RAMAIAH
Institute of Technology

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Database Systems
Course Code: CS52
Credits: 3:1:0
UNIT 4
Term: Oct 2021-Feb 2022

Faculty:
Sini Anna Alex

RAMAIAH
Institute of Technology

Chapter Outline

1. **Informal Design Guidelines for Relational Databases.**
2. **Basics of Functional Dependencies**
3. **Inference Rules**
4. **Normalization for Relational Databases**

RAMAIAH
Institute of Technology

Informal Design Guidelines for Relational Databases

What is relational database design?

- The grouping of attributes to form "good" relation schemas

Two levels of relation schemas

- The logical "user view" level
- The storage "base relation" level

Design is concerned mainly with base relations

What are the criteria for "good" base relations?

RAMAIAH
Institute of Technology

Informal Design Guidelines for Relational Databases

We first discuss informal guidelines for good relational design

Then we discuss formal concepts of functional dependencies and normal forms

- - 1NF (First Normal Form)
- - 2NF (Second Normal Form)
- - 3NF (Third Normal Form)
- - BCNF (Boyce-Codd Normal Form)

Additional types of dependencies, further normal forms, relational design algorithms by synthesis will be discussing later

Informal Design Guidelines for Relational Databases

- 1.1 Semantics of the Relation Attributes
- 1.2 Redundant Information in Tuples and Update Anomalies
- 1.3 Null Values in Tuples
- 1.4 Spurious Tuples

RAMAIAH
Institute of Technology

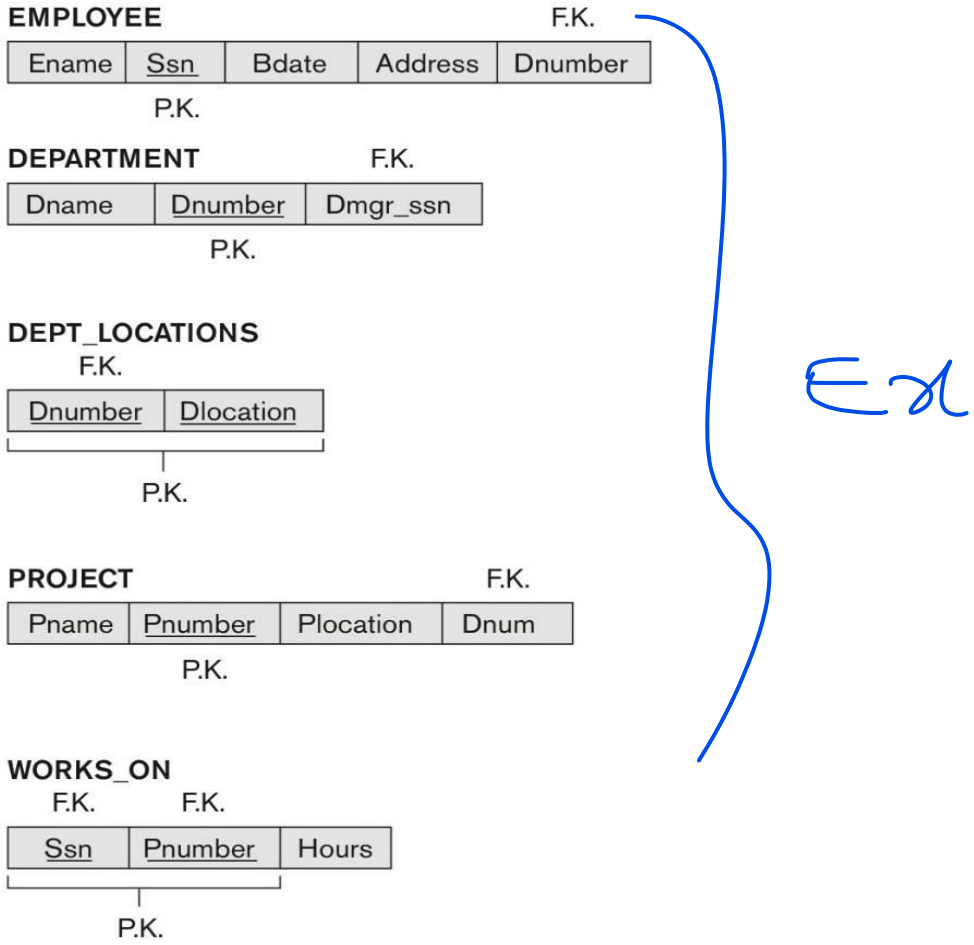
1.1 Semantics of the Relational Attributes must be clear

GUIDELINE 1: Informally, each tuple in a relation should represent one entity or relationship instance. (Applies to individual relations and their attributes).

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

Bottom Line: *Design a schema that can be explained easily relation by relation. The semantics of attributes should be easy to interpret.*

A simplified COMPANY relational database schema



Ex

RAMAIAH
Institute of Technology

1.2 Redundant Information in Tuples and Update Anomalies

- Information is stored redundantly
- Causes problems with update anomalies
- Wastes storage
  - Insertion anomalies
  - Deletion anomalies
  - Modification anomalies

RAMAIAH
Institute of Technology

EXAMPLE OF AN UPDATE ANOMALY

Consider the relation:

- EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)

Update Anomaly:

- Changing the name of project number P1 from “Billing” to “Customer-Accounting” may cause this update to be made for all 100 employees working on project P1.

Update Anomaly: Let say we have 10 columns in a table out of which 2 are called employee Name and employee address. Now if one employee changes it’s location then we would have to update the table. But the problem is, if the table is not normalized one employee can have multiple entries and while updating all of those entries one of them might get missed.

RAMAIAH
Institute of Technology

EXAMPLE OF AN INSERT ANOMALY

Consider the relation:

- EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)

Insert Anomaly:

- Cannot insert a project unless an employee is assigned to it.

Conversely

- Cannot insert an employee unless an he/she is assigned to a project.

RAMAIAH
Institute of Technology

EXAMPLE OF AN DELETE ANOMALY

Consider the relation:

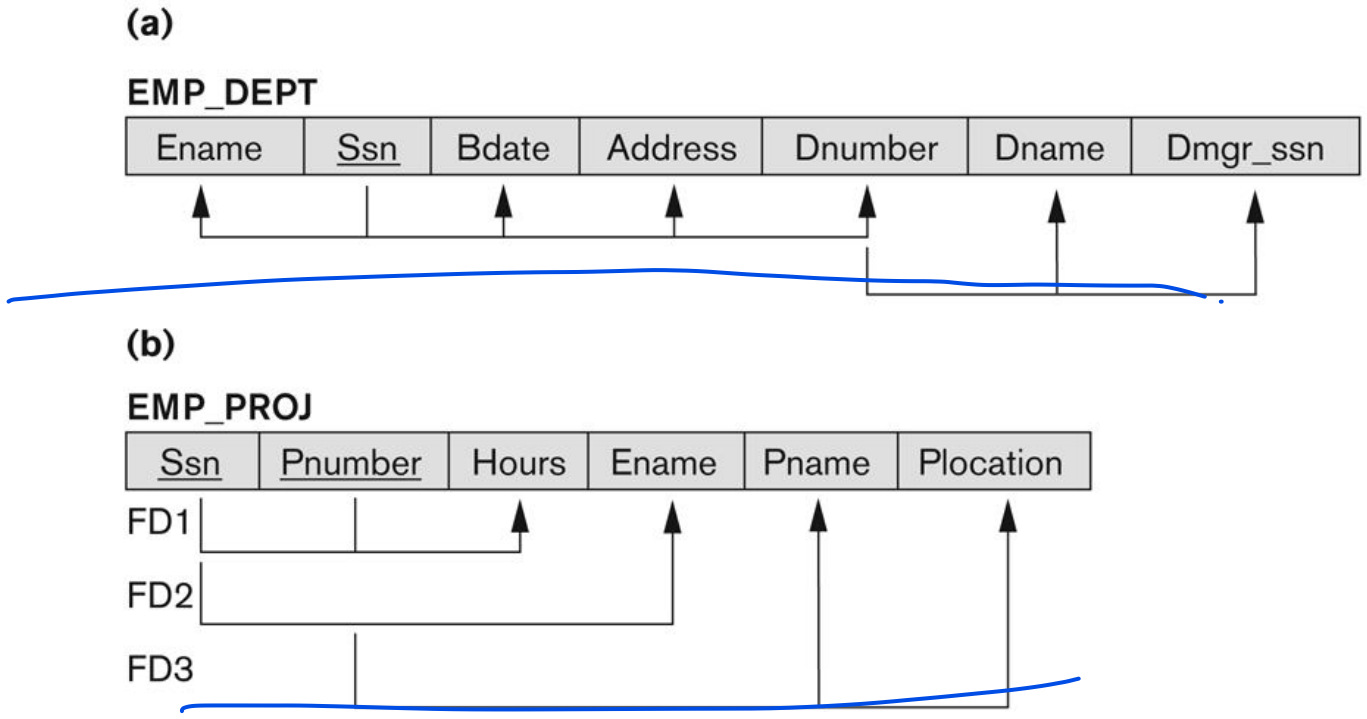
- EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)

Delete Anomaly:

- When a project is deleted, it will result in deleting all the employees who work on that project.
- Alternately, if an employee is the sole employee on a project, deleting that employee would result in deleting the corresponding project.
- **Deletion Anomaly**: This anomaly indicates unnecessary deletion of important information from the table. Let's say we have student's information and courses they have taken as follows (student ID,Student Name, Course, address). If any student leaves the school then the entry related to that student will be deleted. However, that deletion will also delete the course information even though course depends upon the school and not the student.

Figure : Two relation schemas suffering from update anomalies

Figure 10.3
Two relation schemas suffering from update anomalies.
(a) EMP_DEPT and
(b) EMP_PROJ.



RAMAIAH
Institute of Technology

Guideline to Redundant Information in Tuples and Update Anomalies

GUIDELINE 2:

- Design a schema that does not suffer from the insertion, deletion and update anomalies.
- If there are any anomalies present, then note them so that applications can be made to take them into account.

RAMAIAH
Institute of Technology

Null Values in Tuples

GUIDELINE 3:

- Relations should be designed such that their tuples will have as few NULL values as possible
- Attributes that are NULL frequently could be placed in separate relations (with the primary key)

Reasons for nulls:

- Attribute not applicable or invalid
- Attribute value unknown (may exist)
- Value known to exist, but unavailable

RAMAIAH
Institute of Technology

Spurious Tuples

Bad designs for a relational database may result in erroneous results for certain JOIN operations

The "lossless join" property is used to guarantee meaningful results for join operations.

A "**tuple**" is a record in a database: a row in a spreadsheet.

