LECTURE IS COMING

- We start @ 13:15
- Remember to join MS Teams – code: 6esizxc
- You can download the lecture from MS Teams for your convenience

PRINCIPLES OF DATABASE DESIGN

Instructor: **Krystian Wojtkiewicz**

School of Computer Science and Engineering

International University, VNU-HCMC

Lecture 8: Normalization

ACKNOWLEDGEMENT

The following slides have been created based on Database system concepts book, 7th Edition.

The following slides are referenced from Northeastern University.

KEYS AND FDS: REVIEW

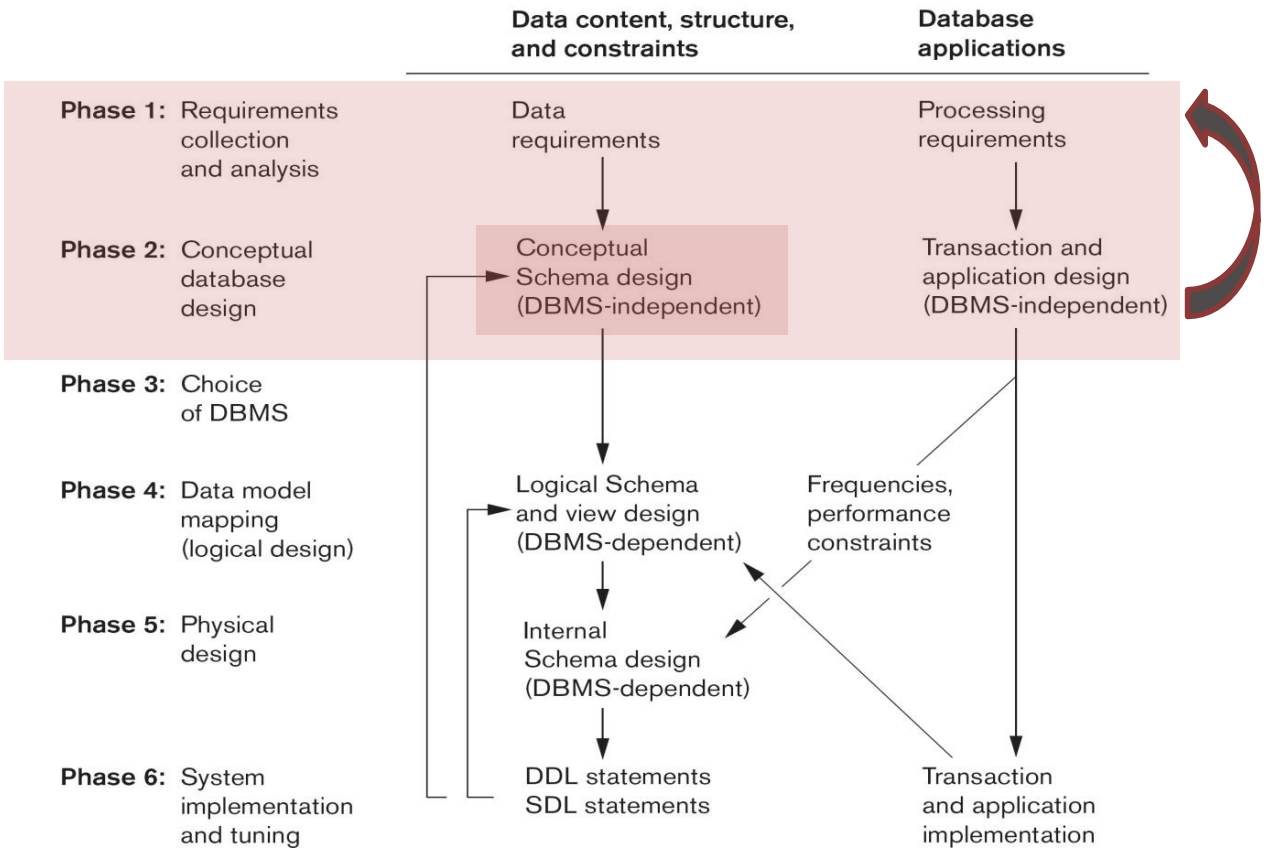
- Functional Dependencies
- Keys/Super keys
- Attribute closure
- Minimal cover

