

Fundamentals of
**DATABASE
SYSTEMS**

FOURTH EDITION

ELMASRI  NAVATHE

Chapter 11

Relational Database Design Algorithms and Further Dependencies



Chapter Outline

0. Designing a Set of Relations
1. Properties of Relational Decompositions
2. Algorithms for Relational Database Schema
3. Multivalued Dependencies and Fourth Normal Form
4. Join Dependencies and Fifth Normal Form
5. Inclusion Dependencies
6. Other Dependencies and Normal Forms

DESIGNING A SET OF RELATIONS (1)

The Approach of Relational Synthesis (Bottom-up Design) :

- Assumes that all possible functional dependencies are known.
- First constructs a minimal set of FDs
- Then applies algorithms that construct a target set of 3NF or BCNF relations.
- Additional criteria may be needed to ensure the the *set of relations* in a relational database are satisfactory (see Algorithms 11.2 and 11.4).

DESIGNING A SET OF RELATIONS (2)

Goals:

- Lossless join property (a must) – algorithm 11.1 tests for general losslessness.
- Dependency preservation property – algorithms 11.3 decomposes a relation into BCNF components by sacrificing the dependency preservation.
- Additional normal forms
 - 4NF (based on multi-valued dependencies)
 - 5NF (based on join dependencies)

1. Properties of Relational Decompositions (1)

Relation Decomposition and Insufficiency of Normal Forms:

- **Universal Relation Schema:** a relation schema $R = \{A_1, A_2, \dots, A_n\}$ that includes all the attributes of the database.
- **Universal relation assumption:** every attribute name is unique.
- **Decomposition:** The process of decomposing the universal relation schema R into a set of relation schemas $D = \{R_1, R_2, \dots, R_m\}$ that will become the relational database schema by using the functional dependencies.

Properties of Relational Decompositions (2)

Relation Decomposition and Insufficiency of Normal Forms (cont.):

- **Attribute preservation condition:** Each attribute in R will appear in at least one relation schema R_i in the decomposition so that no attributes are “lost”.
- Another goal of decomposition is to have each individual relation R_i in the decomposition D be in BCNF or 3NF.
- Additional properties of decomposition are needed to prevent from generating spurious tuples

Properties of Relational Decompositions (3)

Dependency Preservation Property of a Decomposition :

Definition:

- Given a set of dependencies F on R , the **projection** of F on R_i , denoted by $\rho_{R_i}(F)$ where R_i is a subset of R , is the set of dependencies $X \rightarrow Y$ in F^+ such that the attributes in $X \cup Y$ are all contained in R_i . *Hence, the projection of F on each relation schema R_i in the decomposition D is the set of functional dependencies in F^+ , the closure of F , such that all their left- and right-hand-side attributes are in R_i .*

Properties of Relational Decompositions (4)

Dependency Preservation Property of a Decomposition (cont.):

- **Dependency Preservation Property:** a decomposition $D = \{R_1, R_2, \dots, R_m\}$ of R is **dependency-preserving** with respect to F if the union of the projections of F on each R_i in D is equivalent to F ; that is, $((\pi_{R_1}(F)) \cup \dots \cup (\pi_{R_m}(F)))^+ = F^+$

(See examples in Fig 10.12a and Fig 10.11)

Claim 1: It is always possible to find a dependency-preserving decomposition D with respect to F such that each relation R_i in D is in 3nf.

Properties of Relational Decompositions (5)

Lossless (Non-additive) Join Property of a Decomposition:

Definition:

- **Lossless join property:** a decomposition $D = \{R_1, R_2, \dots, R_m\}$ of R has the **lossless (nonadditive) join property** with respect to the set of dependencies F on R if, for *every* relation state r of R that satisfies F , the following holds, where $*$ is the natural join of all the relations in D :

$$* (\pi_{R_1}(r), \dots, \pi_{R_m}(r)) = r$$

Note: The word loss in *lossless* refers to *loss of information*, not to loss of tuples. In fact, for “loss of information” a better term is “addition of spurious information”

Properties of Relational Decompositions (6)

Lossless (Non-additive) Join Property of a Decomposition (cont.):

Algorithm 11.1: Testing for Lossless Join Property

Input: A universal relation R , a decomposition $D = \{R_1, R_2, \dots, R_m\}$ of R , and a set F of functional dependencies.