A **spurious tuple** is, basically, a record in a database that gets created when two tables are joined badly. In database, **spurious tuples** are created when two tables are joined on attributes that are neither primary keys nor foreign keys.

GUIDELINE 4:

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

RAMAIAH
Institute of Technology

Spurious Tuples contd..

There are two important properties of decompositions:

a) Non-additive or losslessness of the corresponding join
b) Preservation of the functional dependencies.

Note that:

- Property (a) is extremely important and *cannot* be sacrificed.
- Property (b) is less stringent and may be sacrificed.

Guidelines - summary

GUIDELINE 1: Design a schema that can be explained easily relation by relation. The semantics of attributes should be easy to interpret.

GUIDELINE 2: Design a schema that does not suffer from the insertion, deletion and update anomalies. If there are any present, then note them so that applications can be made to take them into account

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

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

RAMAIAH
Institute of Technology

Not all designs are equally good

Why is this design bad?

```
Data(sid,sname,address,cid,cname,grade)
```

Why is this one preferable?

```
Student(sid,sname,address)
Course(cid,cname)
Enrolled(sid,cid,grade)
```

An instance of our bad design

sid	sname	address	cid	cname	grade
124	Britney	USA	206	Database	A++
204	Victoria	Essex	202	Semantics	C
124	Britney	USA	201	S/Eng I	A+
206	Emma	London	206	Database	B-
124	Britney	USA	202	Semantics	B+

Recall:

A key is a set of fields where if a pair of tuples agree on a key, they agree everywhere.

In our bad design, if two tuples agree on `sid`, then they also agree on `address`, even though the rest of the tuples may not agree

RAMAIAH
Institute of Technology

Functional Dependencies

Functional dependencies (FDs)

- Are used to specify *formal measures* of the "goodness" of relational designs
- And keys are used to define **normal forms** for relations
- Are **constraints** that are derived from the *meaning* and *interrelationships* of the data attributes

A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y

Functional Dependencies

X -> Y holds if whenever two tuples have the same value for X, they *must have* the same value for Y

- For any two tuples t1 and t2 in any relation instance r(R): If t1[X]=t2[X], *then* t1[Y]=t2[Y]

X -> Y in R specifies a *constraint* on all relation instances r(R)

Written as X -> Y; can be displayed graphically on a relation schema. (denoted by the arrow:).

FDs are derived from the real-world constraints on the attributes

RAMAIAH
Institute of Technology

Figure: What FDs may exist?

A relation *R*(A, B, C, D) with its extension.

Which FDs *may exist* in this relation?

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

The following FDs may hold B->C, C->B
The following FDs do not hold A->B, B->A, D->C
We denote by F the set of functional dependencies specified on relation schema R.

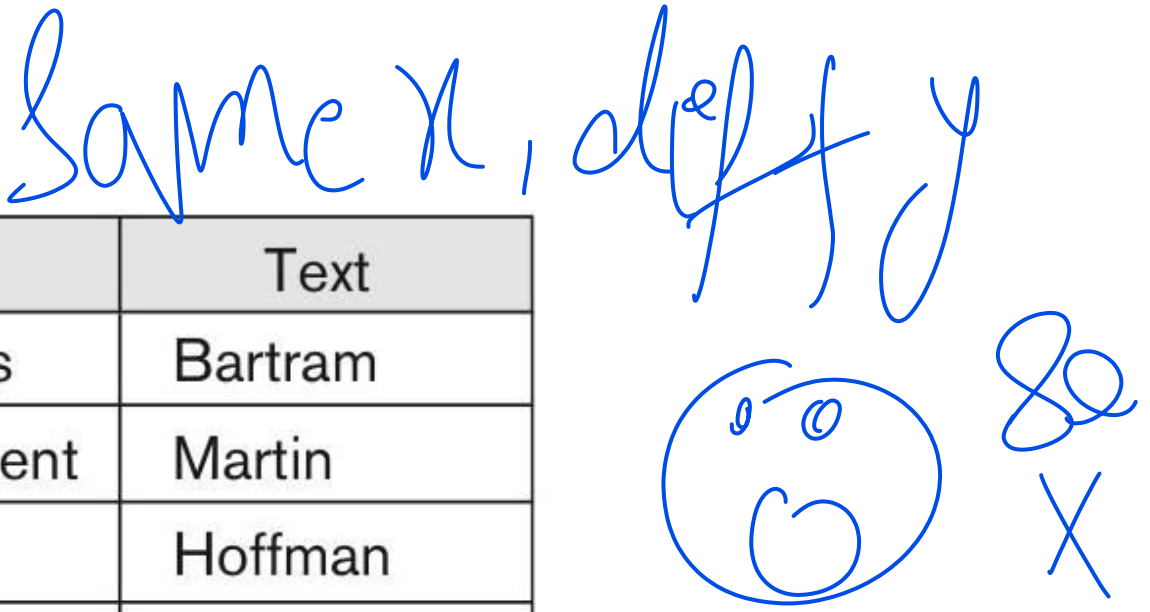
RAMAIAH
Institute of Technology

Ruling Out FDs

Note that given the state of the TEACH relation, The FDs Teacher → Course, Teacher → Text and Couse → Text are ruled out.

TEACH

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



RAMAIAH
Institute of Technology

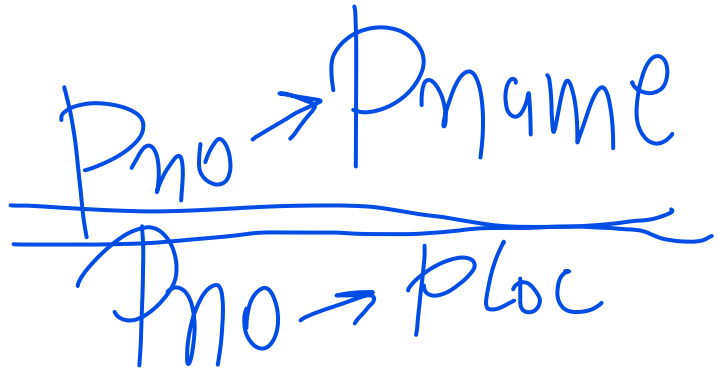
Examples of FD constraints

Social security number determines employee name

- SSN -> ENAME

Project number determines project name and location

- PNUMBER -> {PNAME, PLOCATION}



Employee ssn and project number determines the hours per week that the employee works on the project

- {SSN, PNUMBER} -> HOURS

FD constraints

An FD is a property of the attributes in the schema R

The constraint must hold on *every* relation instance r(R)

If K is a key of R, then K functionally determines all attributes in R

- (since we never have two distinct tuples with t1[K]=t2[K])

RAMAIAH
Institute of Technology

Inference Rules for FDs

Given a set of FDs F, we can **infer** additional FDs that hold whenever the FDs in F hold

Armstrong's inference rules:

- IR1. (**Reflexive**) If Y *subset-of* X, then X -> Y
- IR2. (**Augmentation**) If X -> Y, then XZ -> YZ
  - (Notation: XZ stands for X U Z)
- IR3. (**Transitive**) If X -> Y and Y -> Z, then X -> Z

IR1, IR2, IR3 form a **sound** and **complete** set of inference rules

- These are rules hold and all other rules that hold can be deduced from these

RAMAIAH
Institute of Technology

Inference Rules for FDs

Some additional inference rules that are useful:

- **Decomposition:** If X -> YZ, then X -> Y and X -> Z
- **Union:** If X -> Y and X -> Z, then X -> YZ
- **Psuedotransitivity:** If X -> Y and WY -> Z, then WX -> Z

The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property)

Thank you

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Database Systems

Course Code: CS52

Credits: 3:1:0

UNIT 4

Term: Oct 2021 – Feb 2022

Faculty:
Sini Anna Alex

RAMAIAH
Institute of Technology

Inference Rules for FDs

Given a set of FDs F, we can **infer** additional FDs that hold whenever the FDs in F hold

Armstrong's inference rules:

- IR1. (**Reflexive**) If Y *subset-of* X, then X -> Y
- IR2. (**Augmentation**) If X -> Y, then XZ -> YZ
  - (Notation: XZ stands for X U Z)
- IR3. (**Transitive**) If X -> Y and Y -> Z, then X -> Z

IR1, IR2, IR3 form a **sound** and **complete** set of inference rules

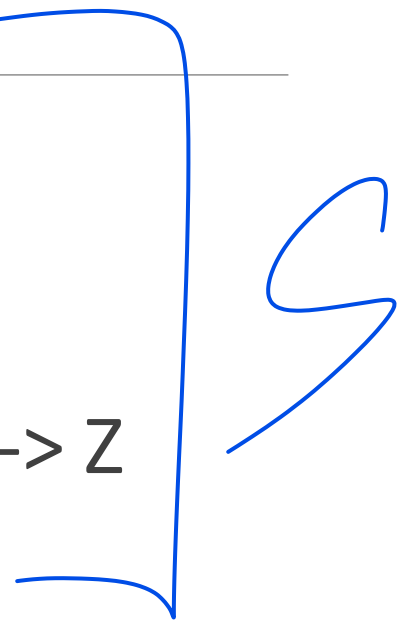
- These are rules hold and all other rules that hold can be deduced from these

RAMAIAH
Institute of Technology

Inference Rules for FDs

Some additional inference rules that are useful:

- **Decomposition:** If X -> YZ, then X -> Y and X -> Z
- **Union:** If X -> Y and X -> Z, then X -> YZ
- **Psuedotransitivity:** If X -> Y and WY -> Z, then WX -> Z



The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property)

RAMAIAH
Institute of Technology

Proof of Inference Rules

Proof of IR1. Suppose that $X \supseteq Y$ and that two tuples t_1 and t_2 exist in some relation instance r of R such that $t_1[X] = t_2[X]$. Then $t_1[Y] = t_2[Y]$ because $X \supseteq Y$; hence, $X \rightarrow Y$ must hold in r.

Proof of IR2 (by contradiction). Assume that $X \rightarrow Y$ holds in a relation instance r of R but that $XZ \rightarrow YZ$ does not hold. Then there must exist two tuples t_1 and t_2 in r such that (1) $t_1[X] = t_2[X]$, (2) $t_1[Y] = t_2[Y]$, (3) $t_1[XZ] = t_2[XZ]$, and (4) $t_1[YZ] \neq t_2[YZ]$. This is not possible because from (1) and (3) we deduce (5) $t_1[Z] = t_2[Z]$, and from (2) and (5) we deduce (6) $t_1[YZ] = t_2[YZ]$, contradicting (4).

