Multivalued Dependency

by

Dr. Pratik Roy

Introduction

- An employee may work on several projects and may have several dependents, and employee's projects and dependents are independent of one another.
- To keep the relation state consistent, and to avoid any spurious relationship between the two independent attributes, we must have a separate tuple to represent every combination of an employee's dependent and an employee's project.
- Whenever two independent 1:N relationships A:B and A:C are mixed in the same relation R(A, B, C), a multivalued dependency (MVD) may arise.

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

Introduction

- Multivalued dependencies are a consequence of first normal form (1NF), which disallows an attribute in a tuple to have a set of values, and the accompanying process of converting an unnormalized relation into 1NF.
- If we have two or more multivalued independent attributes in the same relation schema, we get into a problem of having to repeat every value of one of the attributes with every value of the other attribute to keep the relation state consistent and to maintain the independence among the attributes involved.
- This constraint is specified by a multivalued dependency.

Definition

- A multivalued dependency $X \rightarrow\rightarrow Y$ 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 $t_4[Z] = t_1[Z]$

- Whenever $X \rightarrow\rightarrow Y$ holds, we say that X **multidetermines** Y.
- Because of the symmetry in the definition, whenever $X \rightarrow\rightarrow Y$ holds in R, so does $X \rightarrow\rightarrow Z$. Hence, $X \rightarrow\rightarrow Y$ implies $X \rightarrow\rightarrow Z$, and therefore it is sometimes written as $X \rightarrow\rightarrow Y | Z$.

Example

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

EMP relation with two MVDs: Ename →→ Pname and Ename →→ Dname

EMP schema is in BCNF because *no* functional dependencies hold in EMP

Trivial and Nontrivial Multivalued Dependencies

- An MVD X $\rightarrow\rightarrow$ Y in R is called a **trivial MVD** if
  (a) Y is a subset of X, or
  (b) X $\cup$ Y = R
- An MVD that satisfies neither (a) nor (b) is called a **nontrivial MVD**.
- If we have a nontrivial MVD in a relation, we may have to repeat values redundantly in the tuples.
- This redundancy is clearly undesirable.
- We need to define a fourth normal form that is stronger than BCNF and disallows relation schemas such as EMP.
- Relations containing nontrivial MVDs tend to be **all-key relations**
  - Key is all their attributes taken together.

Fourth Normal Form (4NF)

- 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 $X \rightarrow\rightarrow Y$ in F^+ X is a superkey for R.
- An all-key relation is always in BCNF since it has no FDs.
- An all-key relation such as the EMP relation, which has no FDs but has the MVD Ename$\rightarrow\rightarrow$ Pname | Dname, is not in 4NF.
- A relation that is not in 4NF due to a nontrivial MVD must be decomposed to convert it into a set of relations in 4NF.
- The decomposition removes the redundancy caused by the MVD.

Example

EMP

<u>Ename</u>	<u>Pname</u>	<u>Dname</u>
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

Ename $\rightarrow\rightarrow$ Pname

EMP_PROJECTS

<u>Ename</u>	<u>Pname</u>
Smith	X
Smith	Y

Ename $\rightarrow\rightarrow$ Dname

EMP_DEPENDENTS

<u>Ename</u>	<u>Dname</u>
Smith	John
Smith	Anna

Both EMP_PROJECTS and EMP_DEPENDENTS are in 4NF, because MVDs Ename $\rightarrow\rightarrow$ Pname in EMP_PROJECTS and Ename $\rightarrow\rightarrow$ Dname in EMP_DEPENDENTS are trivial MVDs.

Join Dependency

- A join dependency (JD), denoted by JD(R_1, R_2, ..., 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 nonadditive join decomposition into R_1, R_2, ..., R_n. Hence, for every such r we have

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

- A join dependency JD(R_1, R_2, ..., 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, ..., R_n) is equal to R.
- A relation is said to have join dependency if it can be recreated by joining multiple sub relations and each of these sub relations has a subset of the attributes of the original relation.

Fifth Normal Form (5NF)

- Fifth normal form (5NF) is a level of database normalization designed to reduce redundancy in relational databases.
- A relation is said to be in 5NF if and only if it satisfies 4NF and no join dependency exists.
- **Definition:** A relation schema R is in 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, ..., R_n) in F^+ (that is, implied by F), every R_i is a superkey of R.

Properties of 5NF

- A relation R is in 5NF if and only if it satisfies following conditions:
  - R should be in 4NF (no multi-valued dependency exists)
  - It cannot undergo lossless decomposition (join dependency)

Example

- Consider the relation R below having the schema

R(<u>supplier, product, consumer</u>)

- Primary key is a combination of all three attributes of the relation.