1. Create an initial matrix S with one row i for each relation R_i in D , and one column j for each attribute A_j in R .
2. Set $S(i,j) := b_{ij}$ for all matrix entries. (* each b_{ij} is a distinct symbol associated with indices (i,j) *).
3. For each row i representing relation schema R_i
 {for each column j representing attribute A_j
 {if (relation R_i includes attribute A_j) then set $S(i,j) := a_{ij}$ };};
(* each a_j is a distinct symbol associated with index (j) *)

Properties of Relational Decompositions (7)

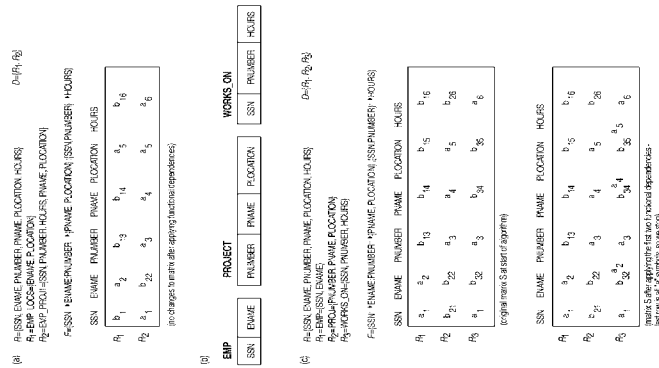
Lossless (Non-additive) Join Property of a Decomposition (cont.):

Algorithm 11.1: Testing for Lossless Join Property (cont.)

4. Repeat the following loop until a *complete loop execution* results in no changes to S
 {for each functional dependency $X \rightarrow Y$ in F
 {for all rows in S which have the same symbols in the columns corresponding to attributes in X
 {make the symbols in each column that correspond to an attribute in Y be the same in all these rows as follows: if any of the rows has an “ a ” symbol for the column, set the other rows to that same “ a ” symbol in the column. If no “ a ” symbol exists for the attribute in any of the rows, choose one of the “ b ” symbols that appear in one of the rows for the attribute and set the other rows to that same “ b ” symbol in the column ;};};
5. If a row is made up entirely of “ a ” symbols, then the decomposition has the lossless join property; otherwise it does not.

Properties of Relational Decompositions (8)

Figure 15.1 The lossless join testing algorithm. (a) Applying the algorithm to test the decomposition of EMP_PROJ into EMP_PROJ1 and EMP_LOCS. (b) Another decomposition of EMP_PROJ. (c) Applying the algorithm to the decomposition in Figure 15.1(b).



© Addison Wesley Longman, Inc. 2000. Elmasri/Navathe, Fundamentals of Database Systems, Third Edition

Note: The above figure is now called Figure 11.1 in Edition 4

Elmasri/Navathe, Fundamentals of Database Systems, Fourth Edition Chapter 11-13

Copyright © 2004 Ramez Elmasri and Shamkant Navathe

Properties of Relational Decompositions (9)

Testing Binary Decompositions for Lossless Join Property:

- **Binary Decomposition:** decomposition of a relation R into two relations.
- **PROPERTY LJ1 (lossless join test for binary decompositions):** A decomposition $D = \{R_1, R_2\}$ of R has the lossless join property with respect to a set of functional dependencies F on R if and only if either
 - The f.d. $((R_1 \cap R_2) \rightarrow (R_1 - R_2))$ is in F^+ , or
 - The f.d. $((R_1 \cap R_2) \rightarrow (R_2 - R_1))$ is in F^+ .

Elmasri/Navathe, Fundamentals of Database Systems, Fourth Edition Chapter 11-14

Copyright © 2004 Ramez Elmasri and Shamkant Navathe

Properties of Relational Decompositions (10)

Successive Lossless Join Decomposition:

- **Claim 2 (Preservation of non-additivity in successive decompositions):**

If a decomposition $D = \{R_1, R_2, \dots, R_m\}$ of R has the lossless (non-additive) join property with respect to a set of functional dependencies F on R , and if a decomposition $D_i = \{Q_1, Q_2, \dots, Q_k\}$ of R_i has the lossless (non-additive) join property with respect to the projection of F on R_i , then the decomposition $D_2 = \{R_1, R_2, \dots, R_{i-1}, Q_1, Q_2, \dots, Q_k, R_{i+1}, \dots, R_m\}$ of R has the non-additive join property with respect to F .

