International Institute of Information Technology Bangalore

Functional Dependencies and Normalization

CS 301

Background

- Each *relation schema* consists of a number of attributes, and the *relational DB schema* consists of a number of relation schemas.

- We have assumed that attributes are grouped to form a relation schema as decided by the DB designer or by mapping a DB schema design from a conceptual data model such as the ER data model.

- These models make the designer "identify entity types and relationship types and their respective attributes", which leads to a natural and logical grouping of the attributes into relations.

- However, we still need some formal way of analyzing **why one grouping of attributes into a relation schema may be better than another**.

- There are two levels at which we can discuss the goodness of relation schema.

1. Logical (or conceptual) level – "*how users interpret the relation schemas and the meaning of their attributes*". Good relation schemas enable users to understand the meaning of the data in the relations and hence to formulate the query correctly.

2. Implementation (or physical storage) level – "*how the tuples in a base relation are stored and updated*". This applies to base relations – which will be physically stored as files.

- DB design may be performed using two approaches:

1. *A bottom-up design methodology* considers the basic relationship among individual attributes as the starting point to construct relation schema.

➢ Problem → We end up collecting a large number of binary relationships among attributes as the starting point. It is very difficult to capture binary relationships among all such pairs of attributes.

2. *A top-down design methodology* starts with a number of groupings of attributes into relations that exist together "naturally". These relations are analysed individually and collectively, leading to further decomposition until all desirable properties are met.

- The implicit goals of the DB design activity are *information preservation* and *minimum redundancy*.

- So, the relational design must preserve all of these concepts, which are originally captured in the conceptual design.

- Minimising redundancy implies minimising redundant storage of the same information and reducing the need for multiple updates to maintain consistency across multiple copies of the same information.

- Functional dependency is a "constraint among attributes that is an essential tool for formally measuring the appropriateness of attribute groupings into relation schemas".

"Goal of the chapter"

"Learn about the theory that has been developed with the goal of evaluating relational schemas for design quality – that is, to measure formally why one set of groupings of attributes into relation schemas is better than another."

1. Define functional dependency as a tool for measuring the appropriateness of attribute groupings into relation schemas.

2. Discuss normal forms and the process of normalization using functional dependencies.

3. Introduce multivalued and join dependency.

Design guidelines for relation schema

- Four informal guidelines that may be used as measures to determine the quality of relation schema design are:

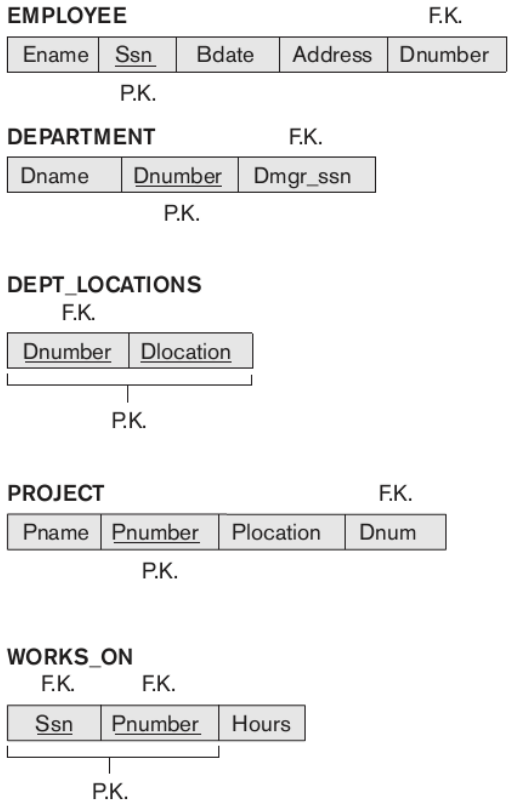
1. Making sure that the semantics of the attributes is clear in the schema.
2. Reducing the redundant information in tuples.
3. Reducing the NULL values in tuples.
4. Disallowing the possibility of generating spurious tuples.

Imparting clear semantics to attributes in relations

- The "semantics" of a relation refers to its meaning resulting from the interpretation of attribute values in a tuple.

- So, in general, the easier it is to explain the semantics of the relation – the better the relation schema design is.

- Consider a simplified version of the COMPANY relational DB schema.



EMPLOYEE

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

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

- The semantics of the EMPLOYEE, DEPARTMENT, PROJECT are clear. Example,

- Dnumber attribute in EMPLOYEE is a foreign key and a primary key for DEPARTMENT.

- Each DEPARTMENT tuple represents a department entity, and each PROJECT tuple represents a project entity.

- The attribute Dmgr_ssn of DEPARTMENT relates a department to the employee who is its manager, whereas Dnum of PROJECT relates a project to its controlling department; both are foreign key attributes.

- Similarly, the schema DEPT_LOCATIONS represents a multivalued attribute of DEPARTMENT, whereas WORKS_ON represents a M:N relationship between EMPLOYEE and PROJECT.

An informed guideline - 1

- Design a relation schema so that it is easy to explain its meaning.
- Do not combine attributes from multiple entry types and relationship types into a single relation.
- Intuitively, if a relation schema corresponds to one entity type or one relationship type, it is straightforward to explain its meaning.
- Otherwise, if the relation corresponds to a mixture of multiple entities and relationships, semantic ambiguities will result and the relation cannot be explained.

Violation of guideline - 1

- Consider:

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation

- They violate guideline – 1 by mixing attributes. Example, EMP_DEPT mixes attributes of employees and departments, and EMP_PROJ mixes attributes of employees and projects and the WORKS_ON relationship.

- However, they can be used as views.

Redundant information in tuples and update anomalies

- Grouping attributes into relation schemas has a significant effect on storage space. Example, compare

EMPLOYEE

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

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

Redundancy

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

Dnumber, Dname and Dmgr_ssn are repeated for every employee who works for that department.

EMPLOYEE

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

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

Redundancy Redundancy

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

Augments the WORKS_ON relation with additional attributes from EMPLOYEE and PROJECT.

- Storing natural joins of base relations leads to an additional problem referred to as update anomalies. These can be classified into
  - insertion,
  - deletion and
  - modification anomalies.

Redundancy

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

- These anomalies are undesirable and cause difficulties to maintain consistency of data as well as require unnecessary updates that can be avoided.

An informed guideline - 2

- Design the base relation schema so that no insertion, deletion, or modification anomalies are present in the relations.

- If any anomalies are present, note them clearly and make sure that the programs that update the DB will operate correctly.

Null values in tuples

- If many of the attributes do not apply to all tuples in the relation, we may have many NULLs in those tuples.
- This can waste space at the storage level and may also lead to problems with understanding the meaning of the attributes and with specifying JOIN operations at the logical level.
- SELECT and JOIN operations involve comparisons, if NULL values are present, the results may become unpredictable.

- NULLs can have multiple interpretations:

  - The attribute *does not apply* to this tuple. For example, Visa_status may not apply to U.S. students.
  - The attribute value for this tuple is *unknown*. For example, the Date_of_birth may be unknown for an employee.
  - The value is *known but absent*; that is, it has not been recorded yet. For example, the Home_Phone_Number for an employee may exist, but may not be available and recorded yet.

- Having the same representation for all NULLS compromises the different meanings they may have.

An informed guideline - 3

- As far as possible, avoid placing attributes in a base relation whose values may frequently be NULL.
- If NULLs are unavoidable, make sure that they apply in exceptional case only and do not apply to a majority of tuples in the relation.

Generation of spurious tuples

- Instead of the single relation:

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation

- Consider:

EMP_LOCS

Ename	Plocation

P.K.

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation

P.K.