TODAY'S TOPICS

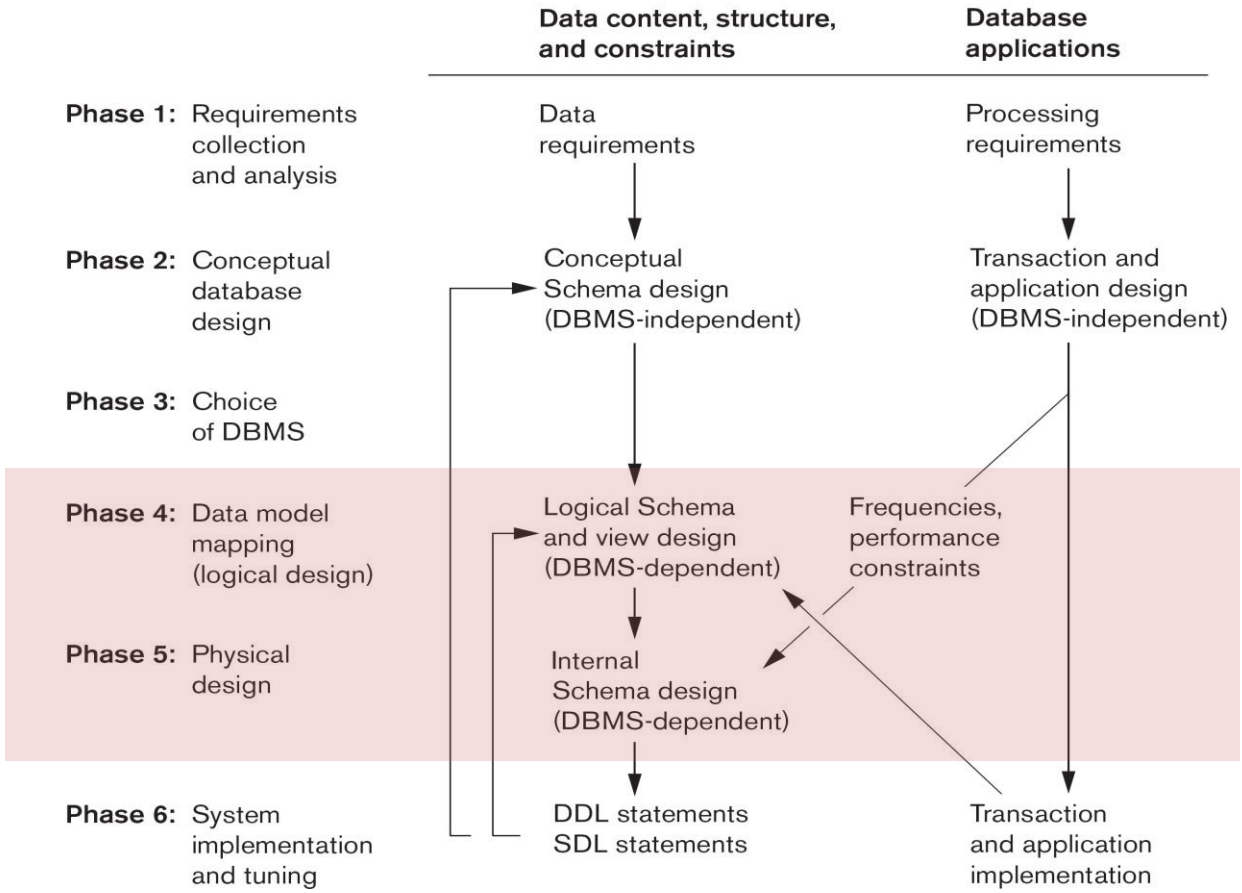
Normalization Objective

Normal forms

DATABASE DESIGN AND IMPLEMENTATION PROCESS



DATABASE DESIGN AND IMPLEMENTATION PROCESS



THE DATABASE INITIAL STUDY

- Overall purpose:
  - Analyze company situation
  - Define problems and constraints
  - Define objectives
  - Define scope and boundaries
- Interactive and iterative processes required to complete first phase of DBLC (Database Life Cycle) successfully.

THE DATABASE INITIAL STUDY (CONT'D)