2. Algorithms for Relational Database Schema Design (1)

Algorithm 11.2: Relational Synthesis into 3NF with Dependency Preservation (*Relational Synthesis Algorithm*)

Input: A universal relation R and a set of functional dependencies F on the attributes of R .

1. Find a minimal cover G for F (use Algorithm 10.2);
2. For each left-hand-side X of a functional dependency that appears in G , create a relation schema in D with attributes $\{X \cup \{A_1\} \cup \{A_2\} \dots \cup \{A_k\}\}$, where $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_k$ are the only dependencies in G with X as left-hand-side (X is the key of this relation) ;
3. Place any remaining attributes (that have not been placed in any relation) in a single relation schema to ensure the attribute preservation property.

Claim 3: Every relation schema created by Algorithm 11.2 is in 3NF.

Algorithms for Relational Database Schema Design (2)

Algorithm 11.3: Relational Decomposition into BCNF with Lossless (non-additive) join property

Input: A universal relation R and a set of functional dependencies F on the attributes of R .

1. Set $D := \{R\}$;
2. While there is a relation schema Q in D that is not in BCNF do {
 choose a relation schema Q in D that is not in BCNF;
 find a functional dependency $X \rightarrow Y$ in Q that violates BCNF;
 replace Q in D by two relation schemas $(Q - Y)$ and $(X \cup Y)$;
};

Assumption: No null values are allowed for the join attributes.

Algorithms for Relational Database Schema Design (3)

Algorithm 11.4 Relational Synthesis into 3NF with Dependency Preservation and Lossless (Non-Additive) Join Property

Input: A universal relation R and a set of functional dependencies F on the attributes of R .

1. Find a minimal cover G for F (Use Algorithm 10.2).
2. For each left-hand-side X of a functional dependency that appears in G , create a relation schema in D with attributes $\{X \cup \{A_1\} \cup \{A_2\} \dots \cup \{A_k\}\}$, where $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_k$ are the only dependencies in G with X as left-hand-side (X is the key of this relation).
3. If none of the relation schemas in D contains a key of R , then create one more relation schema in D that contains attributes that form a key of R . (Use Algorithm 11.4a to find the key of R)

Algorithms for Relational Database Schema Design (4)

Algorithm 11.4a Finding a Key K for R Given a set F of Functional Dependencies

Input: A universal relation R and a set of functional dependencies F on the attributes of R .

1. Set $K := R$.
2. For each attribute A in K {
 - compute $(K - A)^+$ with respect to F ;
 - If $(K - A)^+$ contains all the attributes in R ,
 - then set $K := K - \{A\}$; }

Algorithms for Relational Database Schema Design (5)

Figure 15.2 Illustrating null value join issues. (a) A database with nulls for some join attributes. (b) Result of applying NATURAL JOIN to the EMPLOYEE and DEPARTMENT relations. (c) Result of applying OUTER JOIN to EMPLOYEE and DEPARTMENT.

(a) EMPLOYEE and DEPARTMENT tables:

EMPLOYEE	DEPARTMENT
EMPNO	DEPTNO
1000	10
1001	10
1002	10
1003	10
1004	10
1005	10
1006	10
1007	10
1008	10
1009	10
1010	10
1011	10
1012	10
1013	10
1014	10
1015	10
1016	10
1017	10
1018	10
1019	10
1020	10
1021	10
1022	10
1023	10
1024	10
1025	10
1026	10
1027	10
1028	10
1029	10
1030	10
1031	10
1032	10
1033	10
1034	10
1035	10
1036	10
1037	10
1038	10
1039	10
1040	10
1041	10
1042	10
1043	10
1044	10
1045	10
1046	10
1047	10
1048	10
1049	10
1050	10
1051	10
1052	10
1053	10
1054	10
1055	10
1056	10
1057	10
1058	10
1059	10
1060	10
1061	10
1062	10
1063	10
1064	10
1065	10
1066	10
1067	10
1068	10
1069	10
1070	10
1071	10
1072	10
1073	10
1074	10
1075	10
1076	10
1077	10
1078	10
1079	10
1080	10
1081	10
1082	10
1083	10
1084	10
1085	10
1086	10
1087	10
1088	10
1089	10
1090	10
1091	10
1092	10
1093	10
1094	10
1095	10
1096	10
1097	10
1098	10
1099	10
1100	10