Proof of IR3. Assume that (1) $X \rightarrow Y$ and (2) $Y \rightarrow Z$ both hold in a relation r. Then for any two tuples t_1 and t_2 in r such that $t_1[X] = t_2[X]$, we must have (3) $t_1[Y] = t_2[Y]$, from assumption (1); hence we must also have (4) $t_1[Z] = t_2[Z]$ from (3) and assumption (2); thus $X \rightarrow Z$ must hold in r.

RAMAIAH
Institute of Technology

Proof of Inference Rules

Proof of IR4 (Using IR1 through IR3).

1. $X \rightarrow YZ$ (given).
2. $YZ \rightarrow Y$ (using IR1 and knowing that $YZ \supseteq Y$).
3. $X \rightarrow Y$ (using IR3 on 1 and 2).

Proof of IR5 (using IR1 through IR3).

1. $X \rightarrow Y$ (given).
2. $X \rightarrow Z$ (given).
3. $X \rightarrow XY$ (using IR2 on 1 by augmenting with X; notice that $XX = X$).
4. $XY \rightarrow YZ$ (using IR2 on 2 by augmenting with Y).
5. $X \rightarrow YZ$ (using IR3 on 3 and 4).

Proof of IR6 (using IR1 through IR3).

1. $X \rightarrow Y$ (given).
2. $WY \rightarrow Z$ (given).
3. $WX \rightarrow WY$ (using IR2 on 1 by augmenting with W).
4. $WX \rightarrow Z$ (using IR3 on 3 and 2).

RAMAIAH
Institute of Technology

Problem on Inference Rules:

1. Prove or disprove the following inference rules for functional dependencies. A proof can be made by using inference rules R1 through R3. A disproof should be performed by demonstrating a relation that satisfies the conditions and functional dependencies in the left-hand side of the inference rule but does not satisfy the dependencies in the right-hand side.

Prove:

a. {W ⇨ Y, X ⇨ Z} |= WX ⇨ Y

b. {X ⇨ Y, Z is a subset of Y} |= X ⇨ Z

c. {X ⇨ Y, X ⇨ W, WY ⇨ Z} |= X ⇨ Z

d. {X ⇨ Y, XY ⇨ Z} |= X ⇨ Z

e. {X ⇨ Y, Z ⇨ W} |= XZ ⇨ YZ

f. {X ⇨ Y, Y ⇨ Z} |= X ⇨ YZ

Disprove:

g. {XY ⇨ Z, Y ⇨ W} |= XW ⇨ Z

RAMAIAH
Institute of Technology

Closure of a set F of FDs

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

Algorithm 10.1: Determining X^+, the Closure of X under F

$X^+ := X$;
repeat
    $oldX^+ := X^+$;
    for each functional dependency $Y \rightarrow Z$ in F do
    if $X^+ \supseteq Y$ then $X^+ := X^+ \cup Z$;
until $(X^+ = oldX^+)$;

RAMAIAH
Institute of Technology

The following set F of functional dependencies that should hold on EMP_PROJ;

$$F = \{SSN \longrightarrow ENAME,$$
$$PNUMBER \longrightarrow \{PNAME, PLOCATION\},$$
$$\{SSN, PNUMBER\} \longrightarrow HOURS\}$$

we calculate the following closure sets with respect to F;

$$\{SSN\}^+ = \{SSN, ENAME\}$$
$$\{PNUMBER\}^+ = \{PNUMBER, PNAME, PLOCATION\}$$
$$\{SSN, PNUMBER\}^+ = \{SSN, PNUMBER, ENAME, PNAME, PLOCATION, HOURS\}$$

RAMAIAH
Institute of Technology

Problem:

Consider the following relation schema and set of functional dependancies:

Emp-Dept (SIN, E_Name, B_Date, Address, D_Num, D_Name, D_Manager)

F={SIN → {E_Name, B_Date, Address, D_Num},
D_Num → {D_Name, D_Manager}
}

Calculate the closure of $\{SIN\}^+$ and $\{D\_Num\}^+$ with respect to F.

R(A,B,C,D,E,F)
FDs S = {AB→C, BC→AD, D→E, CF→B}

What is the closure of:
$\{A,B\}^+$

ABCDE

RAMAIAH
Institute of Technology

Equivalence of Sets of Functional Dependencies

Two sets of FDs F and G are **equivalent** if:

- Every FD in F can be inferred from G, and
- Every FD in G can be inferred from F
- Hence, F and G are equivalent if $F^+ = G^+$

Definition (**Covers**):

- F **covers** G if every FD in G can be inferred from F
  - (i.e., if G^+ *subset-of* F^+)

F and G are equivalent if F covers G and G covers F.

RAMAIAH
Institute of Technology

Equivalence of Sets of Functional Dependencies

Definition. A set of functional dependencies F is said to **cover** another set of functional dependencies E if every FD in E is also in F^+; that is, if every dependency in E can be inferred from F; alternatively, we can say that E is **covered by** F.

Definition. Two sets of functional dependencies E and F are **equivalent** if $E^+ = F^+$. Hence, equivalence means that every FD in E can be inferred from F, and every FD in F can be inferred from E; that is, E is equivalent to F if both the conditions E covers F and F covers E hold.

RAMAIAH Institute of Technology

Examples:

R=(A,B,C,D,E,F)

F1={A->BC, B->CDE, AE->F}

F2={A->BCF, B->DE, E->AB}

Check whether F1 and F2 are equivalent or not.

F1+ = F2+

eqv

Solution

To check F1 covers F2 –

A^+={A,B,C,D,E,F} contains B,C,F

B^+={B,C,D,E} contains D,E

E^+={E} contains A,B

So F1 does not cover F2.

Hence F1 and F2 are not equivalent.

Consider another example where two functional dependencies are equivalent.

R=(A,C,D,E,H)

F1={A->C, AC->D, E->AD, E->H},

F2={A->CD, E->AH}

Check whether F1 and F2 are equivalent or not?

Solution

To check F1 covers F2 –

A^+={A,C,D} contains C,D

E^+={A,D,E,H} contains A,H

So F1 covers F2

To check F2 covers F1:

A^+={A,C,D} contains C

{A,C}+={A,C,D} contains D

E^+={A,C,D,E,H} contains A,D,H

So F2 covers F1.

Hence F1 and F2 are equivalent.

Minimal Sets of FDs

A set of FDs is **minimal** if it satisfies the following conditions:

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

RAMAIAH
Institute of Technology

Minimal Sets of Functional Dependencies

Algorithm 10.2: Finding a Minimal Cover F for a Set of Functional Dependencies E

1. Set $F := E$.
2. Replace each functional dependency $X \rightarrow \{A_1, A_2, \ldots, A_n\}$ in F by the n functional dependencies $X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_n$.
3. For each functional dependency $X \rightarrow A$ in F

   for each attribute B that is an element of X

   if $\{\{F - \{X \rightarrow A\}\} \cup \{(X - \{B\}) \rightarrow A\}\}$ is equivalent to F,

   then replace $X \rightarrow A$ with $(X - \{B\}) \rightarrow A$ in F.
4. For each remaining functional dependency $X \rightarrow A$ in F

   if $\{F - \{X \rightarrow A\}\}$ is equivalent to F,

   then remove $X \rightarrow A$ from F.

In Chapter 11 we will see how relations can be synthesized from a given set of dependencies E by first finding the minimal cover F for E.

RAMAIAH
Institute of Technology

Minimal Sets of FDs

- Every set of FDs has an equivalent minimal set.
- There can be several equivalent minimal sets.
- There is no simple algorithm for computing a minimal set of FDs that is equivalent to a set F of FDs.
- To synthesize a set of relations, we assume that we start with a set of dependencies that is a minimal set.

RAMAIAH
Institute of Technology

Find Minimal Cover/ Canonical Cover of E

Let the given set of FDs be E : {B → A, D → A, AB → D}.We have to find the minimum cover of E.

■ All above dependencies are in canonical form; so we have completed step 1 of Algorithm 10.2 and can proceed to step 2.

In step 2 we need to determine if AB → D has any redundant attribute on the left-hand side; that is, can it be replaced by B → D or A → D?

■ Since B → A, by augmenting with B on both sides (IR2), we have BB → AB, or B → AB

(i). However, AB → D as given (ii).

■ Hence by the transitive rule (IR3), we get from (i) and (ii), B → D. Hence AB → D may be replaced by B → D.

■ We now have a set equivalent to original E , say E' : {B → A, D → A, B → D}. No further reduction is possible in step 2 since all FDs have a single attribute on the left-hand side.

■ In step 3 we look for a redundant FD in E'. By using the transitive rule on B → D and D → A, we derive B → A. Hence B → A is redundant in E' and can be eliminated.

■ **Hence the minimum cover of E is {B → D, D → A}**

RAMAIAH
Institute of Technology

Thank you

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Database Systems

Course Code: CS52

Credits: 3:1:0

UNIT 4

Term: Oct 2021 – Feb 2022

Faculty:
Dr. Sini Anna Alex

RAMAIAH
Institute of Technology

3 Normal Forms Based on Primary Keys

Normalization of Relations

Practical Use of Normal Forms

Definitions of Keys and Attributes Participating in Keys

First Normal Form

Second Normal Form

Third Normal Form

Definitions of Keys and Attributes Participating in Keys

A **superkey** of a relation schema R = {A1, A2, ..., An} is a set of attributes S *subset-of* R with the property that no two tuples t1 and t2 in any legal relation state r of R will have t1[S] = t2[S]

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

RAMAIAH
Institute of Technology

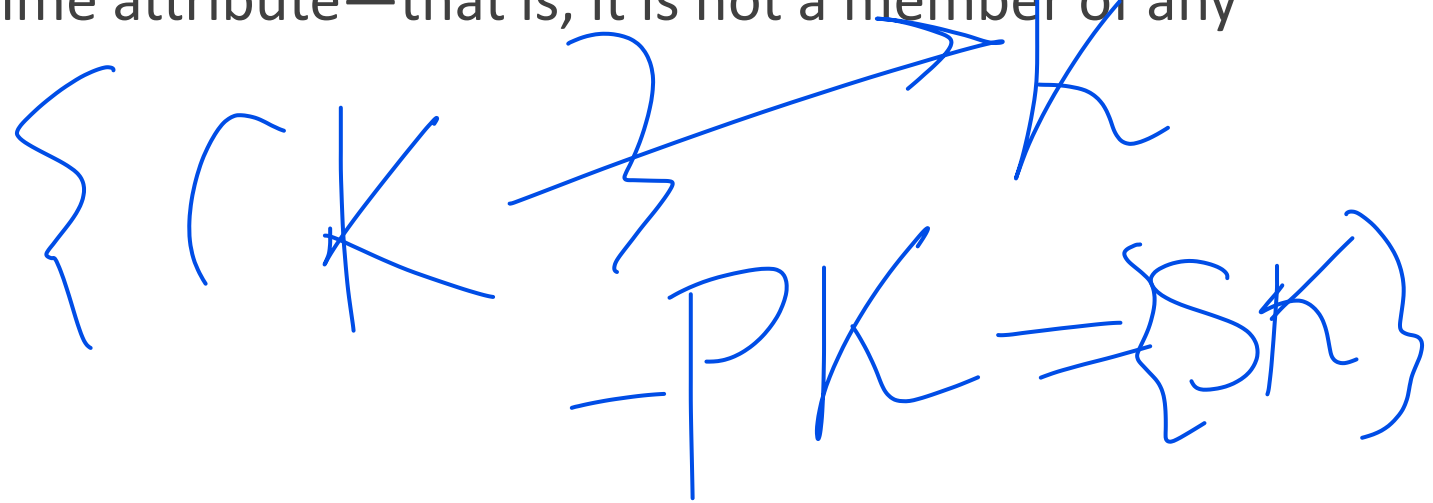
Definitions of Keys and Attributes Participating in Keys

If a relation schema has more than one key, each is called a **candidate** key.