Redundancy Redundancy

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

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

NATURAL JOIN operation on EMP_LOCS and EMP_PROJ1

* Spurious tuples

	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.

An informed guideline – 4

- Design relation schemas so that they can be joined with equality condition on attributes that are appropriately related (primary key, foreign key) pairs in a way that guarantees that no spurious tuples are generated.

- Avoid relations that contain matching attributes that are not (primary key, foreign key) combinations because joining on such attributes may produce spurious tuples.

Functional dependencies

- Let's now see a formal tool for analysis of relational schemas - functional dependency – that is used to define normal forms for relation schemas.

- A functional dependency (FD) is a constraint between two sets of attributes from the DB. Suppose that our relational DB schema has n attributes $A_1, A_2, \ldots, A_n$, where the whole DB is described by a single universal relation schema $R = \{A_1, A_2, \ldots, A_n\}$.

- Definition – A FD denoted as $X \rightarrow Y$, between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuples that can form a relation state r of R. "The constraint is that, for any two tuples t_1 and t_2 in r that have $t_1[X] = t_2[X]$, they must also have $t_1[Y] = t_2[Y]$."

What does this mean?

- $X \rightarrow Y$ is read as "there is a functional dependency from X to Y", or "Y is **functionally dependent** on X".
  - This means that the values of the Y component of a tuple in r depend on, or are *determined by*, the values of the X component.
  - That is, the values of the X component of a tuple uniquely (or **functionally**) determine the values of the Y component.
  - The set of attributes X is called the left-hand side (LHS) of the FD, and Y is called the right-hand side (RHS).

- Therefore, X functionally determines Y in a relation schema R if, and only if, whenever two tuples of $r(R)$ agree on their X-value, they must necessarily agree on their Y-value.

- Note:

1. If a constraint on *R* states that there cannot be more than one tuple with a given *X*-value in any relation instance *r(R)* – that is, *X* is a **candidate key** of *R* – this implies that $X \rightarrow Y$ for any subset of attributes *Y* of *R* (because the key constraint implies that no two tuples in any legal state *r(R)* will have the same value of *X*). If *X* is a candidate key of *R*, then $X \rightarrow R$.

2. If $X \rightarrow Y$ in *R*, this does not say whether or not $Y \rightarrow X$ in *R*.

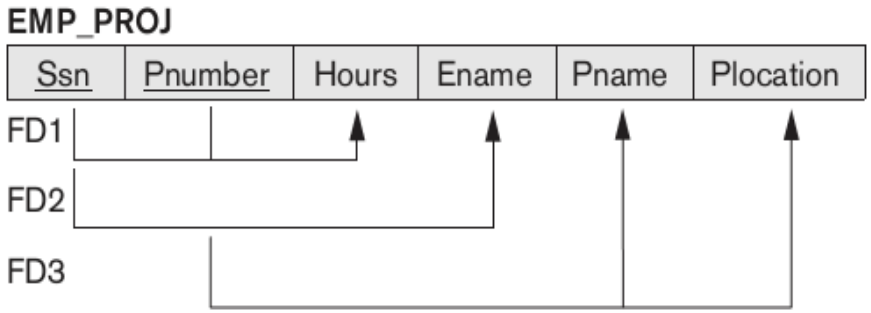
- A functional dependency is a property of the semantics or meaning of the attributes.
- The DB designer will use their understanding of the semantics of the attributes of *R* – that is, how they relate to one another – to specify the functional dependencies that should hold on all relation states *r* of *R*.

- Relation extensions $r(R)$ that satisfy the functional dependency constraints are called legal relation states of R.

- The main use of FD is "to describe a relation schema R by specifying constraints on its attributes that must hold at all times".

- Example-1: {State, Driver_license_number} → Ssn holds true for any adult US resident.

- Example-2: Consider EMP_PROJ schema



a. {Ssn, Pnumber} → Hours

b. Ssn → Ename

c. Pnumber → {Pname, Plocation}

- They specify that

(a) a combination of Ssn and Pnumber values uniquely determines the number of hours the employee currently works on the project per week (Hours),

(b) the value of an employee's Ssn uniquely determines the employee name (Ename), and

- Alternatively, we say that Ename is functionally determined by (or functionally dependent on) Ssn, or given a value of *Ssn*, we know the value of *Ename*.

(c) the value of a project's number (Pnumber) uniquely determines the project name (Pname) and location (Plocation).

- A FD is a property of the relation schema *R*, not of a particular legal relation state *r* of *R*. Therefore, FD cannot be inferred automatically from a given **relation extension** *r* but must be defined explicitly.

- Another example,

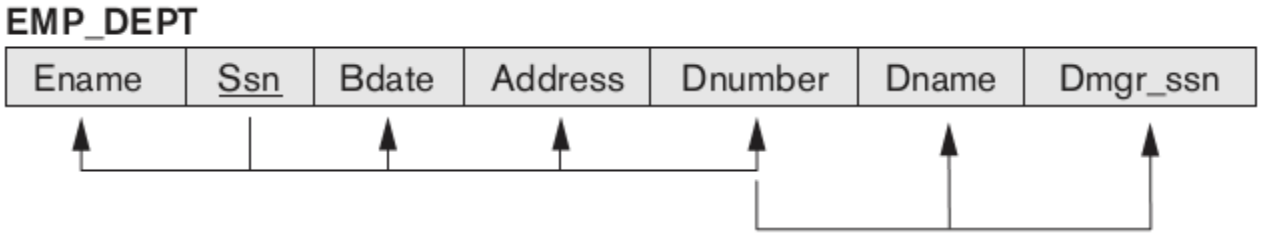
TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

A relation state of TEACH with a *possible* functional dependency TEXT → COURSE. However, TEACHER → COURSE, TEXT → TEACHER and COURSE → TEXT are ruled out.

- Text → Course may have a FD. This can be confirmed if found true while checking *all possible legal states* of TEACH.

- It is also sufficient to demonstrate a single counterexample to disprove a FD. Example, "Smith" teaches both "Data Structures" and "Data Management", we can conclude that "Teacher" does not functionally determine "Course".



A diagrammatic notation of FD.

- Each FD is displayed as a horizontal line.
- The left-hand-side (LHS) attributes of the FD are connected by vertical lines to the line representing the FD, whereas the right-hand-side (RHS) attributes are connected by the lines with arrows pointing towards the attributes.

Normal forms based on primary key

- Assume a set of functional dependencies is given for each relation and each relation has a primary key.

- This information combined with the "tests (conditions)" for normal forms drives the normalisation process for relational schema design.

- Relational design projects take one of the following two approaches:

  1. Perform a conceptual schema design using a conceptual model such as ER and map the conceptual design into a set of relations.
  2. Design the relations based on external knowledge derived from an existing implementation of files or forms or reports.

- Following either approaches, it is useful to evaluate the relations for goodness and decompose them further as needed to achieve higher normal forms.

Normalisation of relations

- Normalisation takes a relation schema through a series of tests to certify whether it satisfies a certain normal form.

- Codd in 1972 proposed first, second and third normal forms based on functional dependencies among the attributes of a relation. Later, Boyce and Codd proposed Boyce-Codd normal form (BCNF).

- Fourth normal form (4NF) and fifth normal form (5NF) were proposed based on the concepts of multivalued dependencies and join dependencies.

- Normalisation of data can be considered a process of analysing the given relation schemas based on their FDs and primary keys to achieve the desirable properties of

  (1) minimising redundancy and

  (2) minimising the insertion, deletion and update anomalies.

- A relation in non-normal form is decomposed into smaller relation schemas that contain a "*subset of the attributes*" and meet the test that was otherwise not met in the original relation.