(b) NATURAL JOIN result:

EMPNO	DEPTNO
1000	10
1001	10
1002	10
1003	10
1004	10
1005	10
1006	10
1007	10
1008	10
1009	10
1010	10
1011	10
1012	10
1013	10
1014	10
1015	10
1016	10
1017	10
1018	10
1019	10
1020	10
1021	10
1022	10
1023	10
1024	10
1025	10
1026	10
1027	10
1028	10
1029	10
1030	10
1031	10
1032	10
1033	10
1034	10
1035	10
1036	10
1037	10
1038	10
1039	10
1040	10
1041	10
1042	10
1043	10
1044	10
1045	10
1046	10
1047	10
1048	10
1049	10
1050	10
1051	10
1052	10
1053	10
1054	10
1055	10
1056	10
1057	10
1058	10
1059	10
1060	10
1061	10
1062	10
1063	10
1064	10
1065	10
1066	10
1067	10
1068	10
1069	10
1070	10
1071	10
1072	10
1073	10
1074	10
1075	10
1076	10
1077	10
1078	10
1079	10
1080	10
1081	10
1082	10
1083	10
1084	10
1085	10
1086	10
1087	10
1088	10
1089	10
1090	10
1091	10
1092	10
1093	10
1094	10
1095	10
1096	10
1097	10
1098	10
1099	10
1100	10

(c) OUTER JOIN result:

EMPNO	DEPTNO
1000	10
1001	10
1002	10
1003	10
1004	10
1005	10
1006	10
1007	10
1008	10
1009	10
1010	10
1011	10
1012	10
1013	10
1014	10
1015	10
1016	10
1017	10
1018	10
1019	10
1020	10
1021	10
1022	10
1023	10
1024	10
1025	10
1026	10
1027	10
1028	10
1029	10
1030	10
1031	10
1032	10
1033	10
1034	10
1035	10
1036	10
1037	10
1038	10
1039	10
1040	10
1041	10
1042	10
1043	10
1044	10
1045	10
1046	10
1047	10
1048	10
1049	10
1050	10
1051	10
1052	10
1053	10
1054	10
1055	10
1056	10
1057	10
1058	10
1059	10
1060	10
1061	10
1062	10
1063	10
1064	10
1065	10
1066	10
1067	10
1068	10
1069	10
1070	10
1071	10
1072	10
1073	10
1074	10
1075	10
1076	10
1077	10
1078	10
1079	10
1080	10
1081	10
1082	10
1083	10
1084	10
1085	10
1086	10
1087	10
1088	10
1089	10
1090	10
1091	10
1092	10
1093	10
1094	10
1095	10
1096	10
1097	10
1098	10
1099	10
1100	10

Note: The above figure is now called Figure 11.2 in Edition 4

Algorithms for Relational Database Schema Design (6)

Figure 15.3 The “dangling tuple” problem. (a) The relation `EMPLOYEE_1` (includes all attributes of `EMPLOYEE` except `DNUMBER`). (b) The relation `EMPLOYEE_2` (includes `DNUMBER` attribute with null values). (c) The relation `EMPLOYEE_3` (includes `DNUMBER` attribute but does not include tuples for which `DNUMBER` has null values).