- One of the candidate keys is *arbitrarily* designated to be the **primary key**, and the others are called **secondary keys**.

A **Prime attribute** must be a member of *some* candidate key

A **Nonprime attribute** is not a prime attribute—that is, it is not a member of any candidate key.



RAMAIAH
Institute of Technology

Finding Candidate Keys and Super Keys of a Relation using FD set

The **set of attributes** whose attribute closure is set of all attributes of relation is called super key of relation.

Consider the following FD set. **{E-ID->E-NAME, E-ID->E-CITY, E-ID->E-STATE, E-CITY->E-STATE}**

Let us calculate attribute closure of different set of attributes: **EMPLOYEE(E-ID, E-NAME, E-CITY, E-STATE)**

(E-ID)+ = {E-ID, E-NAME,E-CITY,E-STATE}

(E-ID,E-NAME)+ = {E-ID, E-NAME,E-CITY,E-STATE}

(E-ID,E-CITY)+ = {E-ID, E-NAME,E-CITY,E-STATE}

(E-ID,E-STATE)+ = {E-ID, E-NAME,E-CITY,E-STATE}

(E-ID,E-CITY,E-STATE)+ = {E-ID, E-NAME,E-CITY,E-STATE}

(E-NAME)+ = {E-NAME}

(E-CITY)+ = {E-CITY,E-STATE}

RAMAIAH
Institute of Technology

Normalization of Relations

Normalization:

- The process of decomposing unsatisfactory "bad" relations by breaking up their attributes into smaller relations.
- It is important that a database is **normalized to minimize redundancy (duplicate data) and to ensure only related data is stored in each table**.
- Normalization makes sure that all of your data looks and reads the same way across all records.

The benefits of normalization include: **Searching, sorting, and creating indexes is faster**

Normal form:

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

RAMAIAH
Institute of Technology

Normalization of Relations

2NF, 3NF, BCNF

- based on keys and FDs of a relation schema

4NF

- based on keys, multi-valued dependencies : MVDs;

5NF

- based on keys, join dependencies : JDs

Additional properties may be needed to ensure a good relational design (lossless join, dependency preservation)

RAMAIAH
Institute of Technology

Practical Use of Normal Forms

Normalization is carried out in practice so that the resulting designs are of high quality and meet the desirable properties

The practical utility of these normal forms becomes questionable when the constraints on which they are based are *hard to understand* or to *detect*

The database designers *need not* normalize to the highest possible normal form

- (usually up to 3NF and BCNF. 4NF rarely used in practice.)

Denormalization:

- The process of storing the join of higher normal form relations as a base relation—which is in a lower normal form

RAMAIAH
Institute of Technology

First Normal Form

Disallows

- composite attributes
- multivalued attributes
- **nested relations**; attributes whose values for an *individual tuple* are non-atomic

Considered to be part of the definition of relation .

Most RDBMSs allow only those relations to be defined that are in First Normal Form

RAMAIAH Institute of Technology

Normalization into 1NF

(a)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations

(b)

DEPARTMENT

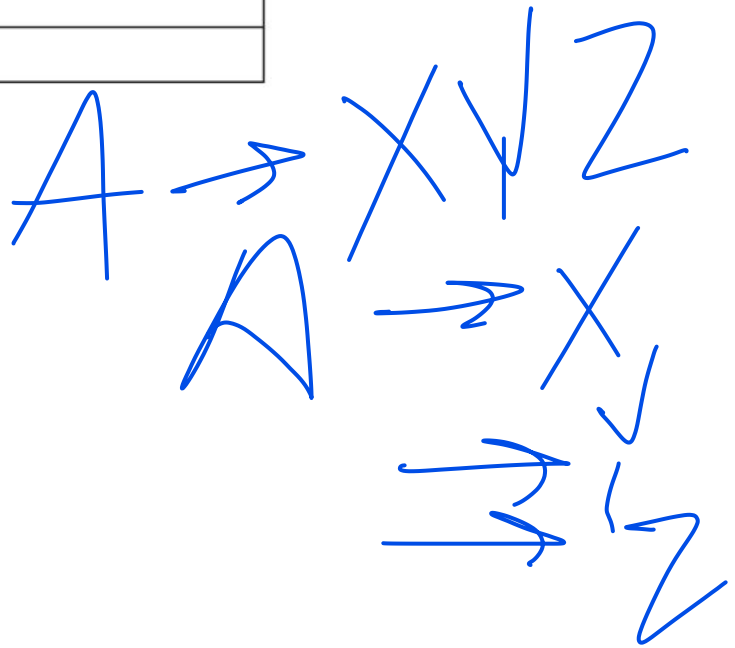
Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)

DEPARTMENT

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

Figure 10.8
Normalization into 1NF. (a) A relation schema that is not in 1NF. (b) Example state of relation DEPARTMENT. (c) 1NF version of the same relation with redundancy.



RAMAIAH
Institute of Technology

Normalization nested relations into 1NF

(a)

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours

(b)

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, AliciaJ.	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

(c)

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

Figure 10.9
Normalizing nested relations into 1NF. (a) Schema of the EMP_PROJ relation with a *nested relation* attribute PROJS. (b) Example extension of the EMP_PROJ relation showing nested relations within each tuple. (c) Decomposition of EMP_PROJ into relations EMP_PROJ1 and EMP_PROJ2 by propagating the primary key.

RAMAIAH
Institute of Technology

Second Normal Form

Second normal form (2NF) is based on the concept of *full functional dependency.* A functional dependency X → *Y* is a **full functional dependency** if removal of any attribute A from X means that the dependency does not hold any more.

Functional dependency X → *Y* is a **partial dependency** if some attribute A ϵ X can be removed from X and the dependency still 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.

RAMAIAH
Institute of Technology

Second Normal Form contd……

A relation schema R is in **second normal form (2NF)** if every non-prime attribute A in R is fully functionally dependent on the primary key

R can be decomposed into 2NF relations via the process of 2NF normalization or “second normalization”

RAMAIAH
Institute of Technology

2NF Example

Key: SSN, PNUMBER

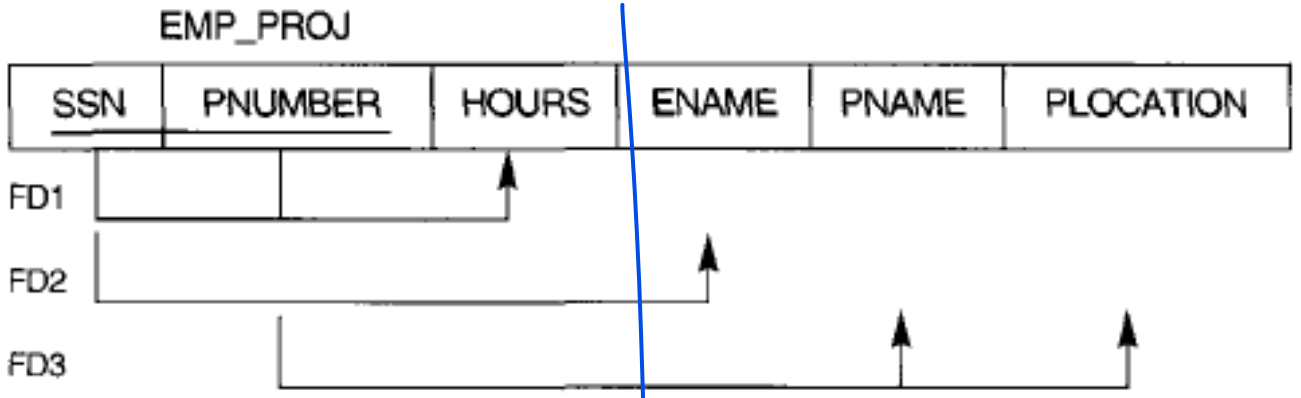
FDs:

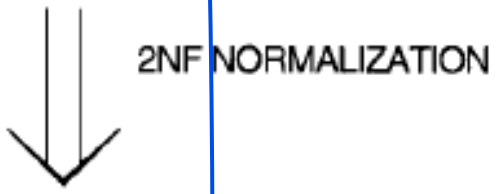
SSN, PNUMBER-> HOURS

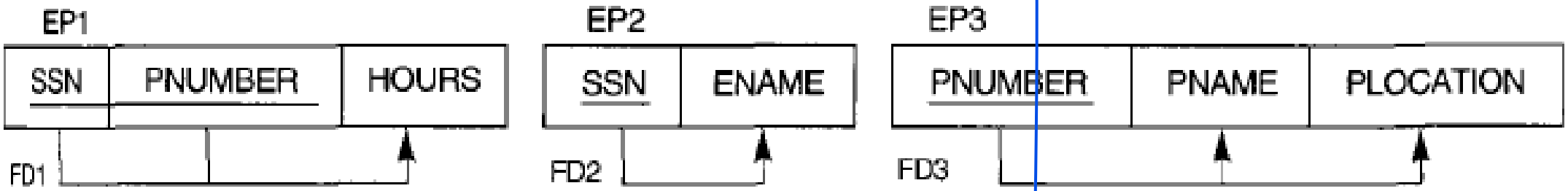
SSN->ENAME

PNUMBER->PNAME

PNUMBER->PLOCATION







RAMAIAH
Institute of Technology

Tutorial 2: Decompose a non-2NF relation to a 2NF relation

Assume a relation R (A, B, C, D, E) with the following set of functional dependencies;

F = {AB → C, B → D, E → D}

The key for this relation is ABE. Then, all three given FDs are partial dependencies, viz., AB → C, B → D, and E → D.
Step 1: separate tables for partial dependencies; hence, R1 (ABC), R2 (BD) and R3 (ED).

