Functional Dependencies and Normalization

Aims:

At the end of this group of four lectures, you should be able to understand functional dependencies and normal forms, as well as to be able to normalize relations.

Reading:

Elmasri & Navathe, Chapters 15 & 16 (6th ed.)
or Chapters 14 & 15 (7th ed.)

Slide 7-1

Overview

1. Informal Design Guidelines
2. Decomposition
3. Functional Dependencies
4. Normal Forms Based on Primary Keys
5. General Normal Form Definitions
6. Boyce-Codd Normal Form
7. Relational Synthesis

Slide 7-2

Informal Design Guidelines

- What is relational database design?
  The grouping of attributes to form "good" relation schemas
- Two levels of relation schemas
  - The logical "user view" level:
    - How the user interprets the schema?
    - Is it clear and easy to understand?
  - The storage "base relation" level
    - How are the tuples from a relation stored?
    - Is the storage space needed minimal?
- Design is concerned mainly with base relations
- What are the criteria for "good" base relations?

Slide 7-3

Informal Design Guidelines

1. Semantics of the Relation Attributes
2. Redundant Information and Update Anomalies
3. Null Values in Tuples
4. Spurious Tuples

Slide 7-4

Semantics of Attributes

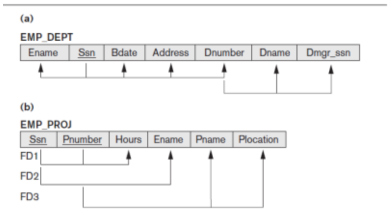
GUIDELINE 1: Informally, each tuple in a relation should represent one entity or relationship instance.

- Attributes of different entities should not be mixed in the same relation
- Only foreign keys should be used to refer to other entities
- Entity and relationship attributes should be kept apart as much as possible.

Slide 7-5

Examples (Fig 14.1, 14.3)

EMPLOYEE (EName, <u>SSN</u>, BDate, Address, DNumber)
DEPARTMENT (DName, <u>DNumber</u>, DMgrSsn)
DEPT_LOCATIONS (<u>DNumber, DLocation</u>)
PROJECT (PName, <u>PNumber</u>, PLocation, DNum)
WORKS_ON (<u>SSN, PNumber</u>, Hours)



Slide 7-6

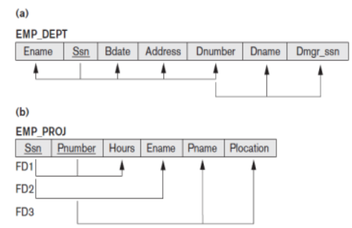
Redundant Information in Tuples and Update Anomalies

- Mixing attributes of multiple entities may cause problems
- Information is stored redundantly wasting storage
- Problems with update anomalies
  - Insertion anomalies
  - Deletion anomalies
  - Modification anomalies

Slide 7-7

Examples of Update Anomalies

Figure 15.3
Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.



Slide 7-8

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Redundancy (Dname, Dmgr_ssn)

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

Redundancy (Ename) Redundancy (Pname, Plocation)

Figure 15.4
Sample states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 15.2. These may be stored as base relations for performance reasons.

Addison-Wesley is an imprint of PEARSON

Guideline 2

- Design relations so that no update anomalies are present
- If any anomalies are present:
  - Note them clearly
  - Make sure that the programs that update the database will operate correctly

Null Values in Tuples

GUIDELINE 3: Relations should be designed such that their tuples will have as few NULL values as possible

- Attributes that are NULL frequently could be placed in separate relations (with the primary key)
- Reasons for nulls:
  - attribute not applicable or invalid
  - attribute value unknown (may exist)
  - value known to exist, but unavailable

Slide 7-11

Spurious Tuples

- Bad designs for a relational database may result in erroneous results for certain JOIN operations
- The "lossless join" property is used to guarantee meaningful results for join operations

GUIDELINE 4: The relations should be designed to satisfy the lossless join condition. No spurious tuples should be generated by doing a natural-join of any relations.

Example: Figures 14.5-6

Slide 7-12

(a)

EMP_LOCS

Ename	Plocation

P.K.