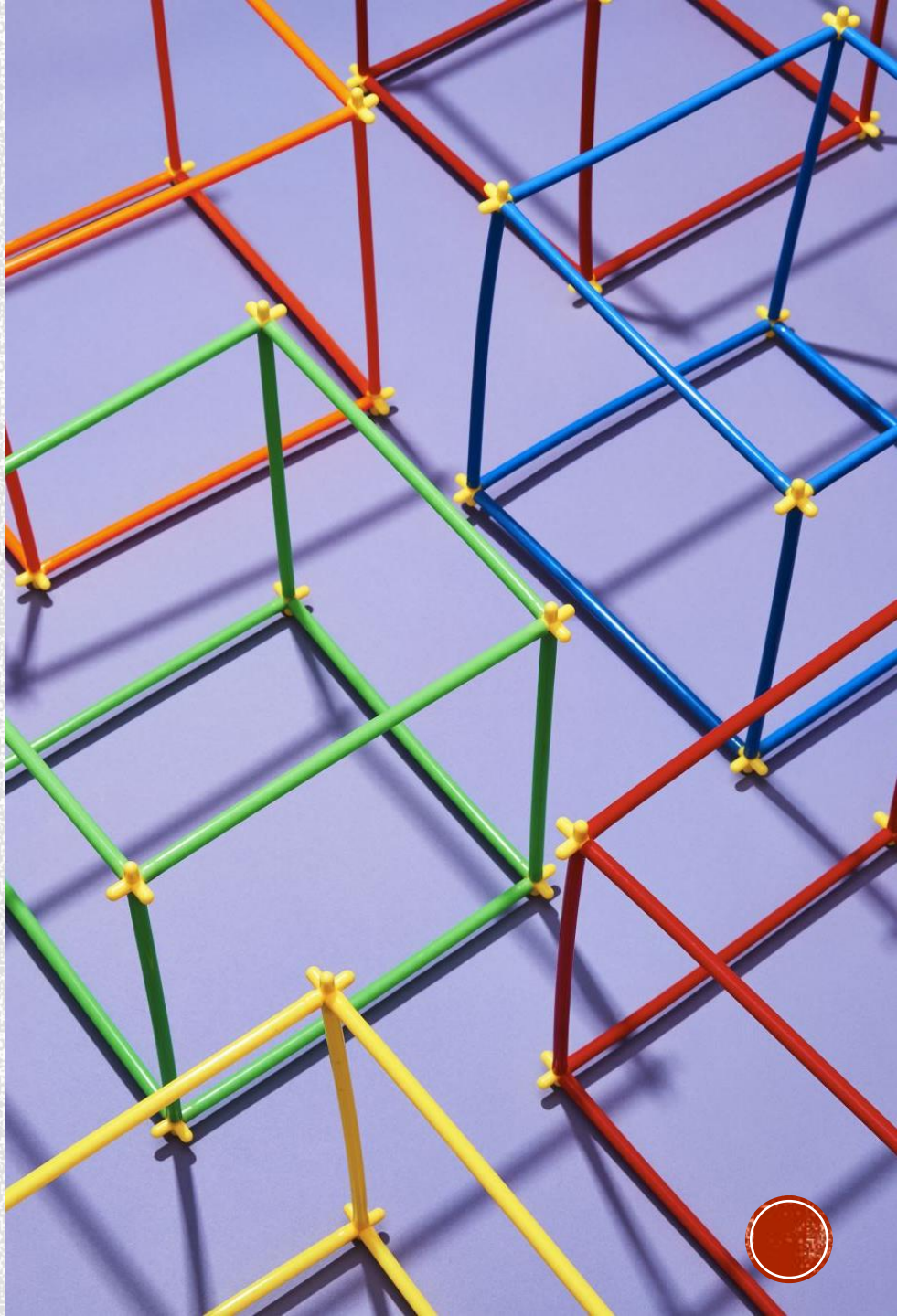
- Analyze the company situation
  - General conditions in which company operates, its organizational structure, and its mission.
  - Discover what company's operational components are, how they function, and how they interact.

DATA ANALYSIS AND REQUIREMENTS

- Discover data element characteristics
  - Obtains characteristics from different sources
- Requires thorough understanding of the company's data types and their extent and uses.
- Take into account business rules
  - Derived from description of operations

DATABASE DESIGN

- Necessary to concentrate on data characteristics required to build database model.
- Two views of data within system:
  - Business view
    - Data as information source
  - Designer's view
    - Data structure, access, and activities required to transform data into information

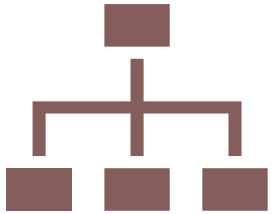


DBMS SOFTWARE SELECTION

- Critical to information system's smooth operation
- Common factors affecting purchasing decisions:
  - Cost
  - DBMS features and tools
  - Underlying model
  - Portability
  - DBMS hardware requirements

MAP THE CONCEPTUAL MODEL TO THE LOGICAL MODEL

Map the conceptual model to the chosen database constructs



Five mapping steps involved:

Strong entities

Supertype/subtype relationships

Weak entities

Binary relationships

Higher degree relationships

DETAILED SYSTEMS DESIGN

Designer completes design of system's processes

Includes all necessary technical specifications

Steps laid out for conversion from old to new system

Training principles and methodologies are also planned

- – Submitted for management approval

IMPLEMENTATION AND LOADING

- Implement all design specifications from the previous phase:
  - Install the DBMS
    - Virtualization: creates logical representations of computing resources independent of physical resources
  - Create the Database
  - Load or Convert the Data

NORMALIZATION

Theory and process by which to evaluate and improve relational database design

Typically divide larger tables into smaller, less redundant tables

Spans both logical and physical database design

OBJECTIVES OF NORMALIZATION

1. Make the schema informative
2. Minimize information duplication
3. Avoid modification anomalies
4. Disallow spurious tuples

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

STRAW MAN SCHEMA

EXAMPLE 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

MAKE THE SCHEMA INFORMATIVE

Design a relational schema so that it is easy to explain its meaning.