Step 2: remove RHS of these two partial FDs from R; hence, R4(A, B, E).

Thus, we have four tables ***R1 (ABC), R2 (BD), R3 (ED)*** and ***R4 (ABE)***.

RAMAIAH
Institute of Technology

Find the candidate keys of a relation, How to find the candidate keys, Which is the key for the given table, concept of candidate key in dbms, candidate key examples

Question:

Consider the relation R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies F = {AB → C, A → DE, B → F, F → GH, D → IJ}. Find the key of relation R.

Let R = (A, B, C, D, E, F) be a relation scheme with the following dependencies-

$$C \rightarrow F$$

$$E \rightarrow A$$

$$EC \rightarrow D$$

$$A \rightarrow B$$

Which of the following is a key for R?

RAMAIAH
Institute of Technology

Tutorial 4: Compute Minimal Cover

1. Find the minimal cover of the set of functional dependencies given;

{A → C, AB → C, C → DI, CD → I, EC → AB, EI → C}

2. F = { AB -> C, C -> A, BC -> D, ACD -> B, D -> E, D -> G, BE -> C, CG -> B, CG -> D, CE -> A, CE -> G}

Solution:

Q1: MC = {A → C, C → D, C → I, EC → A, EC → B, EI → C}

Q2:

Minimal Cover 1: {AB -> C, **C -> A**, BC -> D, **CD -> B**, D -> E, D -> G, BE -> C, **CG -> D**, CE -> G}

Minimal Cover 2 : {AB -> C, **C -> A**, **BC -> D**, D -> E, **D -> G**, BE -> C, **CG -> B**, CE -> G}

RAMAIAH
Institute of Technology

Thank you

RAMAIAH
Institute of Technology

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Database Systems

Course Code: CS52

Credits: 3:1:0

UNIT 4

Term: Oct 2021 – Feb 2022

Faculty:
Dr.Sini Anna Alex

RAMAIAH
Institute of Technology

Find the candidate keys of a relation, How to find the candidate keys, Which is the key for the given table, concept of candidate key in dbms, candidate key examples

Question:

Consider the relation R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies F = {AB → C, A → DE, B → F, F → GH, D → IJ}. Find the key of relation R.

Let R = (A, B, C, D, E, F) be a relation scheme with the following dependencies-

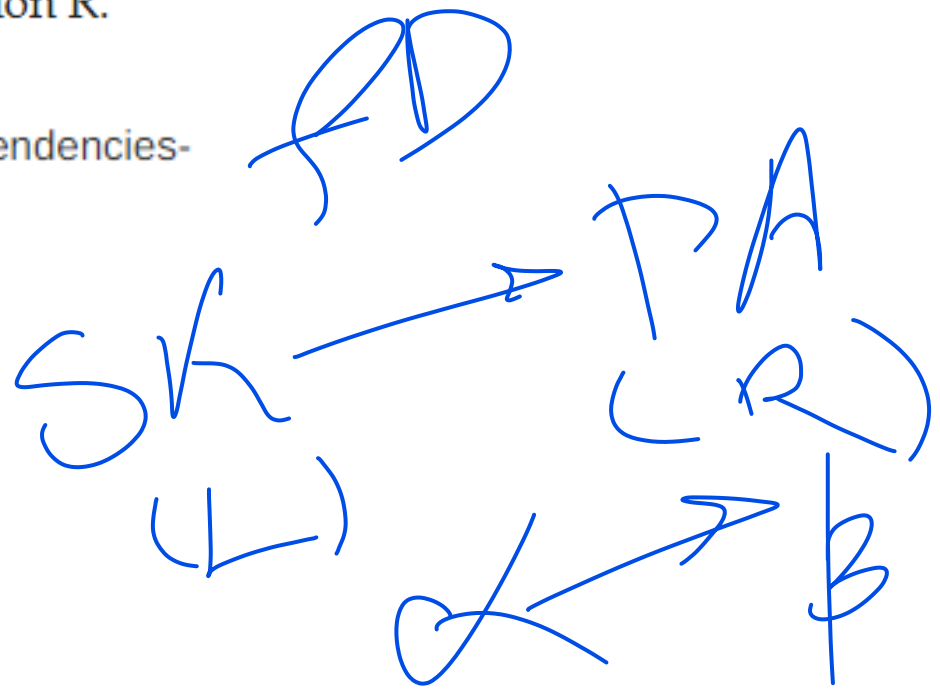
C → F

E → A

EC → D

A → B

Which of the following is a key for R?



RAMAIAH
Institute of Technology

Third Normal Form

Definition:

- **Transitive functional dependency:** a FD X -> Z that can be derived from two FDs X -> Y and Y -> Z

Examples:

- SSN -> DMGRSSN is a **transitive** FD
  - Since SSN -> DNUMBER and DNUMBER -> DMGRSSN hold
- SSN -> ENAME is **non-transitive**
  - Since there is no set of attributes X where SSN -> X and X -> ENAME

RAMAIAH
Institute of Technology

Third Normal Form

A relation schema R is in **third normal form (3NF)** if it is in 2NF *and* **no non-prime attribute A in R is transitively dependent on the primary key.**

R can be decomposed into 3NF relations via the process of 3NF normalization

NOTE:

- In X -> Y and Y -> Z, with X as the primary key, we consider this a problem only if Y is not a candidate key.
- When Y is a candidate key, there is no problem with the transitive dependency .
- E.g., Consider EMP (SSN, Emp#, Salary).
  - Here, SSN -> Emp#, Emp# -> Salary and Emp# is a candidate key.

RAMAIAH
Institute of Technology

Normal Forms Defined Informally

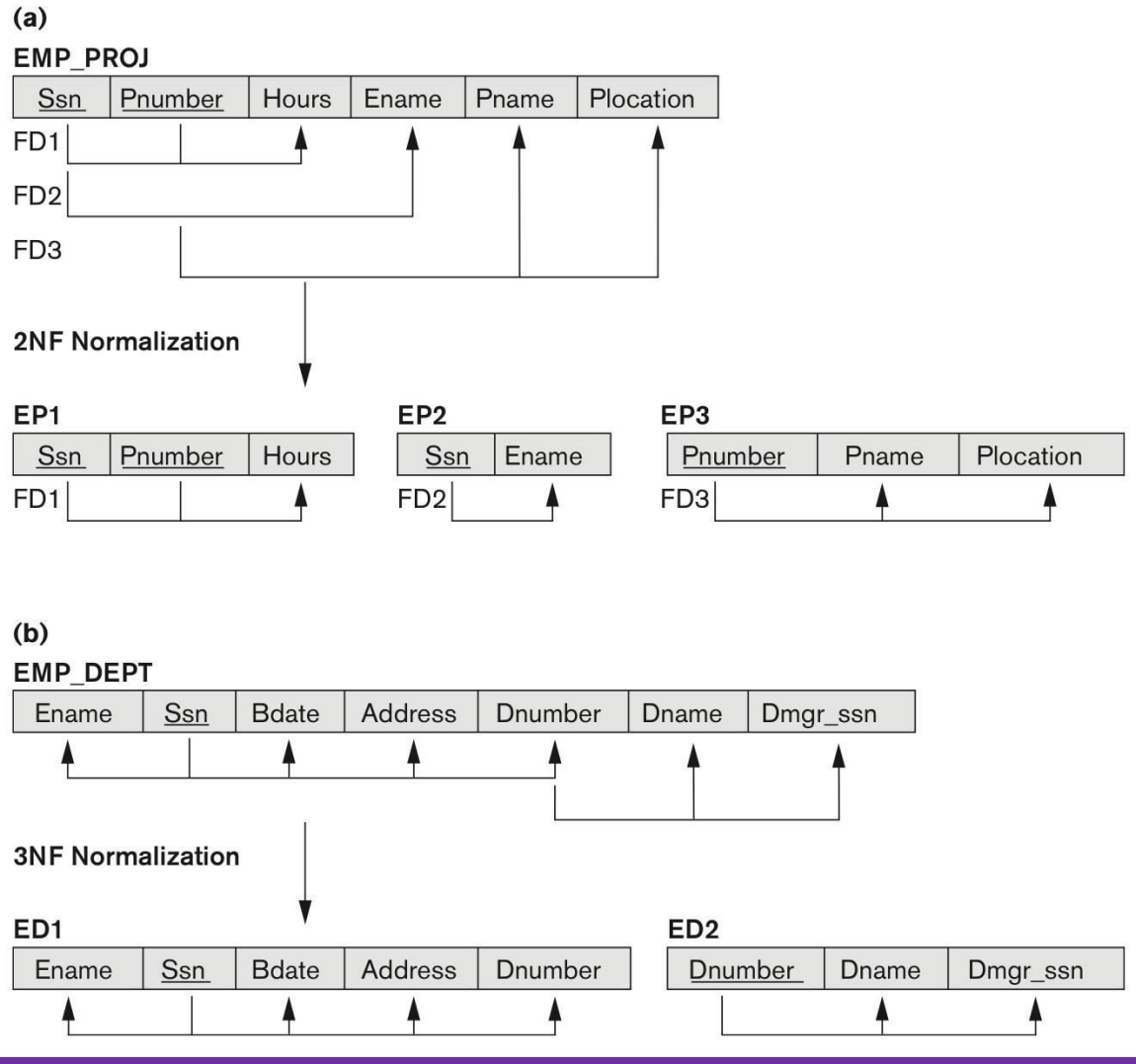
1st normal form
- All attributes depend on **the key**

2nd normal form
- All attributes depend on **the whole key**

3rd normal form
- All attributes depend on **nothing but the key**

RAMAIAH Institute of Technology

Normalizing into 2NF and 3NF



Normalizing into 2NF and 3NF.
(a) Normalizing EMP_PROJ into 2NF relations.
(b) Normalizing EMP_DEPT into 3NF relations.