- So, a normal form of a relation refers to the highest normal form condition that it meets, and hence indicates the degree to which it has been normalised.

- Note: Normal forms, when considered *in isolation* from other factors (see next slide), does not guarantee a good DB design.

- The process of normalisation through decomposition must also confirm the existence of additional properties that the relational schema should possess including:

- The nonadditive join or lossless join property – it guarantees that the spurious tuple generation does not occur with respect to the relation schemas created after decomposition.

Extremely critical.
Must be achieved.

- The dependency preservation property – it ensures that each FD is represented in some individual relation resulting after decomposition.

Although desirable,
sometimes sacrificed.

Practical use of normal forms

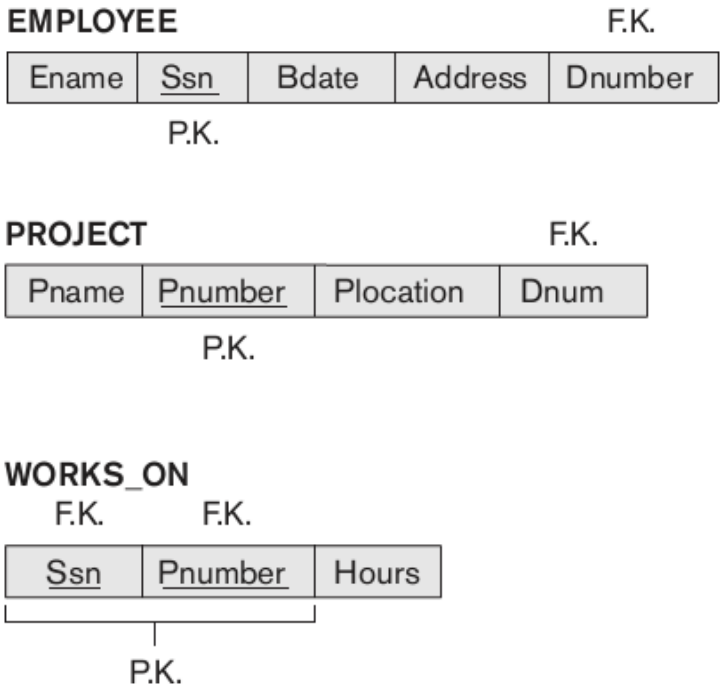
- For practical applications, DBs are evaluated up to only 3NF, BCNF or at most 4NF.

- DB designers need not normalise to the highest possible normal form. So, relations must be left in a lower normalisation status, such as 2NF for performance reasons or else they may be subject to dealing with the anomalies.

- So, denormalisation is the process of storing the join of higher normal form relations as a base relation, which is in a lower normal form.

- Recall superkey, key, candidate key and primary key (see 4_Relational Data Model – slide 26 and 28).

- An attribute of relation schema *R* is called a *prime attribute* of *R* if it is a member of *some candidate key* of *R*.
- In other words, an attribute is called *nonprime* if it is not a prime attribute – that is if it is not a member of any candidate key.
- Example:



Both Ssn and Pnumber are prime attributes of WORKS_ON, whereas other attributes of WORKS_ON are nonprime.

First normal form

- First normal form (1NF) historically was defined to disallow multivalued attributes, composite attributes and their combinations.

- It states that the domain of an attribute must include only **atomic (simple, indivisible)** ***values*** and that the value of any attribute in a tuple must be a **single value** from the domain of that attribute.

- Therefore, 1NF disallows having a set of values, a tuple of values, or a combination of both as an attribute value for a *single tuple*.

or

- 1NF disallows *relations within relations* or *relations as attribute values within tuples*.

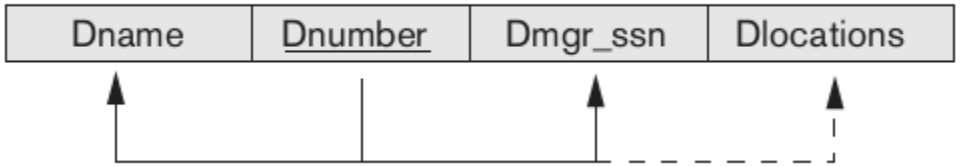
- Consider the DEPARTMENT relation schema whose primary key is Dnumber and suppose that we extend it by including the Dlocations attribute.

DEPARTMENT F.K.

Dname	Dnumber	Dmgr_ssn

P.K.

DEPARTMENT



- A sample relation state is as shown below:

DEPARTMENT

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

- This is not in 1NF because Dlocations is not an atomic attribute as seen in first tuple.

- There are two ways to look at the Dlocations attribute:

1. The domain of Dlocations contains atomic values, but some tuples can have a set of these values. Then, Dlocations **is not functionally dependent** on the primary key "Dnumber".

2. The domain of Dlocations **contains sets of values** and hence is nonatomic. In this case, Dnumber → Dlocation**s** because each set is considered a single member of the attribute domain.

- In either case, DEPARTMENT relation is **not in 1NF**.

- So, there are three possible ways to get the 1NF for this relation:

1. Remove the attribute Dlocations that violates 1NF and place it in a separate relation DEPT_LOCATIONS along with the primary key Dnumber of DEPARTMENT. The primary key of this newly formed relation is the combination {Dnumber, Dlocation}.

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

- So, a distinct tuple in DEPT_LOCATIONS exist for each location of a department. This decomposes the non-1NF relation into two 1NF relations.

2. Expand the key so that there will be a separate tuple in the original DEPARTMENT relation for each location of a DEPARTMENT.

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

- In this case, the primary key becomes the combination {Dnumber, Dlocation}. **This introduces redundancy**.

3. If a maximum number of values is known for the attributes – as such if it is known that at most three locations can exist for a department then replace the Dlocations attribute by three atomic attributes: Dlocation1, Dlocation2, Dlocation3. Disadvantage – **introduces NULL values if most departments have fewer than three location values**.

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

How do you write this query: List the departments that have 'Bellaire' as one of their locations.

- Conclusion – Of the three solutions, the first is the best because it does not suffer from redundancy and places no maximum limit on the number of values.

- First normal form also disallows multivalued attributes that are themselves composite and are called "nested relations" because each tuple "can have a relation *within it*".

- If nesting is allowed, EMP_PROJ is like:

EMP_PROJ		Projs	
Ssn	Ename	Pnumber	Hours

- Each tuple represents an employee entity, and a relation PROJS(Pnumber, Hours) *within each tuple* represents the employee's projects and the hours per week the employee works on each project as represented below:

EMP_PROJ(Ssn, Ename, {PROJS(Pnumber, Hours)})

- "{ }" identify the attribute PROJS as multivalued, and the list of component attributes that form PROJS are placed between "( )".

EMP_PROJ		Projs	
Ssn	Ename	Pnumber	Hours

EMP_PROJ

Ssn	Ename	Pnumber	Hours
123456789	Smith, John B.	1	32.5
		2	7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1	20.0
		2	20.0
333445555	Wong, Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0
999887777	Zelaya, Alicia J.	30	30.0
		10	10.0
987987987	Jabbar, Ahmad V.	10	35.0
		30	5.0
987654321	Wallace, Jennifer S.	30	20.0
		20	15.0
888665555	Borg, James E.	20	NULL

- In the above EMP_PROJ relation, Ssn is the primary key and Pnumber is the partial key* of the nested relation, so within each tuple, the nested relation must have unique values of Pnumber.

- To normalise this into 1NF, we remove the nested relation attributes "into a new relation" and propagate the primary key into it, where the primary key of the new relation will combine the partial key with the primary key of the original relation.

* A weak entity has a partial key (discriminator) – which is the attribute that can uniquely identify weak entities that are related to the same owner entity.