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation

P.K.

(b)

EMP_LOCS

Ename	Plocation
Smith, John B.	Bellaire
Smith, John B.	Sugarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Sugarland
Wong, Franklin T.	Sugarland
Wong, Franklin T.	Houston
Wong, Franklin T.	Stafford
Zelaya, Alicia J.	Stafford
Jabbar, Ahmad V.	Stafford
Wallace, Jennifer S.	Stafford
Wallace, Jennifer S.	Houston
Borg, James E.	Houston

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation
123456789	1	32.5	ProductX	Bellaire
123456789	2	7.5	ProductY	Sugarland
666884444	3	40.0	ProductZ	Houston
453453453	1	20.0	ProductX	Bellaire
453453453	2	20.0	ProductY	Sugarland
333445555	2	10.0	ProductY	Sugarland
333445555	3	10.0	ProductZ	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston
999887777	30	30.0	Newbenefits	Stafford
999887777	10	10.0	Computerization	Stafford
987987987	10	35.0	Computerization	Stafford
987987987	30	5.0	Newbenefits	Stafford
987654321	30	20.0	Newbenefits	Stafford
987654321	20	15.0	Reorganization	Houston
888665555	20	NULL	Reorganization	Houston

Figure 15.5
Particularly poor design for the EMP_PROJ relation in Figure 15.3(b). (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 15.4 onto the relations EMP_LOCS and EMP_PROJ1.

Addison-Wesley is an imprint of PEARSON

	Ssn	Pnumber	Hours	Pname	Plocation	Ename
	123456789	1	32.5	ProductX	Bellaire	Smith, John B.
*	123456789	1	32.5	ProductX	Bellaire	English, Joyce A.
	123456789	2	7.5	ProductY	Sugarland	Smith, John B.
*	123456789	2	7.5	ProductY	Sugarland	English, Joyce A.
*	123456789	2	7.5	ProductY	Sugarland	Wong, Franklin T.
	666884444	3	40.0	ProductZ	Houston	Narayan, Ramesh K.
*	666884444	3	40.0	ProductZ	Houston	Wong, Franklin T.
*	453453453	1	20.0	ProductX	Bellaire	Smith, John B.
	453453453	1	20.0	ProductX	Bellaire	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Smith, John B.
	453453453	2	20.0	ProductY	Sugarland	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	2	10.0	ProductY	Sugarland	Smith, John B.
*	333445555	2	10.0	ProductY	Sugarland	English, Joyce A.
	333445555	2	10.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	3	10.0	ProductZ	Houston	Narayan, Ramesh K.
	333445555	3	10.0	ProductZ	Houston	Wong, Franklin T.
	333445555	10	10.0	Computerization	Stafford	Wong, Franklin T.
*	333445555	20	10.0	Reorganization	Houston	Narayan, Ramesh K.
	333445555	20	10.0	Reorganization	Houston	Wong, Franklin T.

⋮

Figure 15.6
Result of applying NATURAL JOIN to the tuples above the dashed lines in EMP_PROJ1 and EMP_LOCS of Figure 15.5. Generated spurious tuples are marked by asterisks.

Addison-Wesley is an imprint of PEARSON

DECOMPOSITION

- Update anomalies can be solved by decomposing relations
- Decomposition is the process of breaking up a large relation into a set of smaller ones
- When is it necessary to decompose a relation?
- If several decompositions are possible, which one should we use?

NORMALIZATION

- A technique for producing a set of relations with desirable properties
- A list of tests to apply on a relational schema in order to determine its quality
- Introduced by Codd (1972)

FUNCTIONAL DEPENDENCIES

- A functional dependency is a constraint between two sets of attributes in the database.
- Universal relation schema R(A_1, A_2, A_3, ..., A_n)
- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y
- X → Y
- FDs are derived from the *meaning* and *interrelationships* of the attributes