Normalization into 2NF and 3NF

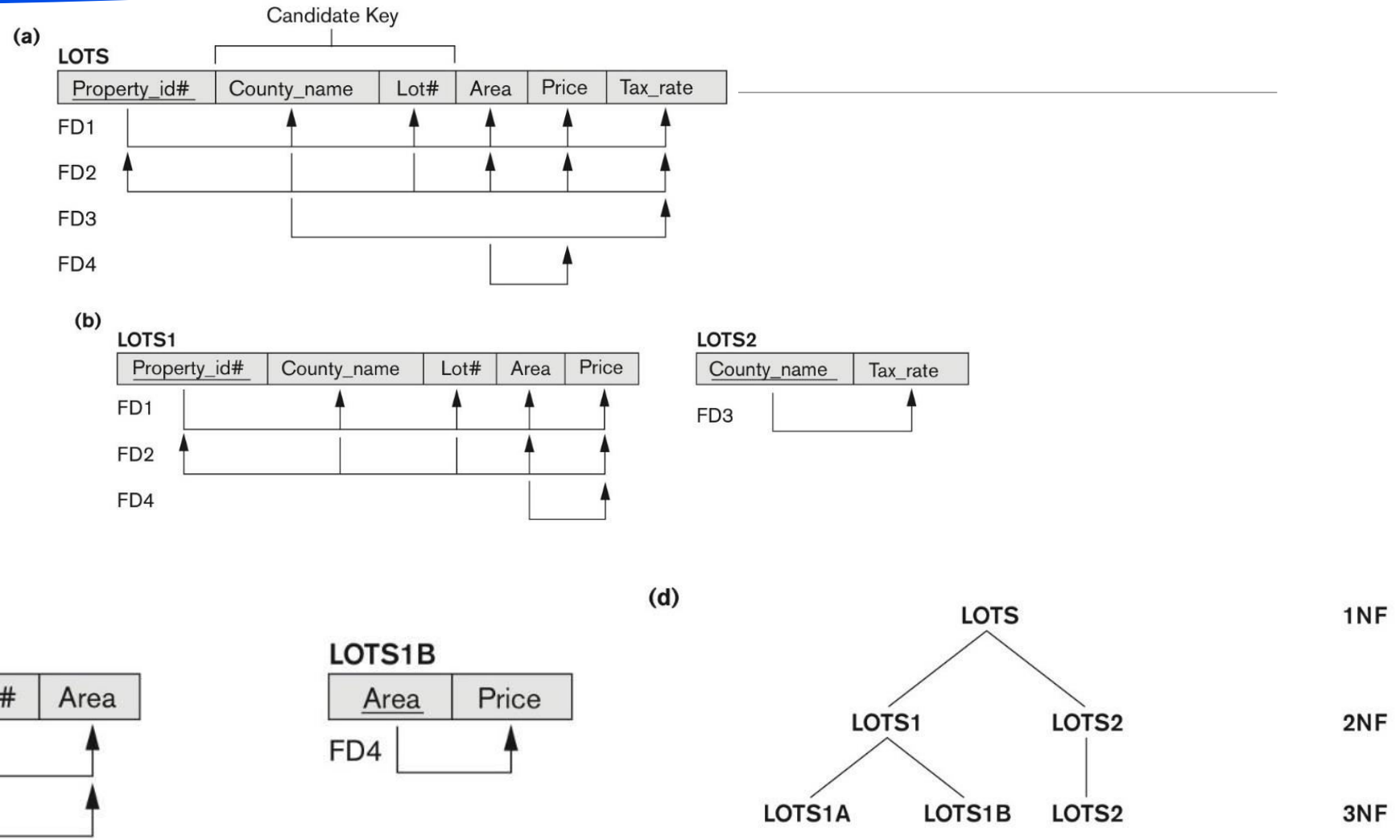
Normalization into 2NF and 3NF.

(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.



RAMAIAH
Institute of Technology

General Normal Form Definitions (For Multiple Keys)

The above definitions consider the primary key only

The following more general definitions take into account relations with multiple candidate keys

Any attribute involved in a candidate key is a *prime attribute*

All other attributes are called *non-prime attributes.*

RAMAIAH
Institute of Technology

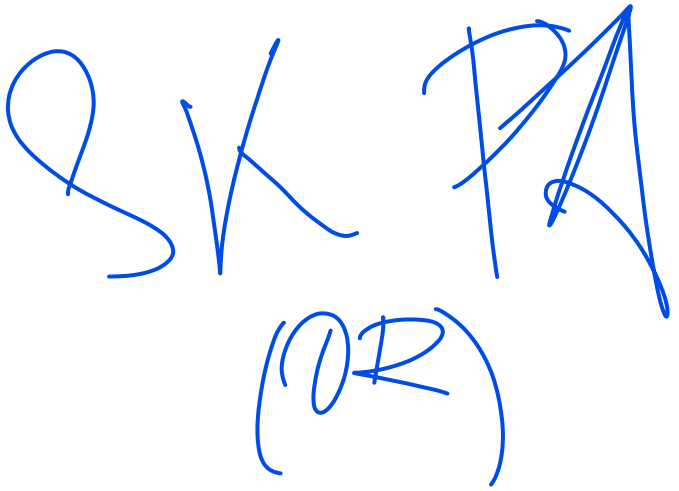
General Definition of Third Normal Form

Definition:

- **Superkey** of relation schema R - a set of attributes S of R that contains a key of R
- A relation schema R is in **third normal form (3NF)** if whenever a FD, X → A holds in R, then either:
  - (a) X is a superkey of R, or
  - (b) A is a prime attribute of R

LOTS1 relation violates 3NF because

Area → Price ; and Area is not a superkey in LOTS1.



RAMAIAH
Institute of Technology

Interpreting the General Definition of Third Normal Form

Consider the 2 conditions in the Definition of 3NF:

A relation schema R is in **third normal form (3NF)** if whenever a FD X → A holds in R, then either:

- (a) X is a superkey of R, or
- (b) A is a prime attribute of R

Condition (a) catches two types of violations :

- one where a prime attribute functionally determines a non-prime attribute. This catches 2NF violations due to non-full functional dependencies.

-second, where a non-prime attribute functionally determines a non-prime attribute. This catches 3NF violations due to a transitive dependency.

Interpreting the General Definition of Third Normal Form

ALTERNATIVE DEFINITION of 3NF: We can restate the definition as:

A relation schema R is in **third normal form (3NF)** if every non-prime attribute in R meets both of these conditions:

- It is fully functionally dependent on every key of R
- It is non-transitively dependent on every key of R

Note that stated this way, a relation in 3NF also meets the requirements for 2NF.

The condition (b) from the last slide takes care of the dependencies that "slip through" (are allowable to) 3NF but are "caught by" BCNF.

RAMAIAH
Institute of Technology

BCNF (Boyce-Codd Normal Form)

A relation schema R is in **Boyce-Codd Normal Form (BCNF)** if whenever an **FD X → A** holds in R, then **X is a superkey** of R

Each normal form is strictly stronger than the previous one

- Every 2NF relation is in 1NF
- Every 3NF relation is in 2NF
- Every BCNF relation is in 3NF

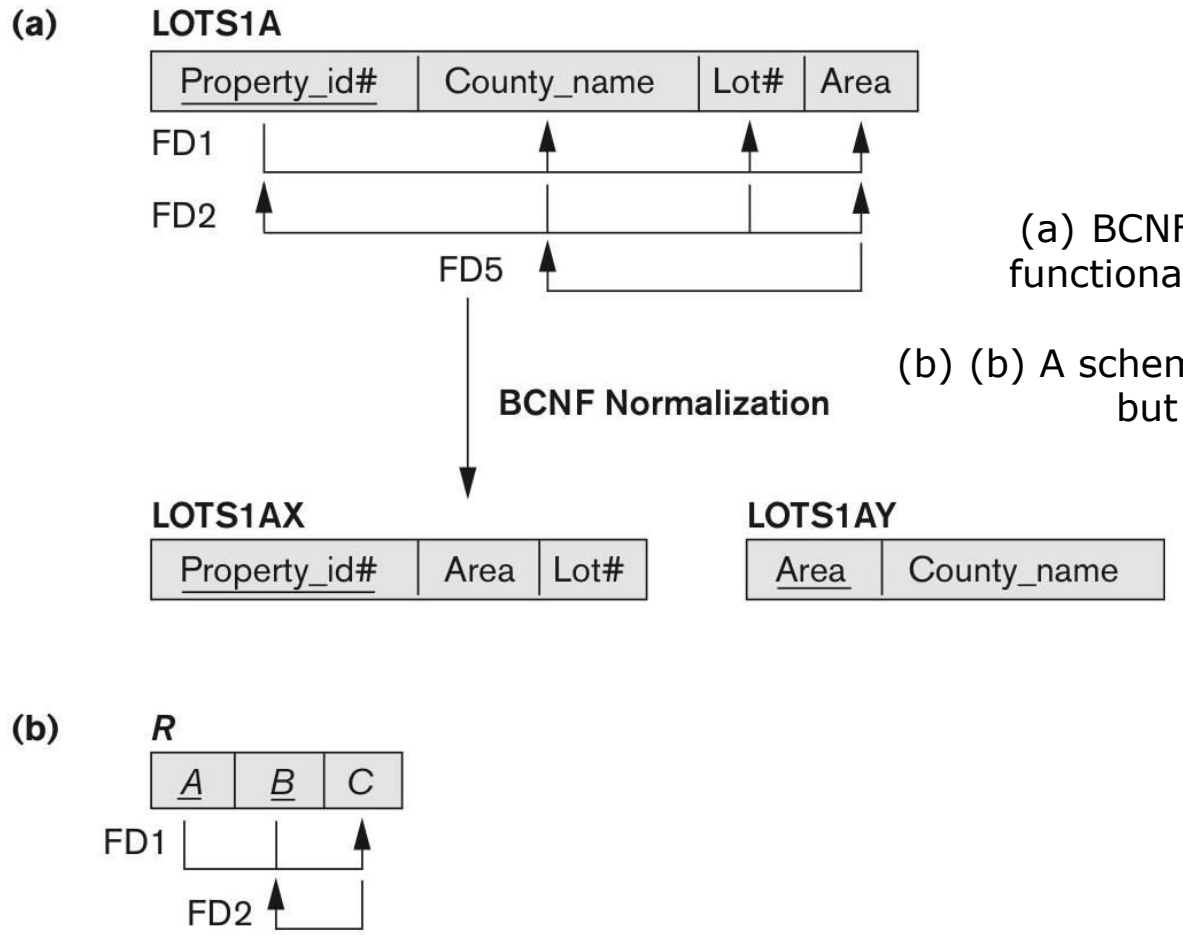
There exist relations that are in 3NF but not in BCNF

Hence BCNF is considered a stronger form of 3NF

The goal is to have each relation in BCNF (or 3NF)

RAMAIAH
Institute of Technology

Boyce-Codd normal form



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

RAMAIAH
Institute of Technology

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

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.

RAMAIAH
Institute of Technology

Achieving the BCNF by Decomposition

Two FDs exist in the relation TEACH:

- fd1: { student, course} -> instructor
- fd2: instructor -> course

{student, course} is a candidate key for this .

So this relation is in 3NF *but not in* BCNF.