Do **not** combine attributes from multiple entity types and relationship types into a single relation; semantic ambiguities will result and the relation cannot be easily explained.

Normalized tables, and the relationship between one normalized table and another, mirror real- world concepts and their interrelationships.

EXAMPLE SCHEMA

What is this table about?

- Employees? Departments?

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

MINIMIZE INFORMATION DUPLICATION

- Avoid data redundancies

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

- Avoid excessive use of NULLs (e.g. fat tables)
  - Wastes space
  - Can make information querying/understanding complicated and error-prone

AVOID MODIFICATION ANOMALIES

An undesired side-effect resulting from an attempt to modify a table (that has not been sufficiently normalized)

Types of modifications:

- Insertion
- Update
- Deletion

INSERTION ANOMALY

Difficult or impossible to insert a new row

- Add a new employee
  - Unknown manager
  - Typo in department/manager info
- Add a new department
  - Requires at least one employee

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

UPDATE ANOMALY

Updates may result in logical inconsistencies

- Change the department name/manager

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

DELETION ANOMALY

Deletion of data representing certain facts necessitates deletion of data representing completely different facts

- Delete James E. Borg

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

DISALLOW SPURIOUS TUPLES

Avoid relational design that matches attributes across relations that are not (foreign key, primary key) combinations because joining on such attributes may produce invalid tuples

BAD DECOMPOSITION

CAR

ID	Make	Color
1	Toyota	Blue
2	Audi	Blue
3	Toyota	Red

CAR1

ID	Color
1	Blue
2	Blue
3	Red

CAR2

Make	Color
Toyota	Blue
Audi	Blue
Toyota	Red

Association between Color and Make is lost.

BAD DECOMPOSITION

ID	Make	Color
1	Toyota	Blue
1	Audi	Blue
2	Toyota	Blue
2	Audi	Blue
3	Toyota	Red

CAR1

CAR2

ID	Color
1	Blue
2	Blue
3	Red

Make	Color
Toyota	Blue
Audi	Blue
Toyota	Red

Join returns more rows than the original relation

ADDITIVE DECOMPOSITION

CAR

ID	Make	Color
1	Toyota	Blue
2	Audi	Blue
3	Toyota	Red

JOIN

ID	Make	Color
1	Toyota	Blue
1	**Audi**	**Blue**
2	**Toyota**	**Blue**
2	Audi	Blue
3	Toyota	Red

LOSSLESS JOIN DECOMPOSITION

Decompose relation R into relations S and T

Attrs(R) = attrs (S) ∪ attrs(T)

S = $\pi_{attrs(S)}(R)$

T = $\pi_{attrs(T)}(R)$

The decomposition is a lossless join decomposition if, given known constraints such as FD's, we can guarantee that R = S ⋈ T

NORMALIZATION PROCESS

- Submit a relational schema to a set of tests (related to FDs) to certify whether it satisfies a **normal form**
- If it does not pass, decompose into smaller relations that satisfy the normal form
  - Must be non-additive (i.e. no spurious tuples!)
- The normal form of a relation refers to the highest normal form that it meets
- The normal form of a database refers to the lowest normal form that any relation meets
  - Practically, a database is normalized if all relations ≥ 3NF

- The domain of an attribute must include only atomic values, and the value of any attribute in a tuple must be a single value from the domain of that attribute
- No relations within relations or relations as attribute values within tuples
- Considered part of the formal definition of a relation in the basic (flat) relational model
  - In other words, an *implicit* constraint

A relation is in first normal form if every attribute in every row can contain only one single (atomic) value.

1NF – FIRST NORMAL FORM

Examples: 1NF?

Student(FirstName, LastName, Knowledge)

FirstName	LastName	Knowledge
Thomas	Mueller	Java, C++, PHP
Ursula	Meier	PHP, Java
Igor	Mueller	C++, Java

The attribute Knowledge can contain multiple values and therefore the relation is not in the first normal form.

But the attributes FirstName and LastName are atomic attributes that can contain only one value.

EXAMPLES: 1NF VIOLATION

FirstName	LastName	Knowledge
Thomas	Mueller	Java, C++, PHP
Ursula	Meier	PHP, Java
Igor	Mueller	C++, Java

FirstName	LastName	Knowledge
Thomas	Mueller	Java
Thomas	Mueller	C++
Thomas	Mueller	PHP
Ursula	Meier	PHP
Ursula	Meier	Java
Igor	Mueller	C++
Igor	Mueller	Java

Full names	Physical address	Movies rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean; Clash of the Titans	Ms.
Robert Phil	3rd street 34	Forgetting Sarah Marshal; Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

EXAMPLES: 1NF?