FUNCTIONAL DEPENDENCIES

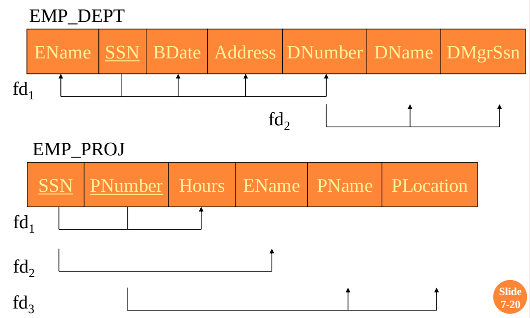
- X → Y
- X functionally determines Y
- Y is functionally dependent on X
- X: the left-hand side of the FD
- Y: the right-hand side of the FD
- can be displayed graphically on a relation schema (denoted by the arrow)
- Notation: XZ stands for X U Z

FUNCTIONAL DEPENDENCIES (CONT)

- X → Y holds if whenever two tuples have the same value for X, they *must have* the same value for Y
- For any two tuples t1 and t2 in any relation instance r(R): *If* $t_1[X]=t_2[X]$, *then* $t_1[Y]=t_2[Y]$
- X → Y in R specifies a *constraint* on all relation instances r(R)
- FDs are derived from the real-world constraints on the attributes

Slide 7-19

EXAMPLES



Slide 7-20

FUNCTIONAL DEPENDENCIES (CONT)

- A FD is a property of the attributes in the schema
- The constraint must hold on *every relation instance* r(R)
- If K is a key of R, then K functionally determines all attributes in R (since we never have two distinct tuples with $t_1[K]=t_2[K]$)

Slide 7-21

EXAMPLE 1

TEACHER	COURSE	TEXT
Smith	Data Structures	Bartram
Smith	Databases	Elmasri
Hall	Compilers	Hoffman
Brown	Data Structures	Augenthaler

Which of the following FDs are correct?

- TEACHER → TEXT
- COURSE → TEXT
- TEXT → COURSE

Slide 7-22

INFERENCE RULES FOR FDS

- Given a set of FDs F, we can *infer* additional FDs that hold whenever the FDs in F hold
- **Closure** of a set F of FDs is the set F^+ of all FDs that can be inferred from F
- Notation: X →Y is inferred from F
  $F \models X \rightarrow Y$

Slide 7-23

INFERENCE RULES FOR FDS (CONT)

- IR1 (**Reflexive**) If $Y \subset X$, then X → Y
- IR2 (**Augmentation**)
  If X → Y, then XZ → YZ
- IR3 (**Transitive**)
  If X → Y and Y → Z, then X → Z
- IR1, IR2, IR3 (Armstrong's inference rules) form a *sound* and *complete* set of inference rules

Slide 7-24

Inference Rules for FDs (cont)

- IR4 (**Decomposition**)
  If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
- IR5 (**Union**)
  If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
- IR6 (**Pseudotransitivity**)
  If $X \rightarrow Y$ and $WY \rightarrow Z$, then $WX \rightarrow Z$
- The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property)

Slide 7-25

Example 2 (Exam Question)

Given relation R(A, B, C, D) and a set of functional dependencies F={A →B, AC →D, BC →A, BC →D, CD →A}, show that AC → D is redundant:

- using Armstrong's theorems
- using the definition of equivalent sets of functional dependencies

Slide 7-26

Inference Rules (cont)

- **Closure** of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X
- X^+ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F
- Algorithm for determining X^+

```
X+ := X;
repeat
    oldX+ := X+;
    for each FD Y → Z in F do
        if Y ⊆ X+ then X+ := X+ ∪ Z;
until (oldX+ = X+);
```

Slide 7-27

Example 3 (Exam question)

- A relational schema R (A, B, C, D, E) has the set of functional dependencies F={A →BC, CD →E, AC →E, B →D, E →AB}. Determine all candidate keys in R. Justify your answers.