- So, decomposition will lead to EMP_PROJ1 and EMP_PROJ2.

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

- This recursive procedure when applied to a relation with multiple level nesting, unnests the relation to obtain a set of 1NF relations.

- Utility - This is useful in converting unnormalised relation schema with many levels of nesting into 1NF relations.

Some more Examples:

- Example 1:

CANDIDATE (Ssn, Name, {JOB_HIST (Company, Highest_position, {SAL_HIST (Year, Max_sal)})})

- The first normalisation using internal partial keys result in

CANDIDATE_1 (Ssn, Name)
CANDIDATE_JOB_HIST (Ssn, Company, Highest_position)
CANDIDATE_SAL_HIST (Ssn, Company, Year, Max-sal)

- Example 2:

PERSON (Ss#, {Car_lic#}, {Phone#})

  - This relation indicates that a person has multiple cars and multiple phones.

PERSON_IN_1NF (Ss#, Car_lic#, Phone#) ← All possible combinations of values lead to redundancy.

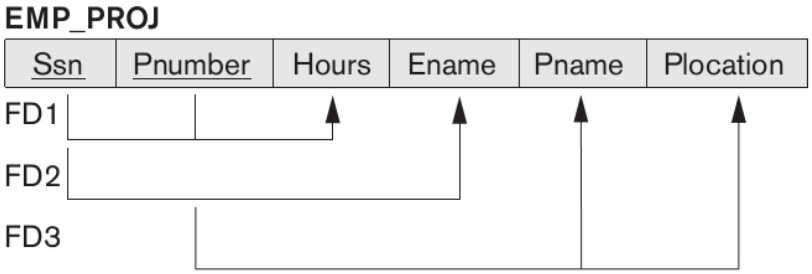
P1(Ss#, Car_lic#) and P2(Ss#, Phone#) ← Further decomposition solves this problem.

Second normal form

- 2NF is based on the concept of *full functional dependency*.

> A functional dependency $X \rightarrow Y$ is a **full functional dependency** if removal of any attribute A from X means that the dependency does not hold anymore; that is, for any attribute $A \in X$, $(X - \{A\})$ does *not* functionally determine Y. A functional dependency $X \rightarrow Y$ is a **partial dependency** if some attribute $A \in X$ can be removed from X and the dependency still holds; that is, for some $A \in X$, $(X - \{A\}) \rightarrow Y$.

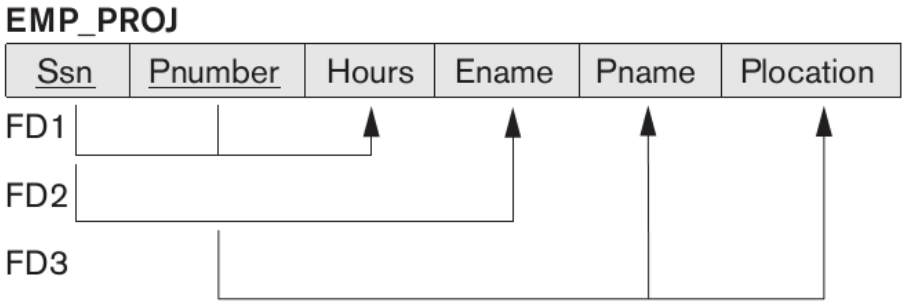
- Example:



- {Ssn, Pnumber} → Hours is a full dependency (neither Ssn → Hours nor Pnumber → Hours hold).
- However, the dependency {Ssn, Pnumber} → Ename is partial because Ssn → Ename holds.

- Definition - A relation schema *R* is in 2NF if every nonprime* attribute *A* in *R* is fully functionally dependent on the primary key of *R*.

- The test for 2NF involves testing for functional dependencies whose left-hand side attributes are part of the primary key. So, EMP_PROJ is in 1NF and not in 2NF.



* An attribute is called *nonprime* if it is not a member of any candidate key.

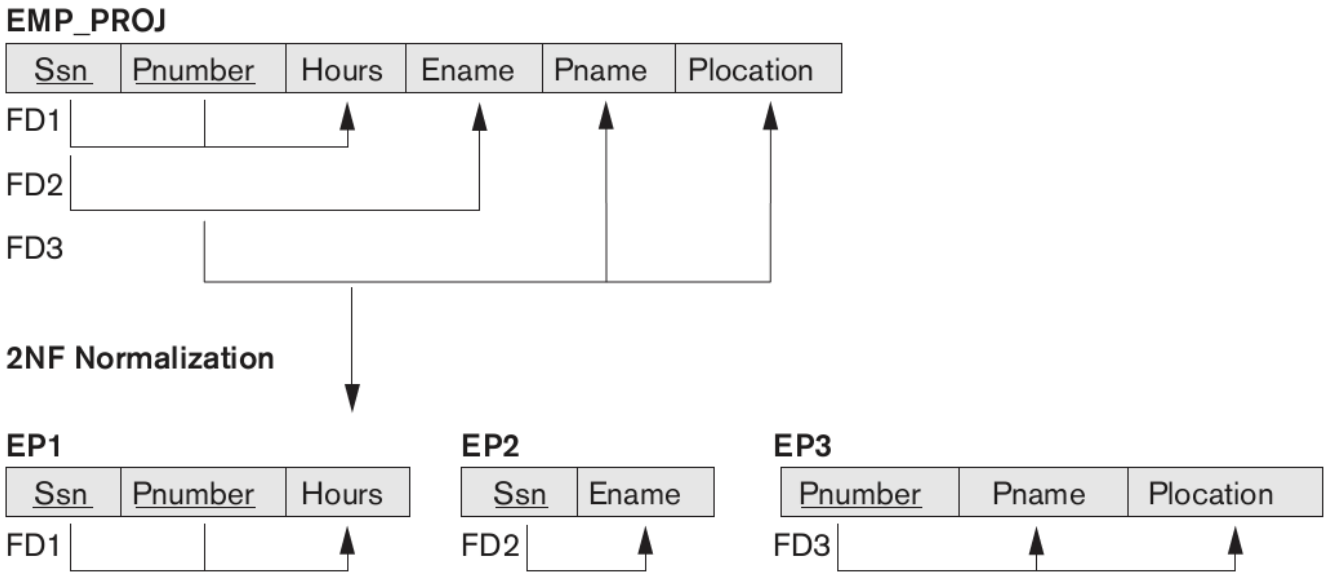
- The nonprime attribute Ename violates 2NF because of FD2, as do the nonprime attributes Pname and Plocation because of FD3.

- Each of the functional dependencies FD2 and FD3 violates 2NF because Ename can be functionally determined by "only Ssn", and both Pname and Plocation can be functionally determined by "only Pnumber".

- Attributes Ssn and Pnumber are a part of the primary key {Ssn, Pnumber} of EMP_PROJ, thus violating the 2NF test.

- 2NF normalisation can be performed in which nonprime attributes are associated only with the part of the primary key on which they are fully functionally dependent.

- So, FD1, FD2 and FD3 lead to decomposition of EMP_PROJ into EP1, EP2 and EP3 which are in 2NF (see next slide).



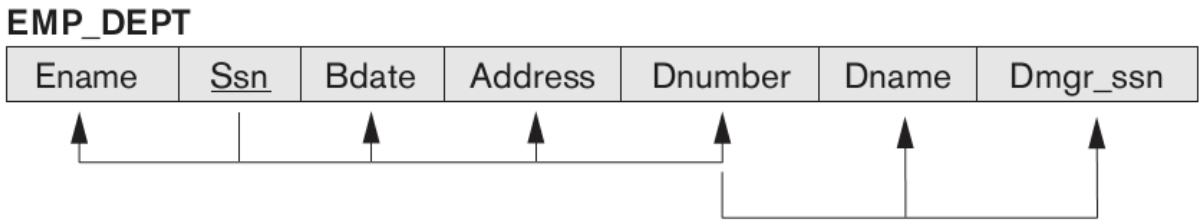
Normalising EMP_PROJ into 2NF relations.