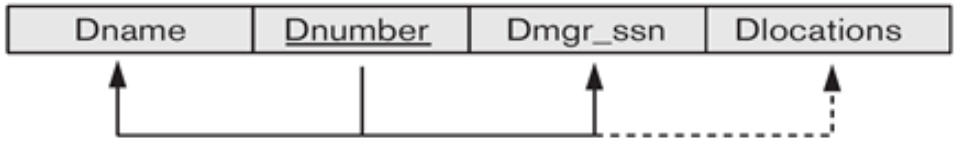
Assume a video library maintains a database of movies rented out. Without any normalization, all information is stored in one table as shown above.

EXAMPLES: 1NF

Full names	Physical address	Movies rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr

EXAMPLES 1NF?

DEPARTMENT



(b)

DEPARTMENT

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

1NF VIOLATION

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}

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

Trivial FD	$X \rightarrow Y, Y \subseteq X$
Non-prime attribute	An attribute that does not occur in any key (opposite: Prime)
Full FD	$X \rightarrow Y, \forall A \in X((X - \{A\}) \not\rightarrow Y)$
Transitive FD	$X \rightarrow Y$ and $Y \rightarrow Z \therefore X \rightarrow Z$

IMPORTANT FD DEFINITIONS

2NF – SECOND NORMAL FORM

1NF **AND** every non-prime attribute is fully FD on the primary key.

- – Must test all FDs whose LHS is part of the PK

To fix, decompose into relations in which non-prime attributes are associated only with the part of the primary key on which they are fully functionally dependent.

A relation is in second normal form if it is in 1NF and every non key attribute is fully functionally dependent on the primary key.

EXAMPLE 2NF?

StudentID	Course	StudentAddress
1	COMP570	555 Huntington
1	COMP285	555 Huntington
2	COMP570	610 Huntington
3	COMP355	Louis Prang
3	COMP553	Louis Prang

$\{StudentID, Course\} \rightarrow \{StudentAddress\}$

$\{StudentID\} \rightarrow \{StudentAddress\}$

StudentID	StudentAddress
1	555 Huntington
2	610 Huntington
3	Louis Prang

StudentID	Course
1	COMP570
1	COMP285
2	COMP570
3	COMP355
3	COMP553

EXAMPLES 2NF?

Students(IDSt, StudentName, IDProf, ProfessorName, Grade)
F={IDProf → ProfessorName; IDSt → StudentName; IDSt, IDProf →Grade}
The attributes IDSt and IDProf are the identification keys.

Students

IDSt	StudentName	IDProf	ProfessorName	Grade
1	Mueller	3	Schmid	5
2	Meier	2	Borner	4
3	Tobler	1	Bernasconi	3

EXAMPLES 2NF?

All attributes a single valued (1NF).

Students

IDSt	StudentName
1	Mueller
2	Meier
3	Tobler

Professors

IDProf	ProfessorName
1	Bernasconi
2	Borner
3	Schmid

Grade

IDSt	IDProf	Grade
1	3	5
2	2	4
3	1	6

EXAMPLES 2NF?

Suppose a school wants to store the data of teachers and the subjects they teach. They create a table that looks like this: Since a teacher can teach more than one subjects, the table can have multiple rows for a same teacher.

Teacher

Teacher_id	Subject	Teacher_age
111	Maths	38
111	Physics	38
222	Biology	38
333	Physics	40
333	Chemistry	40

EXAMPLES 2NF?

Teacher(Teacher_id, Subject, Teacher_age)

F={Teacher_id, Subject → Teacher_age; Teacher_id → Teacher_age}

Only key is: {Teacher_id, Subject}

Teacher

Teacher_id	Subject	Teacher_age
111	Maths	38
111	Physics	38
222	Biology	38
333	Physics	40
333	Chemistry	40

EXAMPLES 2NF?

To make the table complies with 2NF we can break it in two tables like this.

Teacher

Teacher_id	Teacher_age
111	38
222	38
333	40

Teacher_Subject

Teacher_id	Subject
111	Maths
111	Physics
222	Biology
333	Physics
333	Chemistry

2NF CAN SUFFER UPDATE ANOMALIES

Year	Winner	Nationality
1994	Miguel Indurain	Spain
1995	Miguel Indurain	Spain
1996	Bjarne Riis	Denmark
1997	Jan Ullrich	Germany

Relation is in 2NF?

- – Trivially true (why?)