(a) EMPLOYEE_1			
ENAME	SSN	BDATE	ADDRESS
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX
Zolaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX
Wallace, Jennifer S.	987654321	1941-09-20	291 Berry, Belaire, TX
Narayan, Ramesh K.	666884444	1982-09-15	975 Fire Oak, Humble, TX
English, Joyce A.	453453453	1972-07-31	5631 Fike, Houston, TX
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX
Berg, James E.	888665555	1937-11-10	450 Stone, Houston, TX
Berger, Arcturus C.	999775555	1985-04-26	6530 Braes, Bellaire, TX
Benitez, Carlos M.	888664444	1963-01-09	7654 Beech, Houston, TX

(b) EMPLOYEE_2		(c) EMPLOYEE_3	
SSN	DNUM	SSN	DNUM
123456789	5	123456789	5
333445555	5	333445555	5
999887777	4	999887777	4
987654321	4	987654321	4
666884444	5	666884444	5
453453453	5	453453453	5
987987987	4	987987987	4
888665555	1	888665555	1
999775555	null		
888664444	null		

© Addison Wesley Longman, Inc. 2000, Elmasri/Navathe, Fundamentals of Database Systems, Third Edition

Note: The above figure is now called Figure 11.3 in Edition 4

Elmasri/Navathe, Fundamentals of Database Systems, Fourth Edition Chapter 11-21

Copyright © 2004 Ramez Elmasri and Shamkant Navathe

Algorithms for Relational Database Schema Design (7)

Discussion of Normalization Algorithms:

Problems:

- The database designer must first specify *all* the relevant functional dependencies among the database attributes.
- These algorithms are *not deterministic* in general.
- It is not always possible to find a decomposition into relation schemas that preserves dependencies and allows each relation schema in the decomposition to be in BCNF (instead of 3NF as in Algorithm 11.4).

Elmasri/Navathe, Fundamentals of Database Systems, Fourth Edition Chapter 11-22

Copyright © 2004 Ramez Elmasri and Shamkant Navathe

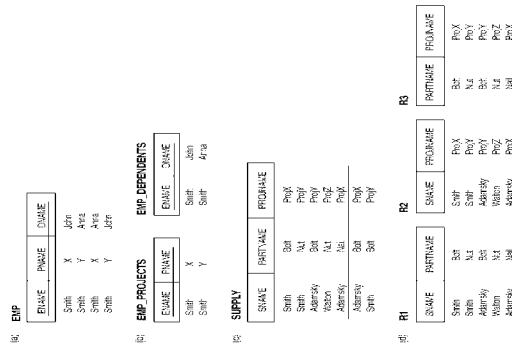
Algorithms for Relational Database Schema Design (8)

Table 11.1 Summary of some of the algorithms discussed above

Algorithm	Input	Output	Properties/Purpose	Remarks
11.1	A decomposition D of R and a set F of functional dependencies	Boolean result: yes or no for lossless join property	Testing for non-additive join decomposition	See a simpler test in Section 11.1.4 for binary decompositions
11.2	Set of functional dependencies F	A set of relations in 3NF	Dependency preservation	No guarantee of satisfying lossless join property
11.3	Set of functional dependencies F	A set of relations in BCNF	Lossless join decomposition	No guarantee of dependency preservation
11.4	Set of functional dependencies F	A set of relations in 3NF	Lossless join and dependency preserving decomposition	May not achieve BCNF
11.4a	Relation schema R with a set of functional dependencies F	Key K of R	To find a key K (which is a subset of R)	The entire relation R is always a default superkey

3. Multivalued Dependencies and Fourth Normal Form (1)

Figure 15.4 Fourth and fifth normal form. (a) The EMP relation with two MVDS: ENAME → PNAME and ENAME → DNAME. (b) Decomposing EMP into two relations in 4NF. (c) The relation SUPPLY with no MVDS satisfies 4NF but does not satisfy 5NF if the JD(R1, R2, R3) holds. (d) Decomposing SUPPLY into three 5NF relations.



Note: The above figure is now called Figure 11.4 in Edition 4

Multivalued Dependencies and Fourth Normal Form (2)

Definition:

- A **multivalued dependency (MVD)** $X \twoheadrightarrow Y$ specified on relation schema R , where X and Y are both subsets of R , specifies the following constraint on any relation state r of R : If two tuples t_1 and t_2 exist in r such that $t_1[X] = t_2[X]$, then two tuples t_3 and t_4 should also exist in r with the following properties, where we use Z to denote $(R - (X \cup Y))$:
 - $t_3[X] = t_4[X] = t_1[X] = t_2[X]$.
 - $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$.
 - $t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$.
- An MVD $X \twoheadrightarrow Y$ in R is called a **trivial MVD** if (a) Y is a subset of X , or (b) $X \cup Y = R$.