Third normal form

- 3NF is based on the concept of "*transitive dependency*".

- A functional dependency $X \rightarrow Y$ in a relation schema R is a ***transitive dependency*** if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold.

- Example, the dependency Ssn → Dmgr_ssn is transitive through "Dnumber" in EMP_DEPT.

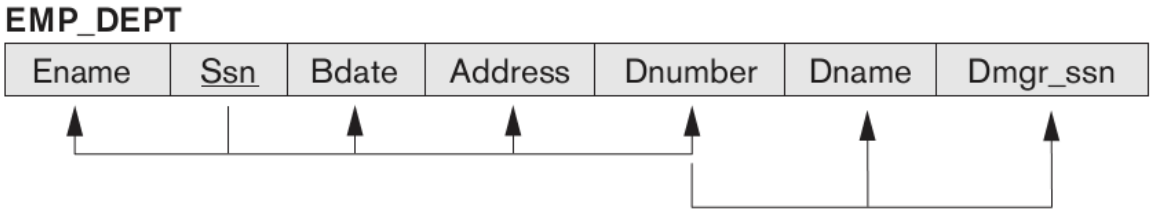


- The reason is that both the dependencies Ssn → Dnumber and Dnumber → Dmgr_ssn hold *and* Dnumber is neither a key itself nor a subset of the key of EMP_DEPT.

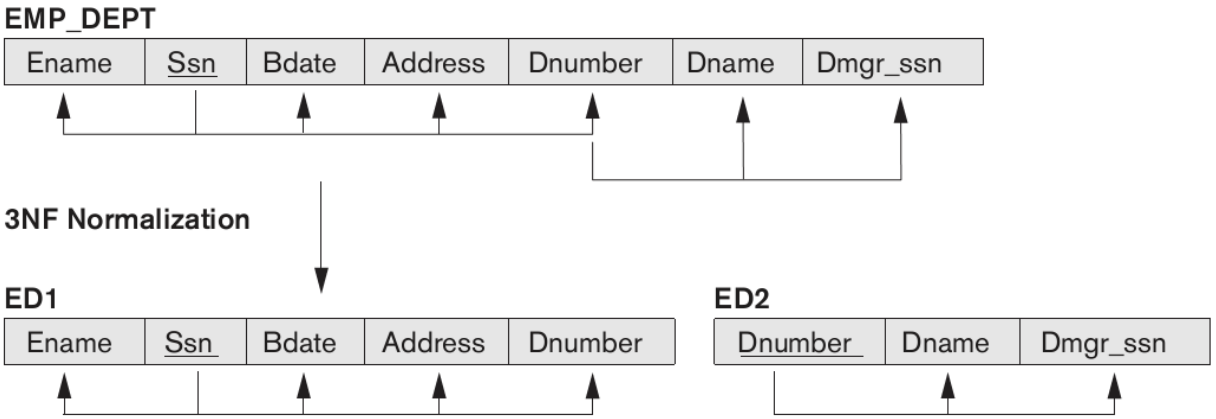
- So, dependency of Dmgr_ssn on Dnumber is undesirable in EMP_DEPT since Dnumber is not a key of EMP_DEPT.

- Definition – A relation schema *R* is in 3NF if it satisfies 2NF and no nonprime attribute of *R* is transitively dependent on the primary key.

- The relation schema EMP_DEPT is in 2NF since no partial dependencies on a key exist. However it is not in 3NF because of the transitive dependency of Dmgr_ssn (and also Dname) on Ssn via Dnumber.



- So, we can normalise EMP_DEPT by decomposing it into the two 3NF relation schemas ED1 and ED2 (see next slide).



- So, ED1 and ED2 represent independent facts about employees and departments (both of which are entities).

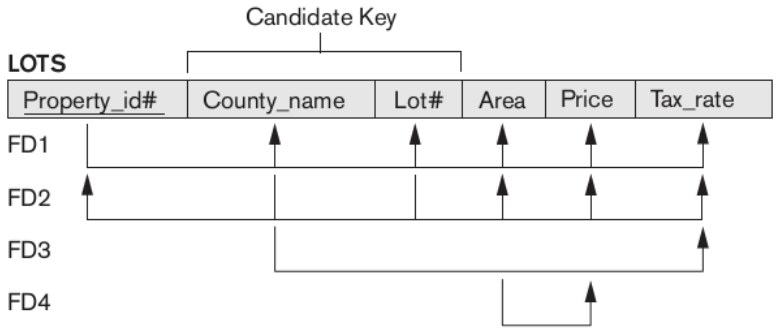
- A NATURAL JOIN on ED1 and ED2 will lead to the original relation EMP_DEPT without generating spurious tuples.

Normal Form	Test	Remedy (Normalization)
First (1NF)	Relation should have no multivalued attributes or nested relations.	Form new relations for each multivalued attribute or nested relation.
Second (2NF)	For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key.	Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it.
Third (3NF)	Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key.	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).

Summary of normal forms based on primary keys, the tests used in each case, and the corresponding remedy or normalisation performed to achieve the normal form.

Problems with third normal form

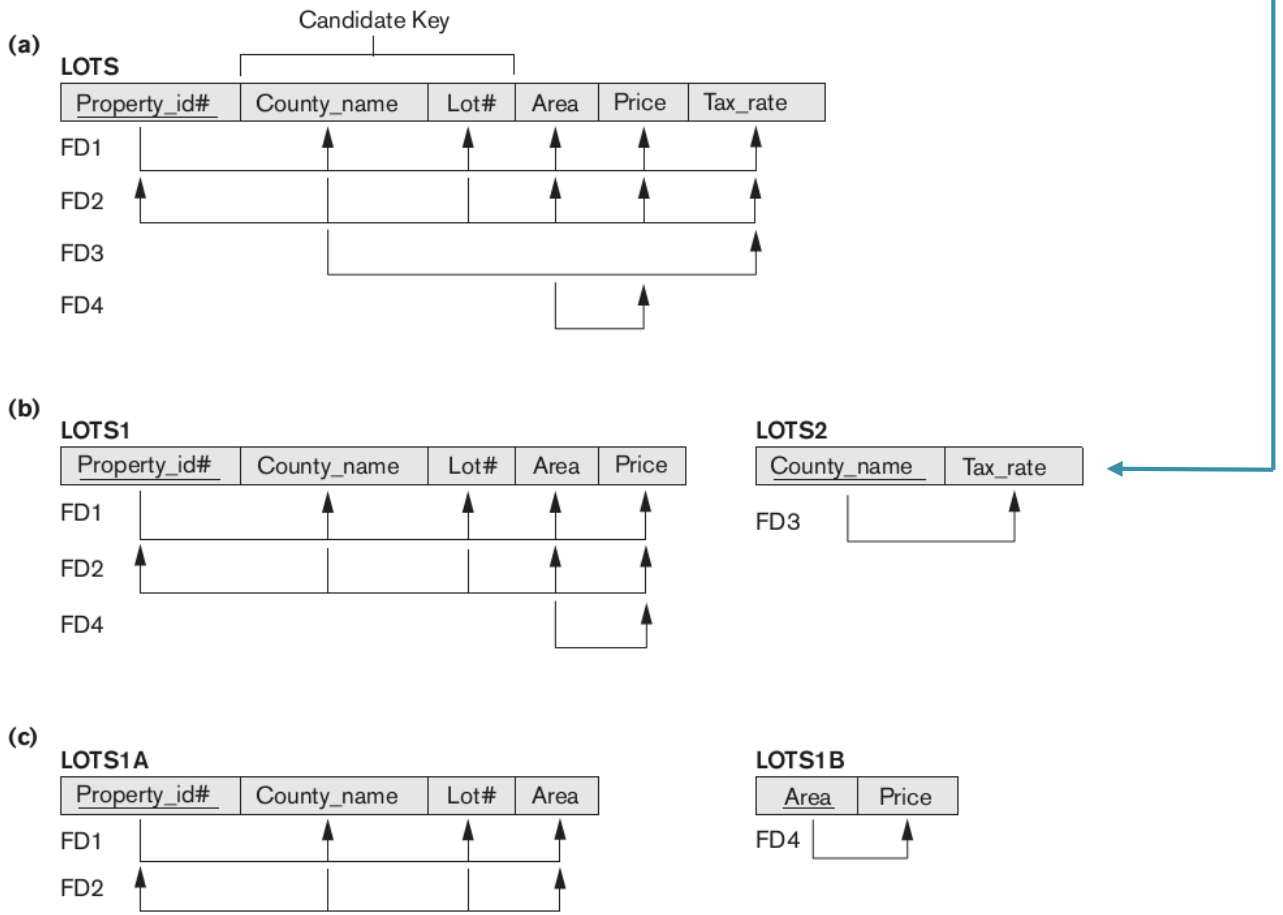
- Consider the relation LOTS (shown below) which describes land parcels for sale in different counties of a state.



