth Normal Form - 5NF

Relation R is in ***the fifth normal form*** (5NF) if and only if for all of its nontrivial JD *(A1, A2,…, An) all sets of attributes Ai are superkeys of R.

This normal form is also called ***project-join normal form*** (**PJNF**).

Assertion. Because of any multivalued dependency is also join dependency, any relation in PJNF (5NF) is also in 4NF .

Classic example to motivate 5NF involves a join n-way decomposition that cannot be derived by a sequence of 2-way decompositions

Example of the relation in the 5NF

Let's the is a relation TBS(TCH, BOK, SBJ), where we record information aboutsuch ssues :

- what teaches what books are used;
- what books in what subjects are used;
- what subjects by what teachers are taught.

The fact that the relation contains the following information:

- *Reznichenko uses in his lectures the book «SQL language»*,
- *The book «SQL language» is used in the subject DB&KB»* and
- *Reznihcenko has lectures by subject DB&KB.*

Does not mean that «Reznichenko uses the book "SQL Language" in his lectures by subject DB&KB»

Relation TBS is in 5NF because it do not have JDs.

Example of the relation that violates 5NF, and reduction it to the 5NF

If relation TBS has additional rule (as a business rule of the application domain):

«From the facts:

- teacher **t** uses in his lectures book **b,**
- book **b** is used in subject **s** and
- teacher **t** has lectures by subject **s**

follows that teacher **t** uses book **b** in lectures by subject **s»,**

then relation TBS has JD *((TCH, BOK), (BOK, SBJ), (TCH, SBJ)) and this relation is not in 5NF because it has the only candidate key that coincide with all attributes of the relation, that is (TCH, BOK, SUBJ).

In this case relation TBS reduces to 5NF in such a way:

TBS(TCH, BOK, SBJ) ⇒TB(TCH,BOK), BS(BOK, SBJ), TS(TCH, SBJ)

Example of the relation that violates 4NF, and reduction it to the 4NF

If relation TBS has additional rule (as a business rule of the application domain):

«From the facts**:**

- teacher **t** uses in his lectures book **b,**

- teacher **t** has lectures by subject **s**

follows that teacher **t** uses book **b** in lectures by subject **s»,**

Then relation TBS has JD *((TCH, BOK), (TCH, SBJ)) or it is the same as the relation has the following MVDs TCH →→ BOK, TCH →→ SBJ, and this relation neither in 5NF nor in 4NF.

In this case TBS reduces to the 4NF (and more over to the 5NF) in such a way:

TBS(TCH, BOK, SBJ) ⇒TB(TCH,BOK), TS(TCH, SBJ)

Design of relational model schema

- Formal description of the relational schema design task
- Decomposition of the relational schema
- Equivalence of relations
  - Loosless decomposition with data preservation
  - Loosless decomposition with dependencies preservation
  - Equivalence of the normal forms
- Criteria of a relation qualities

Formal definition of the relational schema design task

Thesis of universal relation. All application domain may be represented as one universal relation that contains all attributes of the domain.

Formal definition of the design task. Lets given relational schema S0, that contains schema of the only (universal) relation R:

S_0 = {R = <U, G>},

where U – set of attributes, a G – set of dependencies, It is necessary to find equivalent relational schema SD, represented as the set of relations R_1,…, R_n:

S_D = {R_i = <U_i, G_i>, i = 1, 2, ..., n},

that should be better in any sense that schema S_0.

In this definition it is necessary to clarify the following items:

- what procedure must be used to convert one set of relations into another;
- what does equivalence of the two schemas mean ;
- how can we estimate that one relational schema is better than another .

Decomposition of the relational schema

Decomposition relation R(M) with set of attributes M into the set of relations R1, R2,…, Rn with attributes M1, MA2,…, Mn is a procedure that satisfy the following conditions:

- $M1 \cup M2 \cup \ldots \cup Mn = M$. That is any attribute of R belongs at least one of relation Ri and all attributes of Ri should be defined in R .
- All relations Ri, i = 1, 2,..., n, are projections of the relation R over attributes that are contained in the Ri, that is

$$Ri(Mi) = \pi_{Mi}(R)$$

It is said that decomposition has the property of ***looseless join***, if R is a natural join of the relations R1, R2,…, Rn., that is R = R1 * R2 *…* Rn

Decomposition is the only operation that is used while splitting relational schemas

Equivalence of relational schemas by dependencies

Equivalence by dependencies. Two sets of the relations are equivalent by dependencies, if they defined on the same set of attributes and they have the same set of dependencies (functional and multivalued).

Formally, lets given two schemas S_0 and S_D, that was defined previously. They are equivalent by dependencies if:

$$U = \bigcup_{i=1}^{n} U_i \quad \text{и} \quad G^+ = \left(\bigcup_{i=1}^{n} G_i \right)^+$$

Where U, Ui are attributes of the schemas S_0 and S_D: and G, Gi are dependencies of S_0 and S_D.

Equivalence of relational schemas by data

Equivalence by data. Two sets of relations equivalent by data if natural join of both set of relations gives the identical relations.

If source and resulting schemas are S_0 and S_D, equivalence by data means that splitting of the relation is done by using loosless decomposition.

How does loosless decomposition may be achived?

Assretion. If R1(U1) R2(U2) are decomposition of R(U) that preserve functional and/or multivalued dependencies, then this decomposition provides lossless join if and only if:

$$U1 \cap U2 \rightarrow (\text{or} \rightarrow\rightarrow)\ U1 - U2$$

OR

$$U1 \cap U2 \rightarrow (\text{or} \rightarrow\rightarrow)\ U2 - U1$$

Equivalence of the normal forms

Property of loosless join not always guarantee dependency preservation.

At the same time relation splitting that provides dependency preservation not always guantee the property of loosless join.

Equivalence of the normal forms.

Decomposition of the universal relation up to the 3NF preserve equivalence by data and dependencies.

Converting universal relation to the BCNF preserve equivalence by data but not preserve equivalence by dependencies.

Criteria of the relational schema quality

Let us consider how can we estimate the schema quality: that is criteria that one schema is better than other.

One schema is better that other if it does not have data manipulation anomalies .

Actually quality of the schema may be estimated by normal forms. The higher form is used the more qualified schema is received .