Multivalued Dependencies and Fourth Normal Form (3)

Inference Rules for Functional and Multivalued Dependencies:

- IR1 (**reflexive rule for FDs**): If $X \supseteq Y$, then $X \rightarrow Y$.
- IR2 (**augmentation rule for FDs**): $\{X \rightarrow Y\}$ [DEFINES] $XZ \rightarrow YZ$.
- IR3 (**transitive rule for FDs**): $\{X \rightarrow Y, Y \rightarrow Z\}$ [DEFINES] $X \rightarrow Z$.
- IR4 (**complementation rule for MVDs**): $\{X \twoheadrightarrow Y\}$ [DEFINES] $\{X \twoheadrightarrow (R - (X \cup Y))\}$.
- IR5 (**augmentation rule for MVDs**): If $X \twoheadrightarrow Y$ and $W \subseteq Z$ then $WX \twoheadrightarrow YZ$.
- IR6 (**transitive rule for MVDs**): $\{X \twoheadrightarrow Y, Y \twoheadrightarrow Z\}$ [DEFINES] $X \twoheadrightarrow Z$.
- IR7 (**replication rule for FD to MVD**): $\{X \rightarrow Y\}$ [DEFINES] $X \twoheadrightarrow Y$.
- IR8 (**coalescence rule for FDs and MVDs**): If $X \twoheadrightarrow Y$ and there exists W with the properties that (a) $W \cap Y$ is empty, (b) $W \rightarrow Z$, and (c) $Y \supseteq Z$, then $X \rightarrow Z$.

Multivalued Dependencies and Fourth Normal Form (4)

Definition:

- A relation schema R is in **4NF** with respect to a set of dependencies F (that includes functional dependencies and multivalued dependencies) if, for every *nontrivial* multivalued dependency $X \twoheadrightarrow Y$ in F^+ , X is a superkey for R .

Note: F^+ is the (complete) set of all dependencies (functional or multivalued) that will hold in every relation state r of R that satisfies F . It is also called the **closure** of F .

Multivalued Dependencies and Fourth Normal Form (5)

Figure 15.5 Benefits of 4NF. (a) The EMP relation with some additional tuples. (b) Projecting EMP on EMP_PROJECTS and EMP_DEPENDENTS.

(a) EMP			(b) EMP_PROJECTS	
ENAME	PNAME	DNAME	ENAME	PNAME
Smith	X	John	Smith	X
Smith	Y	Anna	Smith	Y
Smith	X	Anna	Brown	W
Smith	Y	John	Brown	X
Brown	W	Jim	Brown	Y
Brown	X	Jim	Brown	Z
Brown	Y	Jim		
Brown	Z	Jim		
Brown	W	Joan		
Brown	X	Joan		
Brown	Y	Joan		
Brown	Z	Joan		
Brown	W	Bob		
Brown	X	Bob		
Brown	Y	Bob		
Brown	Z	Bob		

EMP_DEPENDENTS	
ENAME	DNAME
Smith	Anna
Smith	John
Brown	Jim
Brown	Joan
Brown	Bob

Note: The above figure is now called Figure 11.5 in Edition 4

Multivalued Dependencies and Fourth Normal Form (6)

Lossless (Non-additive) Join Decomposition into 4NF Relations:

● PROPERTY LJ1'

The relation schemas R_1 and R_2 form a lossless (non-additive) join decomposition of R with respect to a set F of functional and multivalued dependencies if and only if

$$(R_1 \cap R_2) \twoheadrightarrow (R_1 - R_2)$$

or by symmetry, if and only if

$$(R_1 \cap R_2) \twoheadrightarrow (R_2 - R_1).$$

Multivalued Dependencies and Fourth Normal Form (7)

Algorithm 11.5: Relational decomposition into 4NF relations with non-additive join property

Input: A universal relation R and a set of functional and multivalued dependencies F .