A relation **NOT** in BCNF should be decomposed so as to meet this property, while possibly forgoing the preservation of all functional dependencies in the decomposed relations.

RAMAIAH
Institute of Technology

Decompose into 2NF and 3NF relations

Consider the universal relation $R = \{A, B, C, D, E, F, G, H, I, J\}$ and the set of functional dependencies $F = \{\{A, B\} \rightarrow \{C\}, \{A\} \rightarrow \{D, E\}, \{B\} \rightarrow \{F\}, \{F\} \rightarrow \{G, H\}, \{D\} \rightarrow \{I, J\}\}$. What is the key for R? Decompose R into 2NF and then 3NF relations.

RAMAIAH
Institute of Technology

Thank you

RAMAIAH
Institute of Technology

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Database Systems

Course Code: CS52

Credits: 3:1:0

UNIT 4

Term: Oct 2021– Feb 2022

Faculty:
Dr. Sini Anna Alex

RAMAIAH
Institute of Technology

Relational Database Design Algorithms and Further Dependencies

The two desirable properties of decompositions, namely,

- the **dependency preservation property** and the **lossless (or nonadditive) join property**, which are both used by the design algorithms to achieve desirable decompositions.

- Fourth normal form (4NF) and fifth normal form (5NF)

RAMAIAH
Institute of Technology

DESIGNING A SET OF RELATIONS

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

RAMAIAH
Institute of Technology

Properties of Relational Decompositions

Relation Decomposition and Insufficiency of Normal Forms:

- Universal Relation Schema:
  - A relation schema R = {A1, A2, …, An} that includes all the attributes of the database.
- Universal relation assumption:
  - Every attribute name is unique.

RAMAIAH
Institute of Technology

Properties of Relational Decompositions

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

- **Decomposition:**
  - The process of decomposing the universal relation schema R into a set of relation schemas D = {R1,R2, …, Rm} that will become the relational database schema by using the functional dependencies.
- **Attribute preservation condition:**
  - Each attribute in R will appear in at least one relation schema Ri in the decomposition so that no attributes are “lost”.

RAMAIAH
Institute of Technology

Properties of Relational Decompositions

Another goal of decomposition is to have each individual relation Ri in the decomposition D be in BCNF or 3NF.

Additional properties of decomposition are needed to prevent from generating spurious tuples.

RAMAIAH
Institute of Technology

Properties of Relational Decompositions

Dependency Preservation Property of a Decomposition:

- Definition: Given a set of dependencies F on R, the **projection** of F on R_i, denoted by $p_{Ri}(F)$ 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

Dependency Preservation Property of a Decomposition (cont.):

- Dependency Preservation Property:
  - A decomposition D = {R1, R2, ..., Rm} of R is **dependency-preserving** with respect to F if the union of the projections of F on each Ri in D is equivalent to F; that is
    $((\pi_{R1}(F)) \cup \ldots \cup (\pi_{Rm}(F)))^+ = F^+$

Claim 1:

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

RAMAIAH
Institute of Technology

Properties of Relational Decompositions

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

- Definition: Lossless join property: a decomposition D = {R1, R2, ..., Rm} 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:

$$* (\pi_{R1}(r), ..., \pi_{Rm}(r)) = r$$

- 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**"

RAMAIAH Institute of Technology

Properties of Relational Decompositions

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 = {R1, R2, ..., Rm} of R, and a set F of functional dependencies.

1. Create an initial matrix S with one row i for each relation Ri in D, and one column j for each attribute Aj in R.

2. Set S(i,j):=bij for all matrix entries. (* each bij is a distinct symbol associated with indices (i,j) *).

3. For each row i representing relation schema Ri

{for each column j representing attribute Aj

{if (relation Ri includes attribute Aj) then set S(i,j):= aj;};};

- (* each aj is a distinct symbol associated with index (j) *)

- CONTINUED on NEXT SLIDE

RAMAIAH Institute of Technology

Properties of Relational Decompositions

4. Repeat the following loop until a complete loop execution results in no changes to S

{for each functional dependency X →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.

RAMAIAH Institute of Technology

Properties of Relational Decompositions

Lossless (nonadditive) join test for *n*-ary decompositions.
(a) Case 1: Decomposition of EMP_PROJ into EMP_PROJ1 and EMP_LOCS fails test.
(b) A decomposition of EMP_PROJ that has the lossless join property.

(a) R={SSN, ENAME, PNUMBER, PNAME, PLOCATION, HOURS} $D=\{R_1, R_2\}$
R_1=EMP_LOCS={ENAME, PLOCATION}
R_2=EMP_PROJ1={SSN, PNUMBER, HOURS, PNAME, PLOCATION}

F={SSN→ENAME;PNUMBER→{PNAME, PLOCATION} ;{SSN,PNUMBER}→HOURS}

	SSN	ENAME	PNUMBER	PNAME	PLOCATION	HOURS
R_1	b_{11}	a_2	b_{13}	b_{14}	a_5	b_{16}
R_2	a_1	b_{22}	a_3	a_4	a_5	a_6

(no changes to matrix after applying functional dependencies)

(b)

EMP		PROJECT			WORKS_ON		
SSN	ENAME	PNUMBER	PNAME	PLOCATION	SSN	PNUMBER	HOURS

RAMAIAH Institute of Technology

Properties of Relational Decompositions

Lossless (nonadditive) join test for n-ary decompositions. (c) Case 2: Decomposition of EMP_PROJ into EMP, PROJECT, and WORKS_ON satisfies test.

(c)

R={SSN, ENAME, PNUMBER, PNAME, PLOCATION, HOURS}

D={R_1, R_2, R_3}

R_1=EMP={SSN, ENAME}

R_2=PROJ={PNUMBER, PNAME, PLOCATION}

R_3=WORKS_ON={SSN, PNUMBER, HOURS}

F={SSN→{ENAME;PNUMBER→{PNAME, PLOCATION} ;{SSN,PNUMBER}→HOURS}

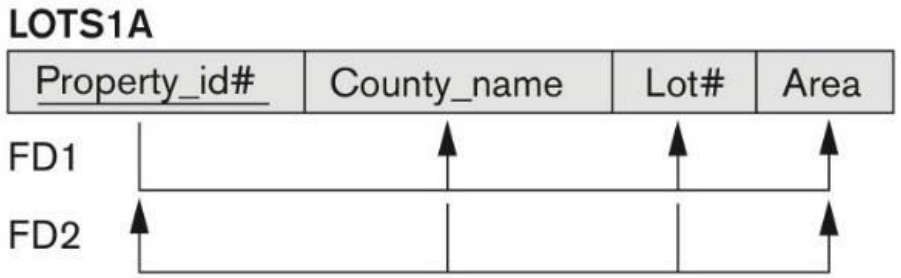
	SSN	ENAME	PNUMBER	PNAME	PLOCATION	HOURS
R_1	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}
R_2	b_{21}	b_{22}	a_3	a_4	a_5	b_{26}
R_3	a_1	b_{32}	a_3	b_{34}	b_{35}	a_6

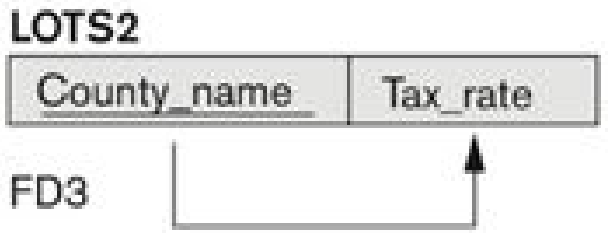
(original matrix S at start of algorithm)

	SSN	ENAME	PNUMBER	PNAME	PLOCATION	HOURS
R_1	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}
R_2	b_{21}	b_{22}	a_3	a_4	a_5	b_{26}
R_3	a_1	~~b_{32}~~ a_2	a_3	~~b_{34}~~ a_4	~~b_{35}~~ a_5	a_6

(matrix S after applying the first two functional dependencies - last row is all "a" symbols, so we stop)

(c)





D1(P,L,C,A)

D2(A,P)

D3(C,T)

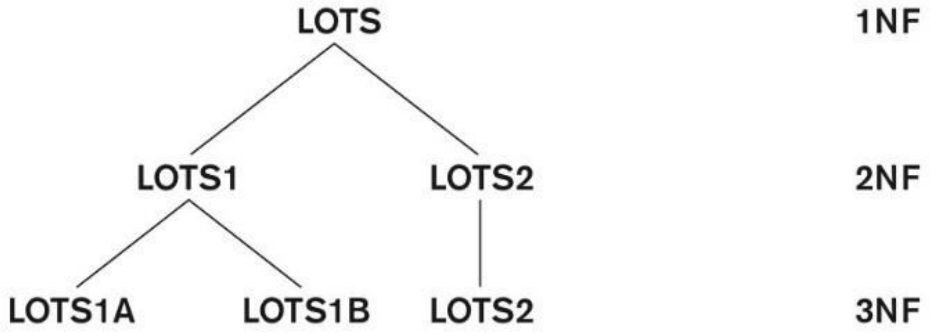
P→LCA

A→P

C→T

CL→AP

(d)



RAMAIAH
Institute of Technology

Properties of Relational Decompositions

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 = {R1, R2} 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. ((R1 ∩ R2) → (R1- R2)) is in F^+, or
  - The f.d. ((R1 ∩ R2) → (R2 - R1)) is in F^+.

RAMAIAH
Institute of Technology

Properties of Relational Decompositions

Consider a schema R (A, B, C, D) and functional dependencies A→B and C→D. Then the decomposition of R into R1(AB) and R2(CD) is _____

A. Dependency preserving and lossless join
B. Lossless join but not dependency preserving
C. Dependency preserving but not lossless join
D. Not dependency preserving and not lossless join

RAMAIAH
Institute of Technology

Thank you

Algorithms for Relational Database Schema Design

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;

2. For each left-hand-side X of a functional dependency that appears in G,

create a relation schema in D with attributes {X ∪ {A1} ∪ {A2} ... ∪ {Ak}},

where X → A1, X → A2, ..., X → Ak 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: Every relation schema created by Algorithm is in 3NF.