Slide 7-28

Equivalence of Sets of FDs

- Two sets of FDs F and G are **equivalent** if:
  - every FD in F can be inferred from G, *and*
  - every FD in G can be inferred from F
- Hence, F and G are equivalent if $F^+ = G^+$
- F **covers** G if every FD in G can be inferred from F (i.e., if $G^+ \subset F^+$)
- F and G are equivalent if F covers G and G covers F
- There is an algorithm for checking equivalence of sets of FDs

Slide 7-29

Algorithm for checking whether F covers E

- Calculate X^+ with respect to F for each FD X→Y in E;
- Check whether this X^+ includes the attributes in Y.
- **Example 4**
  Check whether F and G are equivalent
  F={A →C, AC →D, E →AD, E →H}
  G={A →CD, E →AH}

Slide 7-30

Minimal Sets of FDs

A set of FDs is **minimal**:

(1) Every dependency in F has a single attribute for its RHS;
(2) We cannot remove any dependency from F and have a set of dependencies that is equivalent to F;
(3) We cannot replace any dependency X → A in F with a dependency Y → A, where Y ⊂ X and still have a set of dependencies that is equivalent to F.

Slide 7-31

Minimal Sets of FDs (cont)

- Every set of FDs has an equivalent minimal set
- There can be several equivalent minimal sets
- There is no simple algorithm for computing a minimal set of FDs that is equivalent to a set F of FDs

Slide 7-32

Example 5

Determine whether the set of functional dependencies G is minimal or not. If it is not, find the minimal set of FDs equivalent to G.

G={SSN→{EName, BDate, Address, DNumber},
DNumber → {DName, DMgrSsn}}

Slide 7-33

Normalization of Relations

- **Normalization**: the process of decomposing unsatisfactory "bad" relations by breaking up their attributes into smaller relations

- **Normal form**: Condition using keys and FDs of a relation to certify whether a relation schema is in a particular normal form

Slide 7-34

Practical Use of Normal Forms

- **Normalization** is carried out in practice so that the resulting designs are of high quality and meet the desirable properties
- The practical utility of these normal forms becomes questionable when the constraints on which they are based are **hard to understand** or to **detect**
- The database designers ***need not*** normalize to the highest possible normal form (usually up to 3NF or BCNF)
- **Denormalization:** the process of storing the join of higher normal form relations as a base relation—which is in a lower normal form

Slide 7-35

Definitions of Keys

- A **superkey** of a relation schema $R = \{A_1, A_2, ..., A_n\}$ is a set of attributes $S \subset R$ with the property that no two tuples t_1 and t_2 in any legal relation state r of R will have $t_1[S] = t_2[S]$

- A **key** K is a superkey with the *additional property* that removal of any attribute from K will cause K not to be a superkey any more.

Slide 7-36

Definitions of Keys (cont)

- If a relation schema has more than one key, each is called a **candidate key.** One of the candidate keys is *arbitrarily* designated to be the **primary key,** and the others are called *secondary keys*.
- A **prime attribute** must be a member of *some candidate key*
- A **nonprime attribute** is not a prime attribute—that is, it is not a member of any candidate key.

First Normal Form

- Disallows composite attributes, multivalued attributes, and **nested relations**; attributes whose values *for an individual tuple* are non-atomic
- The only attribute values permitted by 1NF are single atomic (indivisible) values.
- Considered to be part of the definition of relation

(a)
DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations

(b)
DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)
DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

Figure 14.9
Normalization into 1NF. (a) A relation schema that is not in 1NF. (b) Example state of relation DEPARTMENT. (c) 1NF version of the same relation with redundancy.

Second Normal Form

- **Full functional dependency:** a FD Y → Z where removal of any attribute from Y means the FD does not hold any more

  Examples:

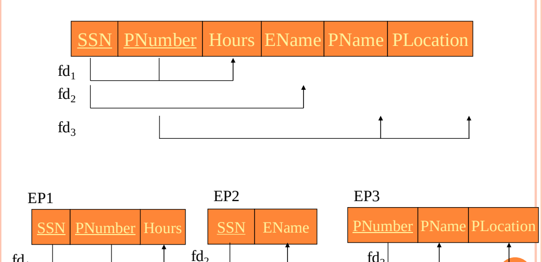
{SSN, PNUMBER} → HOURS is a full FD since neither SSN → HOURS nor PNUMBER → HOURS hold

{SSN, PNUMBER} → ENAME is *not* a full FD (it is called a *partial dependency*) since SSN → ENAME also holds