1. Set $D := \{ R \}$;
2. While there is a relation schema Q in D that is not in 4NF do
{ choose a relation schema Q in D that is not in 4NF;
find a nontrivial MVD $X \twoheadrightarrow Y$ in Q that violates 4NF;
replace Q in D by two relation schemas $(Q - Y)$ and $(X \cup Y)$;
};

4. Join Dependencies and Fifth Normal Form (1)

Definition:

- A **join dependency (JD)**, denoted by $JD(R_1, R_2, \dots, R_n)$, specified on relation schema R , specifies a constraint on the states r of R . The constraint states that every legal state r of R should have a non-additive join decomposition into R_1, R_2, \dots, R_n ; that is, for every such r we have

$$* (\pi_{R_1}(r), \pi_{R_2}(r), \dots, \pi_{R_n}(r)) = r$$

Note: an MVD is a special case of a JD where $n = 2$.

- A join dependency $JD(R_1, R_2, \dots, R_n)$, specified on relation schema R , is a **trivial JD** if one of the relation schemas R_i in $JD(R_1, R_2, \dots, R_n)$ is equal to R .

Join Dependencies and Fifth Normal Form (2)

Definition:

- A relation schema R is in **fifth normal form (5NF)** (or **Project-Join Normal Form (PJNF)**) with respect to a set F of functional, multivalued, and join dependencies if, for every nontrivial join dependency $JD(R_1, R_2, \dots, R_n)$ in F^+ (that is, implied by F), every R_i is a superkey of R .

Relation SUPPLY with Join Dependency and conversion to Fifth Normal Form

Figure 15.4 Fourth and fifth normal form. (a) The EMP relation with two MVDS: ENAME → PNAME and ENAME → DNAME. (b) Decomposing EMP into two relations in 4NF. (c) The relation SUPPLY with no MVDS satisfies 4NF but does not satisfy 5NF if the JD(R1, R2, R3) holds. (d) Decomposing SUPPLY into three 5NF relations.

EMP			
ENAME	PNAME	DNAME	
Smith	X	John	
Smith	Y	AKA	
Smith	X	AKA	
Smith	Y	John	

EMP_PROJECTS			
ENAME	PNAME	DNAME	
Smith	Y	John	
Smith	Y	AKA	

EMP_DEPENDENTS			
ENAME	PNAME	DNAME	
Smith	Y	John	
Smith	Y	AKA	

SUPPLY			
SNAME	PNAME	PQUANTITY	
Smith	MI	Proj1	
Adamsky	MI	Proj1	
Adamsky	MI	Proj2	
Adamsky	MI	Proj3	
Adamsky	MI	Proj4	
Smith	MI	Proj1	

R1		
SNAME	PNAME	
Smith	MI	
Adamsky	MI	
Adamsky	MI	

R2		
SNAME	PNAME	PQUANTITY
Smith	MI	Proj1
Adamsky	MI	Proj2
Adamsky	MI	Proj3
Adamsky	MI	Proj4

R3		
SNAME	PNAME	PQUANTITY
Smith	MI	Proj1
Adamsky	MI	Proj1
Adamsky	MI	Proj2
Adamsky	MI	Proj3
Adamsky	MI	Proj4

© Addison Wesley Longman, Inc. 2000. Elmasri/Navathe, Fundamentals of Database Systems, Third Edition

Note: Only show parts c and d of the the above figure. (now called Figure 11.4 in Edition 4)

Elmasri/Navathe, Fundamentals of Database Systems, Fourth Edition Chapter 11-33

Copyright © 2004 Ramez Elmasri and Shamkant Navathe

5. Inclusion Dependencies (1)

Definition:

- An **inclusion dependency** $R.X < S.Y$ between two sets of attributes— X of relation schema R , and Y of relation schema S —specifies the constraint that, at any specific time when r is a relation state of R and s a relation state of S , we must have

$$\pi_X(r(R)) \text{ <subset-or-equal> } \pi_Y(s(S))$$

Note: The \subset (subset) relationship does not necessarily have to be a proper subset. The sets of attributes on which the inclusion dependency is specified— X of R and Y of S —must have the same number of attributes. In addition, the domains for each pair of corresponding attributes should be compatible.