RAMAIAH Institute of Technology

U(Essn, Pno, Esal, Ephone, Dno, Pname, Plocation)

FD:

Essn→ Esal, Ephone, Dno

Pno→Pname, Plocation

Essn, Pno → Esal, Ephone, Dno, Pname, Plocation

Minimal Cover:

Essn→ Esal, Ephone, Dno

Pno→Pname, Plocation

Decomposed relations:

R1(<u>Essn</u>, Esal, Ephone, Dno)

R2(<u>Pno</u>,Pname, Plocation)

RAMAIAH Institute of Technology

Algorithms for Relational Database Schema Design

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 → Y in Q that violates BCNF;

replace Q in D by two relation schemas (Q - Y) and (X ∪ Y);

};

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

DB(Patno,PatName,appNo,time,doctor)
Patno -> PatName
Patno -> Time,doctor
Time -> appNo
appNo-> Patno

2NF AND 3NF

DB(Patno,appNo,Time,doctor, PatName)

BCNF

Q=DB(Patno,appNo,Time,doctor, PatName)
Time -> appNo (X→Y, violates BCNF)
Replace Q into: Q-Y AND XUY
D1(Patno,time,doctor, PatName)
D2(Time, appNo)

RAMAIAH Institute of Technology

General Procedure for achieving BCNF when a relation fails BCNF

Let R be the relation not in BCNF, let X be a subset-of R, and let $X \rightarrow A$ be the FD that causes a violation of BCNF. Then R may be decomposed into two relations:

(i) $R - A$ and (ii) $X \cup A$.

If either $R - A$ or $X \cup A$. is not in BCNF, repeat the process.

Note that the f.d. that violated BCNF in TEACH(Instructor, Student, Course)

was Instructor $\rightarrow$Course. Hence its BCNF decomposition would be (TEACH – COURSE) and (Instructor ∪ Course), which gives

the relations: (Instructor, Student) and (Instructor, Course) that we obtained before in decomposition D3.

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 attributes of R.

Method:

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}
   };

Let R = (A, B, C, D, E) be a relation scheme with the following dependencies-

$AB \rightarrow C$
$C \rightarrow D$
$B \rightarrow E$

RAMAIAH
Institute of Technology

Algorithms for Relational Database Schema Design

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.

2. For each left-hand-side X of a functional dependency that appears in G,

create a relation schema in D with attributes {X ∪ {A1} ∪ {A2} ... ∪ {Ak}},

where X → A1, X → A2, ..., X –>Ak 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 to find the key of R)*

4. Eliminate redundant relations from the resulting set of relations in the relational database schema.

RAMAIAH Institute of Technology

Example

FD:

{P→LCA, LC→AP, A→C}

Minimal Cover: P→LC,LC→AP, A→C

Design X: R1(P,L,C), R2(L,C, A, P) and R3(A,C)

R3 is subsumed by R2, R1 by R2 //eliminating redundant relations

Final Design: R2(L,C,A,P)

F: {P→LCA, LC→AP, A→C}

LC→P , P→A => LC→A
P→A , A→C => P→C

MC:{P→LA, LC→P, A->C}

Alternate Design: S1(P,A,L),S2(L,C,P) AND S3(A,C)

Problems with NULL values and Dangling Tuples

Problems with NULL values

when some tuples have NULL values for attributes that will be used to join individual relations in the decomposition that may lead to incomplete results.

E.g., see Figure 15.2(a), where two relations EMPLOYEE and DEPARTMENT are shown. The last two employee tuples—'Berger' and 'Benitez'—represent newly hired employees who have not yet been assigned to a department (assume that this does not violate any integrity constraints).

If we want to retrieve a list of (Ename, Dname) values for all the employees. If we apply the NATURAL JOIN operation on EMPLOYEE and DEPARTMENT (Figure 15.2(b)), the two aforementioned tuples will *not* appear in the result.

In such cases, LEFT OUTER JOIN may be used. The result is shown in Figure 15.2 (c).

(a)

EMPLOYEE

Ename	Ssn	Bdate	Address	Dnum
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	291 Berry, 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
Berger, Anders C.	999775555	1965-04-26	6530 Braes, Bellaire, TX	NULL
Benitez, Carlos M.	888664444	1963-01-09	7654 Beech, Houston, TX	NULL

DEPARTMENT

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

Figure 15.2
Issues with NULL-value joins. (a) Some EMPLOYEE tuples have NULL for the join attribute Dnum.

RAMAIAH
Institute of Technology

Problems with Null Values and Dangling Tuples

(b)

Ename	Ssn	Bdate	Address	Dnum	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 Fire Oak, 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

(c)

Ename	Ssn	Bdate	Address	Dnum	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 Fire Oak, 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
Berger, Anders C.	999775555	1965-04-26	6530 Braes, Bellaire, TX	NULL	NULL	NULL
Benitez, Carlos M.	888665555	1963-01-09	7654 Beech, Houston, TX	NULL	NULL	NULL

Figure 15.2
Issues with NULL-value joins.
(b) Result of applying NATURAL JOIN to the EMPLOYEE and DEPARTMENT relations.
(c) Result of applying LEFT OUTER JOIN to EMPLOYEE and DEPARTMENT

RAMAIAH
Institute of Technology

About Normalization Algorithms

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.

RAMAIAH
Institute of Technology

Algorithms for Relational Database Schema Design

Table 11.1
Summary of the Algorithms Discussed in Sections 11.1 and 11.2

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 nonadditive join property	Testing for nonadditive 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	Nonadditive join decomposition	No guarantee of dependency preservation
11.4	Set of functional dependencies F	A set of relations in 3NF	Nonadditive join *and* dependency-preserving decomposition	May not achieve BCNF, but achieves *all* desirable properties and 3NF
11.4a	Relation schema R with a set of functional dependencies F	Key K of R	To find a key K (that is a subset of R)	The entire relation R is always a default superkey

RAMAIAH
Institute of Technology

Thank you

RAMAIAH Institute of Technology

Algorithms for Relational Database Schema Design

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 → Y in Q that violates BCNF;

replace Q in D by two relation schemas (Q - Y) and (X ∪ Y);

};

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

DB(Patno,PatName,appNo,time,doctor)
Patno -> PatName
Patno -> Time,doctor
Time -> appNo
appNo-> Patno

2NF AND 3NF

DB(Patno,appNo,Time,doctor, PatName)

BCNF

Q=DB(Patno,appNo,Time,doctor, PatName)
Time -> appNo (X→Y, violates BCNF)
Replace Q into: Q-Y AND XUY
D1(Patno,time,doctor, PatName)
D2(Time, appNo)

RAMAIAH Institute of Technology

General Procedure for achieving BCNF when a relation fails BCNF

Let R be the relation not in BCNF, let X be a subset-of R, and let $X \rightarrow A$ be the FD that causes a violation of BCNF. Then R may be decomposed into two relations:

(i) $R - A$ and (ii) $X \cup A$.

If either $R - A$ or $X \cup A$. is not in BCNF, repeat the process.

Note that the f.d. that violated BCNF in TEACH(Instructor, Student, Course)

was Instructor $\rightarrow$Course. Hence its BCNF decomposition would be (TEACH – COURSE) and (Instructor ∪ Course), which gives

the relations: (Instructor, Student) and (Instructor, Course) that we obtained before in decomposition D3.

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 attributes of R.

Method:

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}
   };

Let R = (A, B, C, D, E) be a relation scheme with the following dependencies-

$AB \rightarrow C$
$C \rightarrow D$
$B \rightarrow E$

RAMAIAH
Institute of Technology

Algorithms for Relational Database Schema Design

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.

2. For each left-hand-side X of a functional dependency that appears in G,

create a relation schema in D with attributes {X ∪ {A1} ∪ {A2} ... ∪ {Ak}},

where X → A1, X → A2, ..., X –>Ak 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 to find the key of R)*

4. Eliminate redundant relations from the resulting set of relations in the relational database schema.

RAMAIAH Institute of Technology

Example

FD:

{P→LCA, LC→AP, A→C}

Minimal Cover: P→LC,LC→AP, A→C

Design X: R1(P,L,C), R2(L,C, A, P) and R3(A,C)

R3 is subsumed by R2, R1 by R2 //eliminating redundant relations

Final Design: R2(L,C,A,P)

F: {P→LCA, LC→AP, A→C}

LC→P , P→A => LC→A
P→A , A→C => P→C

MC:{P→LA, LC→P, A->C}

Alternate Design: S1(P,A,L),S2(L,C,P) AND S3(A,C)

Problems with NULL values and Dangling Tuples

Problems with NULL values

when some tuples have NULL values for attributes that will be used to join individual relations in the decomposition that may lead to incomplete results.

E.g., see Figure 15.2(a), where two relations EMPLOYEE and DEPARTMENT are shown. The last two employee tuples—'Berger' and 'Benitez'—represent newly hired employees who have not yet been assigned to a department (assume that this does not violate any integrity constraints).

If we want to retrieve a list of (Ename, Dname) values for all the employees. If we apply the NATURAL JOIN operation on EMPLOYEE and DEPARTMENT (Figure 15.2(b)), the two aforementioned tuples will *not* appear in the result.

In such cases, LEFT OUTER JOIN may be used. The result is shown in Figure 15.2 (c).

(a)

EMPLOYEE

Ename	Ssn	Bdate	Address	Dnum
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	291 Berry, 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
Berger, Anders C.	999775555	1965-04-26	6530 Braes, Bellaire, TX	NULL
Benitez, Carlos M.	888664444	1963-01-09	7654 Beech, Houston, TX	NULL

DEPARTMENT

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

Figure 15.2
Issues with NULL-value joins. (a) Some EMPLOYEE tuples have NULL for the join attribute Dnum.

RAMAIAH
Institute of Technology

Problems with Null Values and Dangling Tuples

(b)

Ename	Ssn	Bdate	Address	Dnum	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 Fire Oak, 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

(c)

Ename	Ssn	Bdate	Address	Dnum	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 Fire Oak, 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
Berger, Anders C.	999775555	1965-04-26	6530 Braes, Bellaire, TX	NULL	NULL	NULL
Benitez, Carlos M.	888665555	1963-01-09	7654 Beech, Houston, TX	NULL	NULL	NULL

Figure 15.2
Issues with NULL-value joins.
(b) Result of applying NATURAL JOIN to the EMPLOYEE and DEPARTMENT relations.
(c) Result of applying LEFT OUTER JOIN to EMPLOYEE and DEPARTMENT

RAMAIAH
Institute of Technology

About Normalization Algorithms

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.

RAMAIAH
Institute of Technology

Algorithms for Relational Database Schema Design

Table 11.1
Summary of the Algorithms Discussed in Sections 11.1 and 11.2

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 nonadditive join property	Testing for nonadditive 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	Nonadditive join decomposition	No guarantee of dependency preservation
11.4	Set of functional dependencies F	A set of relations in 3NF	Nonadditive join *and* dependency-preserving decomposition	May not achieve BCNF, but achieves *all* desirable properties and 3NF
11.4a	Relation schema R with a set of functional dependencies F	Key K of R	To find a key K (that is a subset of R)	The entire relation R is always a default superkey

RAMAIAH
Institute of Technology

Thank you