Second Normal Form

- A relation schema R is in **second normal form** (**2NF**) if every non-prime attribute A in R is fully functionally dependent on the entire primary key
- R can be decomposed into 2NF relations via the process of 2NF normalization

Example 6: EMP_PROJ



Transitive Functional Dependency

- a FD $X \rightarrow Z$ that can be derived from two FDs $X \rightarrow Y$ and $Y \rightarrow Z$
- SSN → DMGRSSN is a *transitive* FD since SSN → DNUMBER and DNUMBER → DMGRSSN hold
- SSN → ENAME is *non-transitive* since there is no set of attributes X where SSN → X and X → ENAME

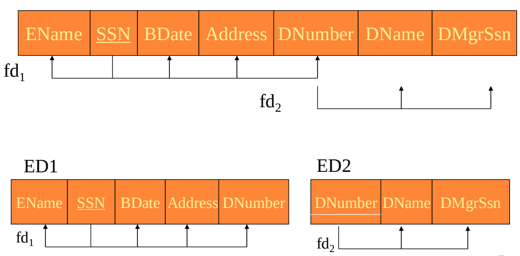
Slide 7-43

Third Normal Form

- A relation schema R is in **third normal form** (**3NF**) if it is in 2NF *and* no non-prime attribute A in R is transitively dependent on the primary key
- R can be decomposed into 3NF relations via the process of 3NF normalization

Slide 7-44

Example 7: EMP_DEPT



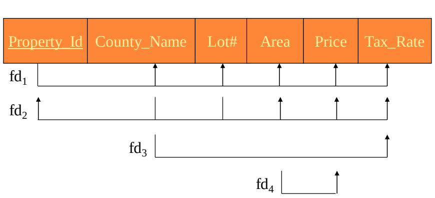
Slide 7-45

General Normal Form Definitions

- The following more general definitions take into account candidate keys, not just primary keys
- A relation schema R is in **second normal form** (**2NF**) if every non-prime attribute A in R is fully functionally dependent on *every key* of R
- A relation schema R is in **third normal form** (**3NF**) if whenever a FD X → A holds in R, then either:
  - (a) X is a superkey of R, or
  - (b) A is a prime attribute of R

Slide 7-46

Example 8: LOTS



(Fig 14.12)

Slide 7-47

Example 9

- Consider R(A, B, C, D, E, F, G, H, I, J) and the set of functional dependencies F = {AB → C, A → DE, B → F, F → GH, D → IJ}.
- What is the key of R?
- Decompose R into 2NF, then 3NF relations.

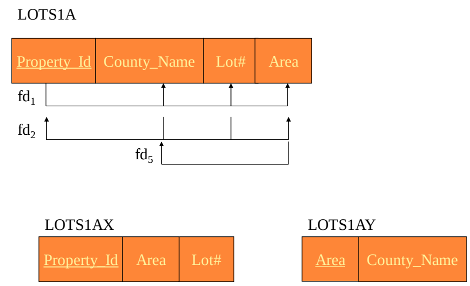
Slide 7-48

BCNF (Boyce-Codd Normal Form)

- A relation schema R is in **Boyce-Codd Normal Form** (**BCNF**) if whenever an FD X → A holds in R, then X is a superkey of R
- Each normal form is strictly stronger than the previous one
  - Every 2NF relation is in 1NF
  - Every 3NF relation is in 2NF
  - Every BCNF relation is in 3NF
- There exist relations that are in 3NF but not in BCNF
- The goal is to have each relation in BCNF (or 3NF)

Slide 7-49

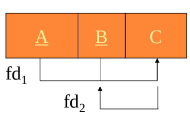
Decomposition to BCNF (Fig.14.12)



Slide 7-50

Achieving BCNF by Decomposition

- It is not always desirable to transform a relation into BCNF
- Some FDs may be lost



Slide 7-51

Example 10

- Determine the normal form of the relation R1(A, B, C, D) and, if necessary, decompose it into BCNF.
  F = {AB → CD, C → ABD, D → C}

Slide 7-52

Example 11

- Relation T lists dentist/patient data. A patient is given an appointment at a specific time and date with a dentist located at a particular room. On each day a dentist is allocated to a specific room for that day.
- Provide examples of insertion, deletion and update anomalies.
- Describe the process of normalizing the table to BCNF.