Elmasri/Navathe, Fundamentals of Database Systems, Fourth Edition Chapter 11-34

Copyright © 2004 Ramez Elmasri and Shamkant Navathe

Inclusion Dependencies (2)

Objective of Inclusion Dependencies:

To formalize two types of interrelational constraints which cannot be expressed using F.D.s or MVDs:

- Referential integrity constraints
- Class/subclass relationships

● Inclusion dependency inference rules

IDIR1 (reflexivity): $R.X < R.X$.

IDIR2 (attribute correspondence): If $R.X < S.Y$

where $X = \{A_1, A_2, \dots, A_n\}$ and $Y = \{B_1, B_2, \dots, B_n\}$ and A_i Corresponds-to B_i , then $R.A_i < S.B_i$
for $1 \leq i \leq n$.

IDIR3 (transitivity): If $R.X < S.Y$ and $S.Y < T.Z$, then $R.X < T.Z$.

6. Other Dependencies and Normal Forms (1)

Template Dependencies:

- Template dependencies provide a technique for representing constraints in relations that typically have no easy and formal definitions.
- The idea is to specify a template—or example—that defines each constraint or dependency.
- There are two types of templates: tuple-generating templates and constraint-generating templates.
- A template consists of a number of **hypothesis tuples** that are meant to show an example of the tuples that may appear in one or more relations. The other part of the template is the **template conclusion**.

Other Dependencies and Normal Forms (2)

Figure 15.6 Templates for common types of dependencies. (a) Template for the functional dependency $X \rightarrow Y$. (b) Template for the multivalued dependency $X \twoheadrightarrow Y$. (c) Template for the inclusion dependency $R.X \subset S.Y$.

	$R = \{ A, B, C, D \}$		
(a)			
hypothesis	$\begin{array}{ccc} a_1 & b_1 & c_1 \\ \hline a_1 & b_1 & c_2 \end{array}$	$X = \{ A, B \}$ $Y = \{ C, D \}$	
conclusion	$c_1 = c_2 \text{ and } d_1 = d_2$		
(b)			
hypothesis	$\begin{array}{cccc} a_1 & b_1 & c_1 & d_1 \\ \hline a_1 & b_1 & c_2 & d_2 \end{array}$	$X = \{ A, B \}$ $Y = \{ C \}$	
conclusion	$\begin{array}{cccc} a_1 & b_1 & c_2 & d_1 \\ a_1 & b_1 & c_1 & d_2 \end{array}$		
(c)			
hypothesis	$\begin{array}{cccc} a_1 & b_1 & c_1 & d_1 \\ \hline & & & \end{array}$	$S = \{ E, F, G \}$ $X = \{ C, D \}$ $Y = \{ E, F \}$	
conclusion		$c_1 \ d_1 \ g$	

© Addison Wesley Longman, Inc. 2000, Elmasri/Navathe, Fundamentals of Database Systems, Third Edition

Note: The above figure is now called Figure 11.6 in Edition 4

Other Dependencies and Normal Forms (3)

Figure 15.7 Templates for the constraint that an employee's salary must be less than the supervisor's salary.

$EMPLOYEE = \{ NAME, SSN, \dots, SALARY, SUPERVISORSSN \}$

	a	b	c	d
hypothesis	e	d	f	g
conclusion			$c < f$	

© Addison Wesley Longman, Inc. 2000, Elmasri/Navathe, Fundamentals of Database Systems, Third Edition

Note: The above figure is now called Figure 11.7 in Edition 4

Other Dependencies and Normal Forms (4)

Domain-Key Normal Form (DKNF):

- **Definition:** A relation schema is said to be in **DKNF** if all constraints and dependencies that should hold on the valid relation states can be enforced simply by enforcing the domain constraints and key constraints on the relation.
- The **idea** is to specify (theoretically, at least) the “*ultimate normal form*” that takes into account all possible types of dependencies and constraints. .
- For a relation in DKNF, it becomes very straightforward to enforce all database constraints by simply checking that each attribute value in a tuple is of the appropriate domain and that every key constraint is enforced.
- The practical utility of DKNF is limited