List all non-trivial FDs for this relation state

- {*Year*} → {*Winner, Nationality*}
- {Winner} →{Nationality}

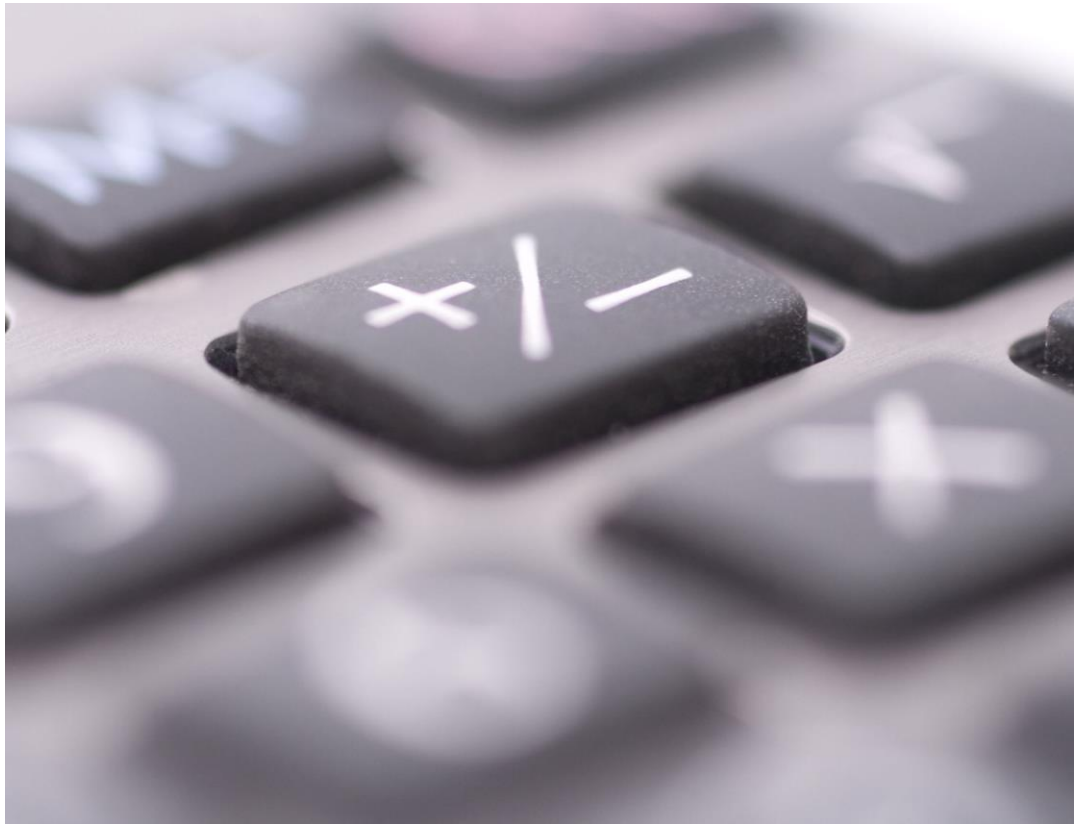
What if we insert (1998, Jan Ullrich, USA)?

EXERCISE 2NF?

Patients(StaffNo, ApptDate, ApptTime, DentistName, PatientNo, PatientName, SurgeryNo)

F={StaffNo, ApptDate, ApptTime → PatientNo, PatientName; StaffNo →DentistName; PatientNo → PatientName,SurgeryNo; StaffNo, ApptDate → SugeryNo; ApptDate, ApptTime, PatientNo →StaffNo, DentistName}

EXERCISE 2NF?



R(ABCDEGH)

F = {ABC → EG, A → D, E → GH, AB → H, BCE → AD}

SA: BC

IA: AE

Keys: ABC, BCE

R1(AD) F1 = {A → D}

R2(ABCEGH)

F2 = {ABC → EG, E → GH, AB → H, BCE → A}

Keys: ABC, BCE

R2(ABCEGH)

F2 = {ABC → EG, E → GH, AB → H, BCE → A}

Keys: ABC, BCE

R21(EGH) F21 = {E → GH}

R22(ABCE) F22 = {ABC → E}

2NF **AND** every non-prime attribute is non-transitively dependent on every key.

> *"A non-key field must provide a fact about the key, the whole key, and nothing but the key. So help me Codd."*

To fix, decompose into multiple relations, whereby the intermediate non-key attribute(s) functionally determine other non-prime attributes.

A table design is said to be in 3NF if both the following conditions hold:

- Table must be in 2NF
- Transitive functional dependency of non-prime attribute on any superkey should be removed.

An attribute that is not part of any candidate key is known as non-prime attribute.

In other words, 3NF can be explained like this:

A table is in 3NF if it is in 2NF, and for each functional dependency $X \rightarrow Y$ at least one of the following conditions holds:

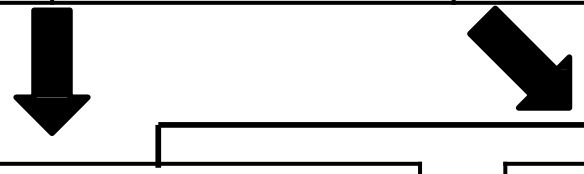
- X is a super key of the table
- Y is a prime attribute of the table

An attribute that is a part of one of the candidate keys is known as a prime attribute.