- Suppose that there are two candidate keys: Property_id# and {County_name, Lot#}, that is lot numbers are unique only within each county, but Property_id# numbers are unique across counties for the entire state.
- Assume Property_id# as the primary key (but no special consideration is given to this key over the other candidate key).
- FD1 and FD2 are based on Property_id# and {County_name, Lot#}.

- FD3: County_name → Tax_rate – means tax rate is fixed for a given county (does not vary by lot within the same county).
- FD4: Area → Price – means price of a lot is determined by its area regardless of which county it is in.

- LOTS violates 2NF because Tax_rate is partially dependent on the candidate key {County_name, Lot#} due to FD3. To normalise LOTS into 2NF, we decompose it into LOTS1 and LOTS2.



Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B.

- As per the definition of the second normal form "A relation schema *R* is in 2NF if every nonprime attribute *A* in *R* is fully functionally dependent on the primary key of *R*."

- However, there is "another general definition" of Second Normal Form:

"A relation schema *R* is in 2NF if every nonprime attribute *A* in *R* is not partially dependent on *any* key of *R*."

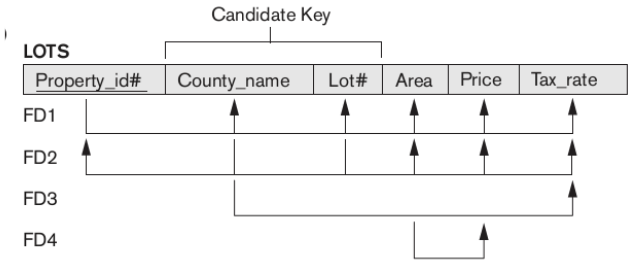
OR

"A relation schema *R* is in 2NF if every nonprime attribute *A* in *R* is fully functionally dependent on *every* key of *R*."

- Having stated this, it is now important to highlight that "The test for 2NF involves testing for functional dependencies whose left-hand side attributes are part of the primary key. If the primary key contains a single attribute, the test need not be applied at all".

- Suppose that there are two candidate keys: Property_id# and {County_name, Lot#}, that is lot numbers are unique only within each county, but Property_id# numbers are unique across counties for the entire state.

- Based on the two candidate keys Property_id and {Country_name, Lot#}, FD1 and FD2 hold (see Figure below).



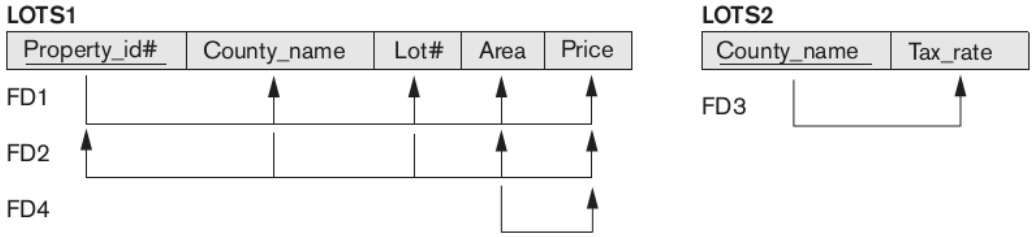
- Property_id# is chosen as primary key, so it has been underlined. However, no special consideration has been given to this key (in the above example) over the other "candidate key". Therefore, we are looking at the "candidate key" for deciding whether the relation is in 2NF. This is considered as a special case - as an example.

- Having said that, for all practical purposes, please restrict to the earlier definition "A table is in 2NF if every non-prime attribute *A* in *R* is fully functionally dependent on the primary key of *R*."

Third normal form definition – a variant

- A relation schema *R* is in 3NF if, whenever a nontrivial functional dependency $X \rightarrow A$ holds in *R*, either
  - (a) *X* is a superkey of *R*, or
  - (b) *A* is a prime attribute * of *R*.

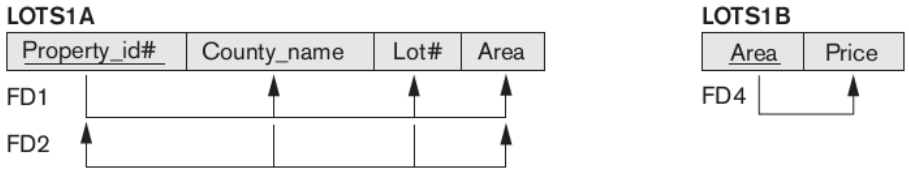
- Example, LOTS2 is in 3NF.



- FD4 in LOTS1 violates 3NF because Area is not a superkey and Price is not a prime attribute.
- So, we decompose it into LOTS1A and LOTS1B.

* An attribute of relation schema *R* is called a *prime attribute* of *R* if it is a member of *some candidate key* of *R*.

Third normal form definition – a variant



- We construct LOTS1A by removing Price that violates 3NF from LOTS1 and place it with Area (that causes transitive dependency). Now LOTS1A and LOTS1B are in 3NF.

- The first condition – "(a) *X* is a superkey of *R*" "catches" two types of problematic dependencies:
  - A nonprime attribute determines another nonprime attribute. Here we have a "transitive dependency" that violates 3NF.
  - A proper subset of a key of *R* functionally determines a nonprime attribute. Here we have a "partial dependency" that violates 2NF.
- Thus (a) takes care of the problematic dependencies that were causes for second and third normalisation.

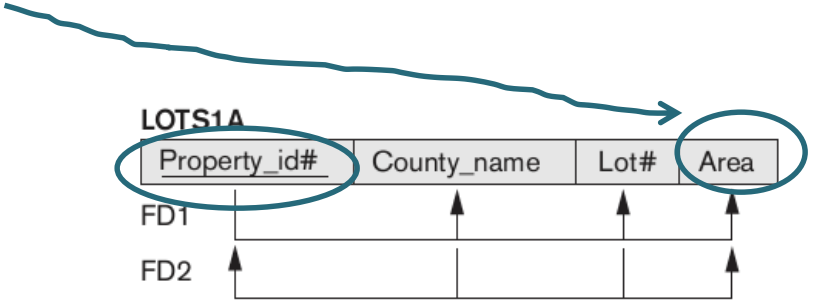
Third normal form definition – a variant

Definition: A relation schema *R* is in 3NF if every nonprime attribute of *R* meets the following conditions:

- It is fully functionally dependent on every key of *R*.
- It is nontransitively dependent on every key of *R*.