Slide 7-53

Example 11 (cont)

DNo	DName	PNo	PName	ADate	ATime	Room
101	Tony Smith	100	Gill White	12-sep	10:00	15
101	Tony Smith	105	Jill Bell	12-sep	12:00	15
124	Helen Adams	108	Ian McKay	12-sep	10:00	10
124	Helen Adams	108	Ian McKay	15-sep	14:00	10
132	Robin Pearson	110	John Walker	15-sep	16:30	15
132	Robin Pearson	105	Jill Bell	15-sep	18:00	15

Database Design Revisited

- Top-down design
- Relational synthesis

Slide 7-55

Relational decomposition

- R – universal relation schema
- Decomposition D={R_1, R_2, …, R_n}
- Properties of decompositions:
  - Attribute preservation $\cup_i R_i = R$
  - Dependency preservation
  - Lossless join

Slide 7-56

Dependency preservation

- D={R_1, R_2, …, R_n}
- The projection of F on Ri, $\pi_F(R_i)$, is the set of FDs in F^+ such that attributes in X ∩ Y are all contained in R_i.
- $((\pi_F(R_1)) \cup \ldots \cup (\pi_F(R_n)))^+ = F^+$
- It is always possible to find a dependency-preserving decomposition such that each relation is in 3NF

Slide 7-57

Lossless join

- Non-additive join
- $*(\pi_{R1}(r), \ldots \pi_{Rm}(r)) = r$
- Decomposition D = {R1, R2} is lossless iff either:
- $((R1 \cap R2) \rightarrow (R1 - R2)) \in F^+$ or
- $((R1 \cap R2) \rightarrow (R2 - R1)) \in F^+$

Slide 7-58

Relational synthesis

Dependency-preserving decomposition into 3NF relations

All FDs must be known!

1. Find a minimal cover G for F.
2. For each LHS X of a FD that appears in G create a relation schema ($X \cup A_1 \cup A_2 \ldots \cup A_m$), where $X \rightarrow A_1$, $X \rightarrow A_2$, …, $X \rightarrow A_m$ are the only FDs in G with X as LHS.
3. Place any remaining (unplaced) attributes in a single relation schema to ensure the attribute preservation property.

Slide 7-59

Finding a minimal cover G for F

1. Initialize G to F.
2. Replace each FD $X \rightarrow A_1, A_2, \ldots, A_n$ in G by n FDs: $X \rightarrow A_1$, $X \rightarrow A_2$, …, $X \rightarrow A_n$.
3. For each FD $X \rightarrow A$ in G:
   For each attribute B in X:
   Let Y = X - B, and J = $((G - (X \rightarrow A)) \cup \{Y \rightarrow A\}$.
   Compute Y^+ with respect to J, and Y^+ with respect to G.
   If Y^+ under G = Y^+ under J,
   replace $X \rightarrow A$ with $Y \rightarrow A$ in G, and set X = Y.
4. For each remaining FD $X \rightarrow A$ in G
   Compute X^+ with respect to $(G - (X \rightarrow A))$;
   If X^+ contains A, remove $X \rightarrow A$ from G.

Slide 7-60

Lossless Join and FD-Preserving Decomposition into 3NF

1. Find a minimal cover G for F.
2. For each LHS X of a FD that appears in G create a relation schema ($X \cup A_1 \cup A_2 \ldots \cup A_m$), where $X \rightarrow A_1$, $X \rightarrow A_2$, …, $X \rightarrow A_m$ are the only FDs in G with X as LHS.
3. Place any remaining (unplaced) attributes in a single relation schema to ensure the attribute preservation property.
4. If none of the relation schemas contain a key of R, create one more relation that contains the key.

Slide 7-61

Examples

- **Example 12:** Find a minimal cover for

  F={ABC → D, AB → C, C → B}.

- **Example 13:** Find the minimal set of 3NF relations given F={A → BC, BC → D}.

Slide 7-62