3NF EXAMPLE

F={Year → Winner, Nationality; Winner → Nationality}

Year	Winner	Nationality
1994	Miguel Indurain	Spain
1995	Miguel Indurain	Spain
1996	Bjarne Riis	Denmark
1997	Jan Ullrich	Germany



Year	Winner
1994	Miguel Indurain
1995	Miguel Indurain
1996	Bjarne Riis
1997	Jan Ullrich

Winner	Nationality
Miguel Indurain	Spain
Bjarne Riis	Denmark
Jan Ullrich	Germany

Suppose a company wants to store the complete address of each employee, they create a table named **employee_details** that looks like this:

Employees(Emp_id, Emp_Name, Emp_zip, Emp_state, Emp_city, Emp_district)

F={Emp_zip → Emp_state, Emp_city, Emp_district; Emp_id → Emp_zip; Emp_id → Emp_Name}

Only key is {Emp_id}

Non-prime attributes (Emp_state, Emp_city & emp_district) transitively dependent on superkey (Emp_id).

This violates the rule of 3NF.

To make this table complies with 3NF we have to break the table into two tables to remove the transitive dependency:

Employees_zip (Emp_zip, Emp_state, Emp_city, Emp_district)

F1={Emp_zip → Emp_state, Emp_city, Emp_district}

Employees (Emp_id, Emp_name, Emp_zip)

F2 ={Emp_id →Emp_zip, Emp_Name}

EXAMPLES: 3NF

A bank uses the following relation:

Vendors(ID, Name, Account_No, Bank_Code_No, Bank)

F={ID → Name, Account_No, Bank_Code_No; Bank_Code_No → Bank}

Only key is {ID}

EXAMPLES: 3NF

The non-prime attribute (Bank) transitively depends on the superkey (ID).

This violates the rule of 3NF.

To make this table complies with 3NF, we have to break the table into two tables to remove the transitive dependency:

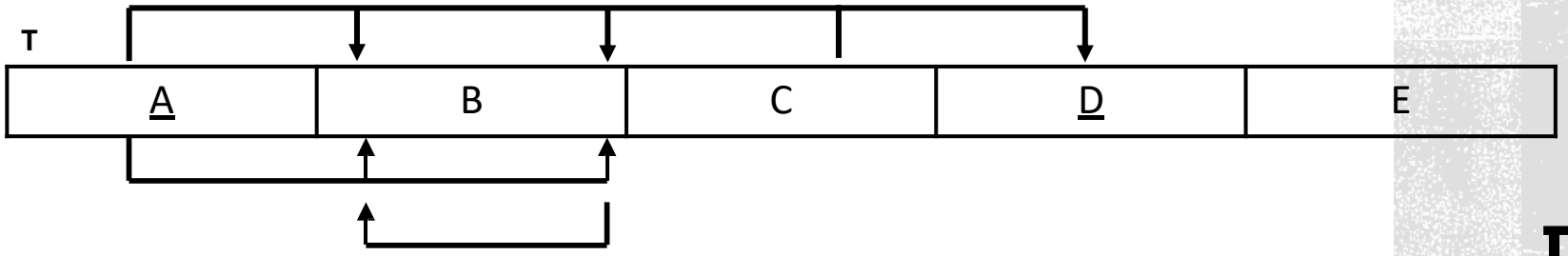
Vendors1(Bank_Code_No, Bank)

$F1=\{Bank\_Code\_No \rightarrow Bank\}$

Vendors2(ID, Name, Account_No, Bank_Code_No)