- Clause (b) in 3NF ((b) *A* is a prime attribute of *R*) allows certain FDs to slip through or escape while considering 3NF and are therefore "not caught" by the 3NF definition.

- Example, In FD1: "Area is a prime attribute of *R*" is skipped. In fact, Area is not a prime attribute of *R*.



- Hence, the Boyce-Codd normal form was proposed that "catches" these dependencies.

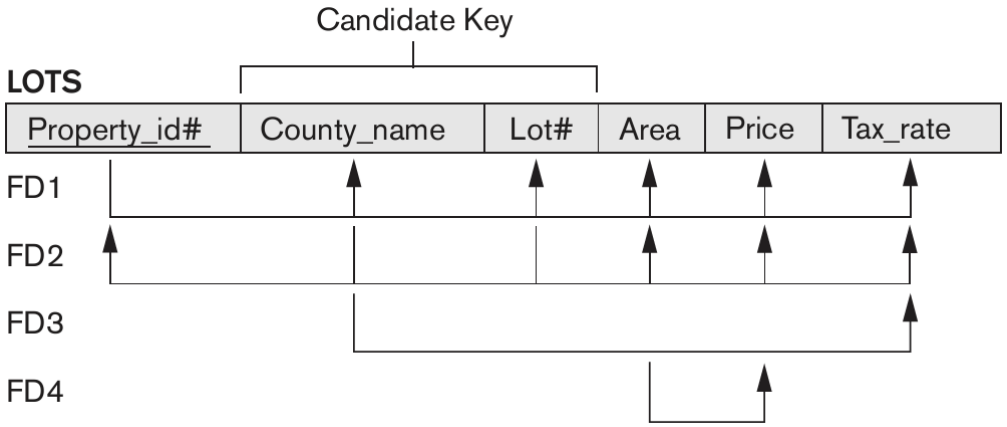
Boyce-Codd normal form

- Boyce-Codd normal form (BCNF) is a simpler form of 3NF, but is actually stricter than 3NF.

- Therefor, every relation in BCNF is also in 3NF, however, a relation in 3NF is not necessarily in BCNF.

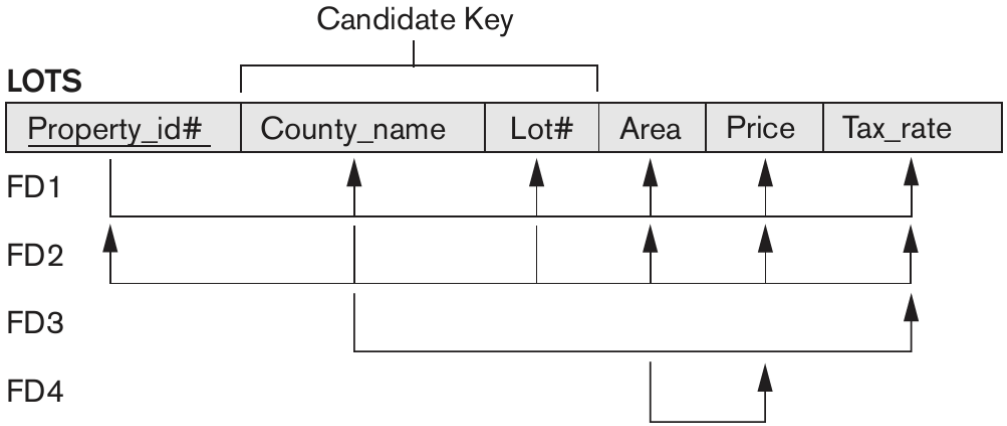
- So, although 3NF allows FDs that conform to the clause (b) in the 3NF definition, BCNF disallows them and is a stricter definition of a normal form.

- Example:



Boyce-Codd normal form

- Example,



- LOTS describes parcels of land for sale in various counties of a state. There are two candidate keys: Property_id# and {County_name, Lot#}; that is lot numbers are unique only within each county, but Property_id# numbers are unique across counties for the entire state.

- Based on the two candidate keys Property_id# and {County_name, Lot#}, FD1 and FD2 hold.

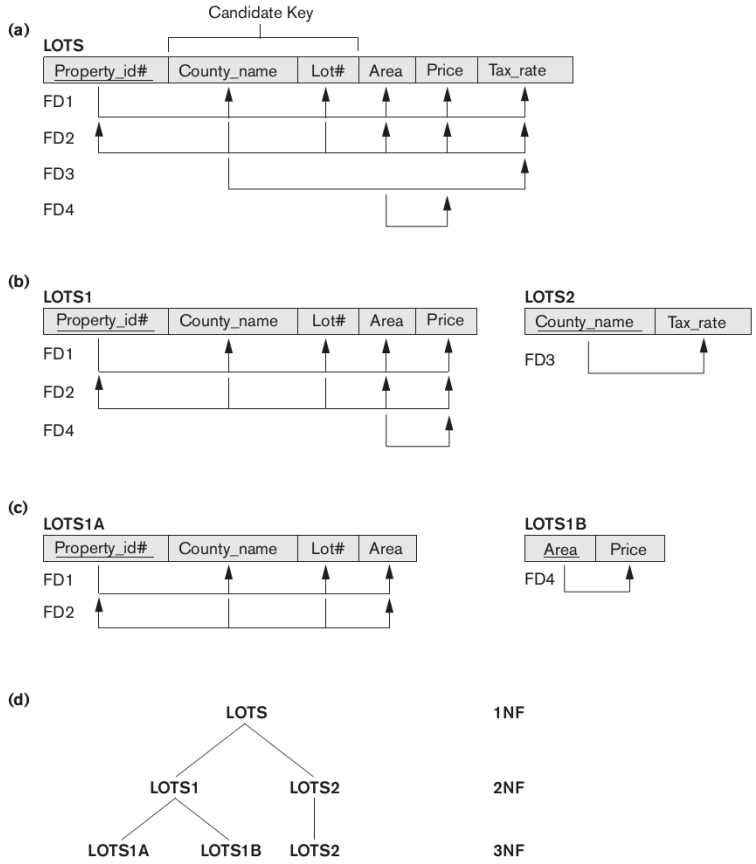
- Property_id# is the primary key.

- Consider
  - FD3: County_name → Tax_rate
  - FD4: Area → Price

- FD3 says that the tax rate is fixed for a given county (does not vary lot by lot within the same county), whereas FD4 says that the price of a lot is determined by its area regardless of which county it is in.

- LOTS relation schema violates the general definition of 2NF because Tax_rate is partially dependent on the candidate key {County_name, Lot#}, due to FD3.