Supplier	Product	Consumer
S1	P1	C1
S1	P2	C1
S2	P1	C1
S3	P3	C3

Example

Table 1

<u>Supplier</u>	<u>Product</u>	<u>Consumer</u>
S1	P1	C1
S1	P2	C1
S2	P1	C1
S3	P3	C3

➡

Table 2

Supplier	Product
S1	P1
S1	P2
S2	P1
S3	P3

Table 3

Consumer	Product
C1	P1
C1	P2
C3	P3

Table 4

Supplier	Consumer
S1	C1
S2	C1
S3	C3

- Table 2, Table 3 and Table 4 when joined yield the original table (Table 1). Hence join dependency exists in Table 1.
- Table 1 is not in 5NF or PJNF.
- Table 2, Table 3 and Table 4 satisfy 5NF as it has no multivalued dependency and cannot be decomposed further (join dependency does not exists).

5NF

- In some cases when we combine decomposed tables, the resultant table may not be equivalent to the original table, in that case the original table is said to be in 5NF provided it is already in 4NF.

Inclusion Dependencies

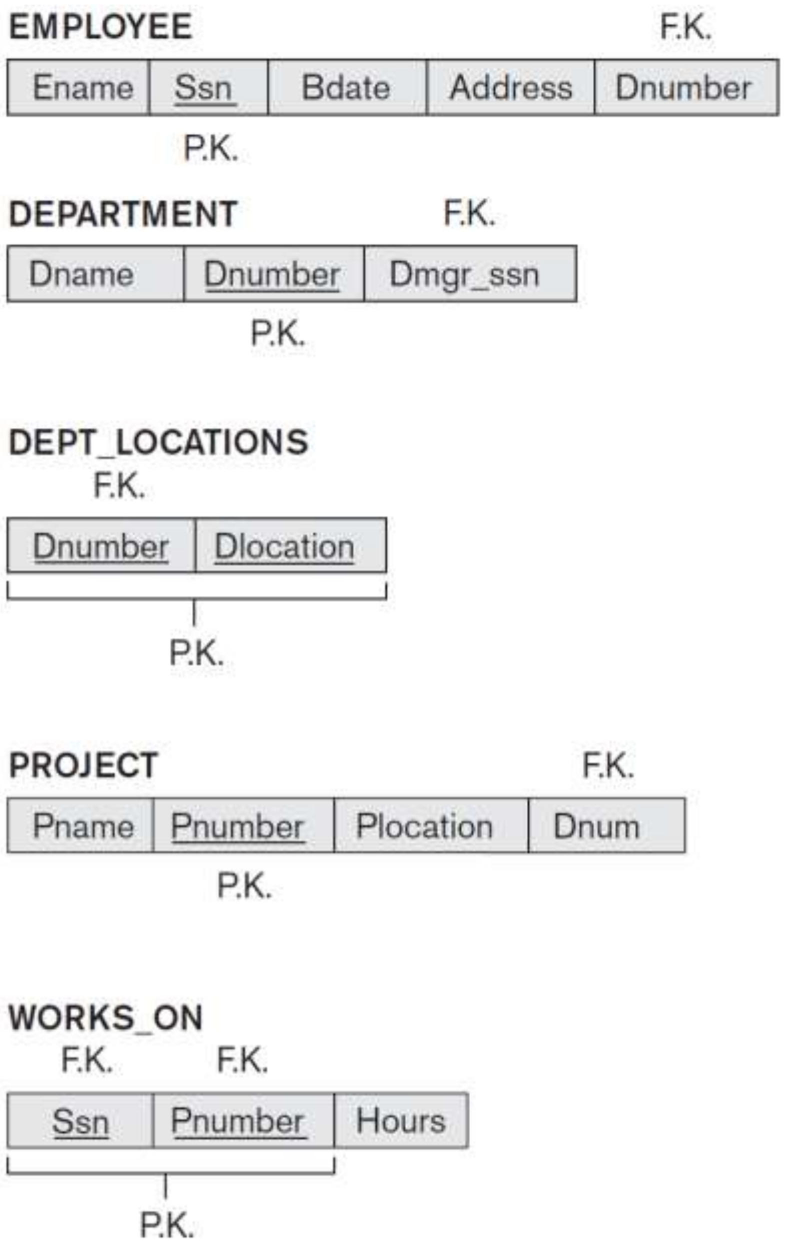
- Inclusion dependencies are defined in order to formalize two types of interrelational constraints:
  - The foreign key (or referential integrity) constraint cannot be specified as a functional or multivalued dependency because it relates attributes across relations.
  - The constraint between two relations that represent a class/subclass relationship also has no formal definition in terms of the functional, multivalued, and join dependencies.
- 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

$$\pi_X(r(R)) \subseteq \pi_Y(s(S))$$

Inclusion Dependencies

- Sets of attributes on which the inclusion dependency is specified—X of R and Y of S—must have same number of attributes.
- Domains for each pair of corresponding attributes should be compatible.
  - If $X = \{A_1, A_2, ..., A_n\}$ and $Y = \{B_1, B_2, ..., B_n\}$, one possible correspondence is to have dom(A_i) compatible with dom(B_i) for $1 \leq i \leq n$.
  - A_i corresponds to B_i.

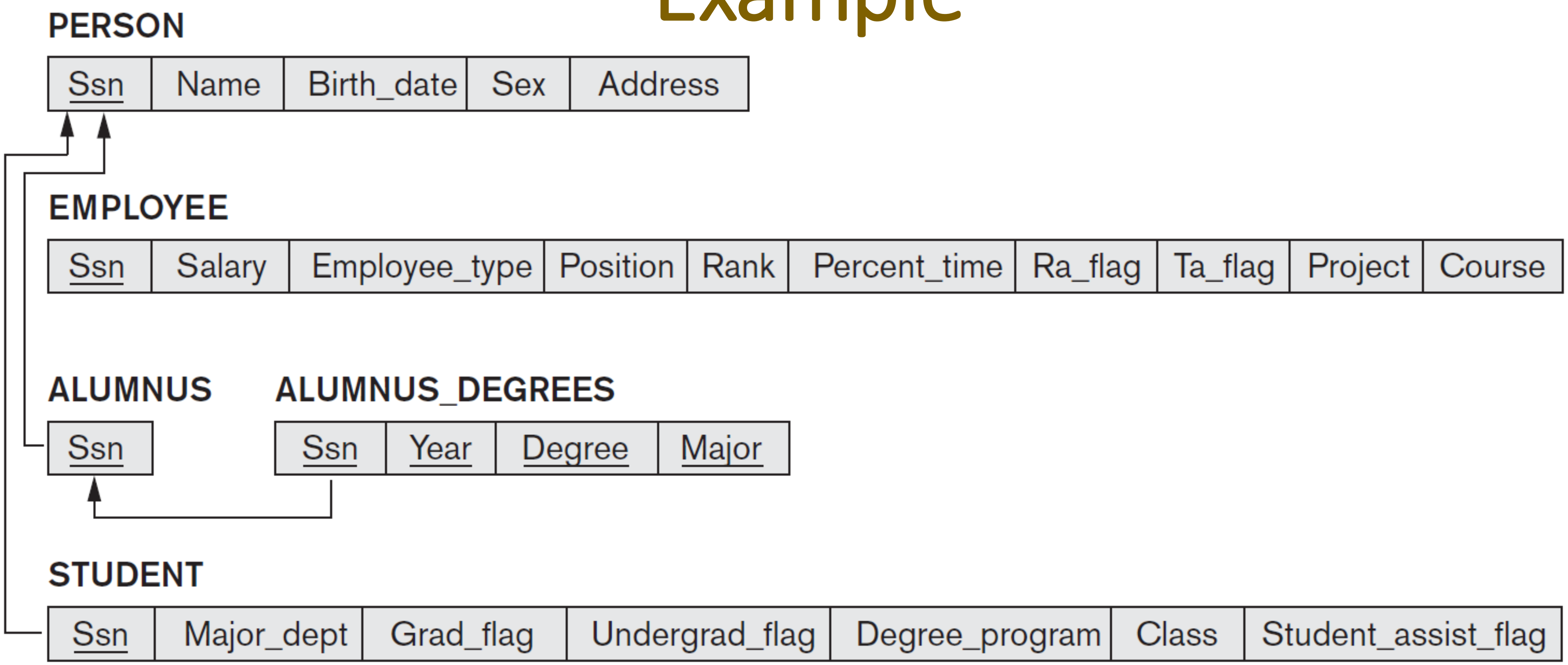
Example



- DEPARTMENT.Dmgr_ssn < EMPLOYEE.Ssn
- WORKS_ON.Ssn < EMPLOYEE.Ssn
- EMPLOYEE.Dnumber < DEPARTMENT.Dnumber
- PROJECT.Dnum < DEPARTMENT.Dnumber
- WORKS_ON.Pnumber < PROJECT.Pnumber
- DEPT_LOCATIONS.Dnumber < DEPARTMENT.Dnumber

These inclusion dependencies represent **referential integrity constraints**.

Example



- EMPLOYEE.Ssn < PERSON.Ssn
- ALUMNUS.Ssn < PERSON.Ssn
- STUDENT.Ssn < PERSON.Ssn

These inclusion dependencies represent **class/subclass relationships**.

Inclusion Dependency Inference Rules

- **IDIR1 (reflexivity)**

$$R.X < R.X.$$

- **IDIR2 (attribute correspondence)**
  - If $R.X < S.Y$, where $X = \{A_1, A_2, ..., A_n\}$ and $Y = \{B_1, B_2, ..., B_n\}$ 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 $R.X < T.Z$.

So far, no normal forms have been developed based on inclusion dependencies.