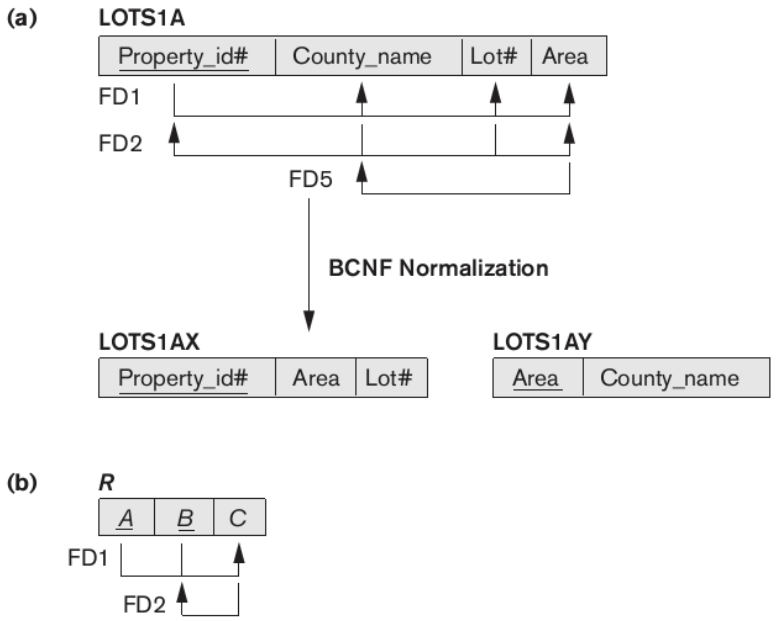
- Now, suppose that we have thousands of lots in the relation but the lots are from only two counties: "A" and "B." Suppose also that lot sizes in A County are only 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0 acres, whereas lot sizes in B County are restricted to 1.1, 1.2, …, 1.9 and 2 acres.
- So, we have the additional FD FD5: Area → County_name.



Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Progressive normalization of LOTS into a 3NF design.

- If we add FD5 to other FDs, LOTS1A still is in 3NF because this FD conforms to clause (b), County_name being a prime attribute. [Clause (b) - A relation schema *R* is in 3NF if, whenever a nontrivial functional dependency $X \rightarrow A$ holds in *R* when (b) *A* is a prime attribute of *R*.]

- The area of a lot that determines the county (as in FD5), can be represented by 16 tuples in a separate relation *R(Area, County_name)*, since there are only 16 possible Area values (0.5, 0.6, 0.7, 0.8, 0.9, 1.0,1.1, 1.2, …, 1.9 and 2 acres).



Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. $C \rightarrow B$.

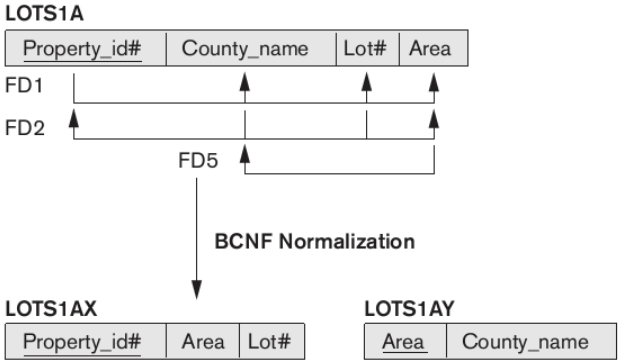
- This reduces the redundancy of repeating the same information in the thousands of LOTS1A tuples.

- BCNF is a stronger normal form that would disallow LOTS1A and suggest the need for decomposing it.

- Definition – A relation schema *R* is in BCNF if whenever a nontrivial FD $X \rightarrow A$ holds in *R*, then *X* is a superkey of *R*.

- Conclusion – BCNF differs from the definition of 3NF in clause (b) of 3NF, which allows FDs having the RHS as a prime attribute, is absent from BCNF. This makes BCNF stronger than 3NF.

- FD5 violated BCNF in LOTS1A because "Area is not a superkey" of LOTS1A. So, we can decompose LOTS1A into two BCNF relations LOTS1AX and LOTS1AY.



- This decomposition loses FD2 because its attributes no longer coexist in the same relation after decomposition.

Caution

- Ideally, relational DB design should strive to achieve BCNF or 3NF for every relation.
- Achieving 1NF or 2NF is not adequate.

Decomposition of relations not in BCNF

- Consider relation TEACH with two dependencies:

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

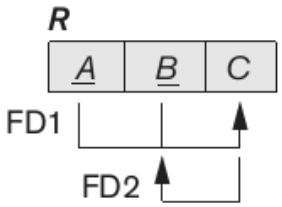
A relation TEACH that is in 3NF but not BCNF.

- FD1: {Student, Course} → Instructor
- FD2: Instructor → Course

Note that {Student, Course} is a candidate key for this relation.

- The dependency
  - FD1: {Student, Course} → Instructor
  - FD2: Instructor → Course

  follow this pattern:



TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

with Student as *A*, Course as *B* and Instructor as *C*. Hence this relation is in 3NF but not in BCNF.

- It may be decomposed into one of the three following possible pairs:

1. R1 (Student, Instructor) and R2(Student, Course)
2. R1 (Course, Instructor) and R2(Course, Student)
3. R1 (Instructor, Course) and R2(Instructor, Student)

- All the three decompositions lose FD1.

So, which of the three is a desirable decomposition?

- We generally strive to meet "two properties" of decomposition during normalisation (See slide 34):
  1. The nonadditive join property
  2. The functional dependency property.

- We must meet the nonadditive join property which can be tested using the binary decomposition of a relation through NJB (Nonadditive Join Test for Binary Decomposition)*.

- NJB confirms that only third decomposition meets the test. Therefore, the proper decomposition of TEACH into BCNF is

TEACH1(Instructor, Course) and TEACH2 (Instructor, Student)

* Please refer any standard text book for more details.

Multivalued dependency and fourth normal form

- Consider the relation EMP, with a tuple representing the fact that an employee Ename works on Pname and has dependent Dname.

EMP

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

- An employee may work on several projects and may have several dependents which are independent of each other and are multivalued.

- To avoid any spurious relationship between projects and dependents, we must have a separate tuple to represent every combination of an employee's dependent and an employee's project.

EMP

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

- Example, Smith works on projects X and Y and has dependents "John" and "Anna"; therefore has 4 tuples.
- EMP is an **all-key relation** that is with "key made up of all attributes" and therefore has no FDs and as such qualifies to be a BCNF relation.

- But there is redundancy is the relation, that is, the dependent information is repeated for every project and the project information is repeated for every dependent.

- To address this, the concept of *multivalued dependency* (MVD) is introduced on which the *fourth normal form* is defined.

- So, whenever two independent 1:N relationships A:B and A:C are mixed in the same relation, *R(A, B, C)*, a MVD may exist.

- Definition – A multivalued dependency $X \rightarrow Y$ 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 with the following properties, where Z is $(R-(X \cup Y))$:

- $t_3[X] = t_4[X] = t_1[X] = t_2[X]$
- $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$
- $t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$

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

- Definition – A relation schema is in 4NF with respect to a set of dependencies F (that includes FDs and MVDs) if, for every nontrivial MVD $X \twoheadrightarrow Y$ in F, X is a superkey for R.

- So,
  - An **all-key relation** is always in BCNF since it has no FDs.
  - 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.
- The process of normalising a relation involving the nontrivial MVDs that is not in 4NF consists of decomposing it so that each MVD is represented by a separate relation where it becomes a trivial MVD.

Summary

- We saw some of the measures for indicating whether a relation schema is good or bad, and provided informal guidelines for a good design.
- Proper enforcement of these guidelines and removing redundancy will avoid insertion/deletion/update anomalies and generation of spurious data.
- We should limit NULL values which cause problems during SELECT, JOIN and aggregation.
- We saw functional dependency which specifies semantic constraints among the attributes of a relation schema.

- We saw 1NF, 2NF, 3NF, Boyce-Codd normal form and 4NF.
- We studied multivalued dependency (MVD).
- In practice, most designs follow the normal forms up to BCNF.

Practice questions (check yourself)

1. What are the design guidelines for relation schema?
2. Explain functional dependency.
3. What is normalisation of relations?
4. Discuss first, second and third normal forms.
5. Discuss BCNF and fourth normal form.