$F2=\{ID \rightarrow Name, Account\_No, Bank\_Code\_No\}$

EXERCISES: 3NF

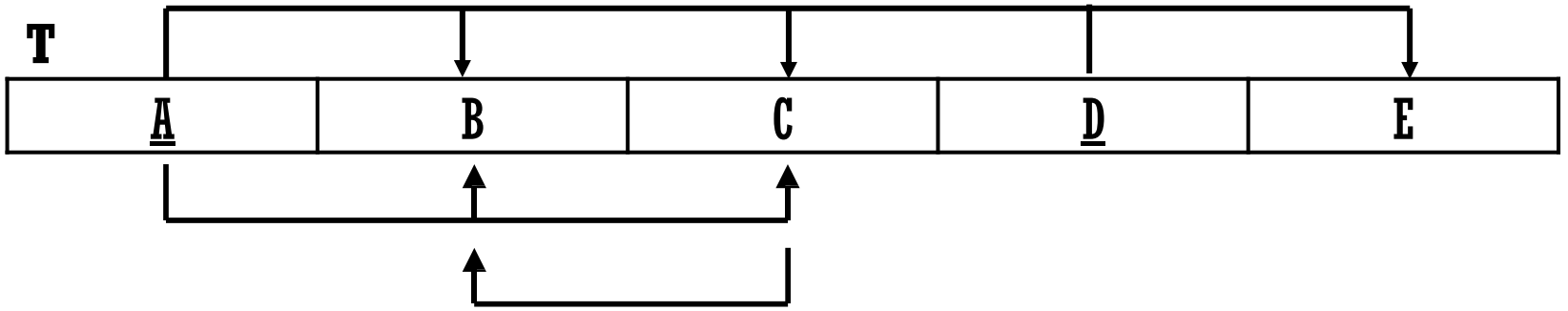


Consider the schema for relation T, as well as all FDs.
What is the normal form of T?

If T violates 3NF, provide a 3NF decomposition that satisfies the FDs (including the primary key) and does not produce spurious tuples.

Show and explain all steps of your analysis and decomposition (if applicable).

ANSWER (1)



List non-trivial FDs

$AD \rightarrow BCE$

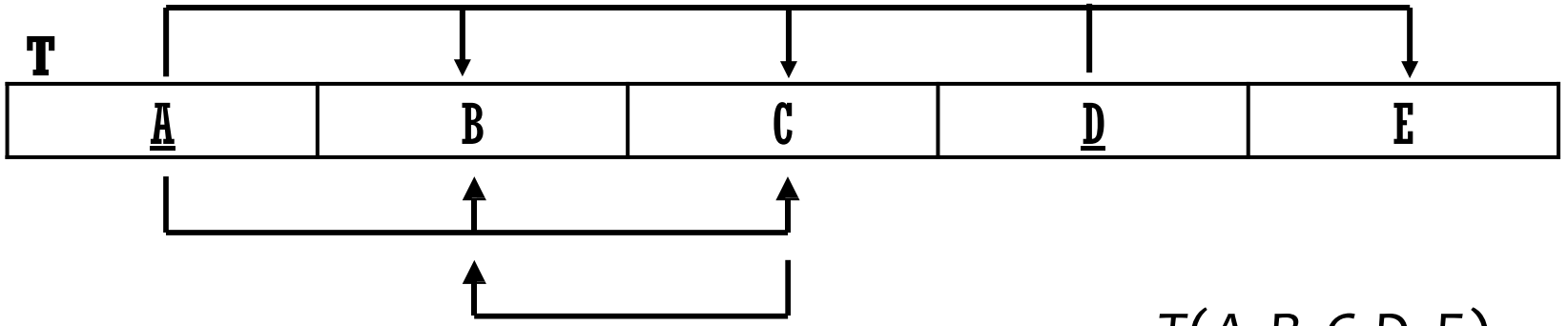
$A \rightarrow BC$

$C \rightarrow B$

Written algebraically

$T(\underline{A}, B, C, \underline{D}, E)$

ANSWER (2)



$T(\underline{A}, B, C, \underline{D}, E)$

$AD \rightarrow BCE$

$A \rightarrow BC$

$C \rightarrow B$

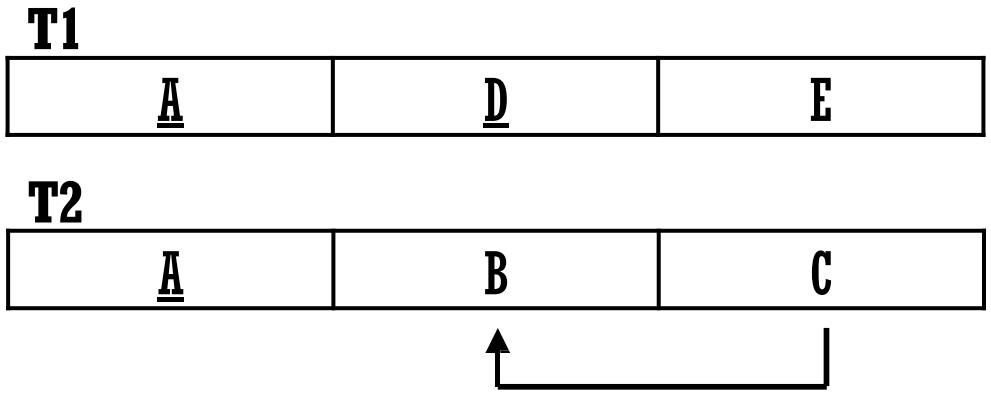
T is in … 1NF

Both B & C are FD on A

– Thus, not fully FD on PK (AD)

Decompose!

ANSWER (3)



$T1(\underline{A,D},E)$
$T2(\underline{A},B,C)$
$AD \rightarrow E$
$A \rightarrow BC$
$C \rightarrow B$

T1 is in….3NF

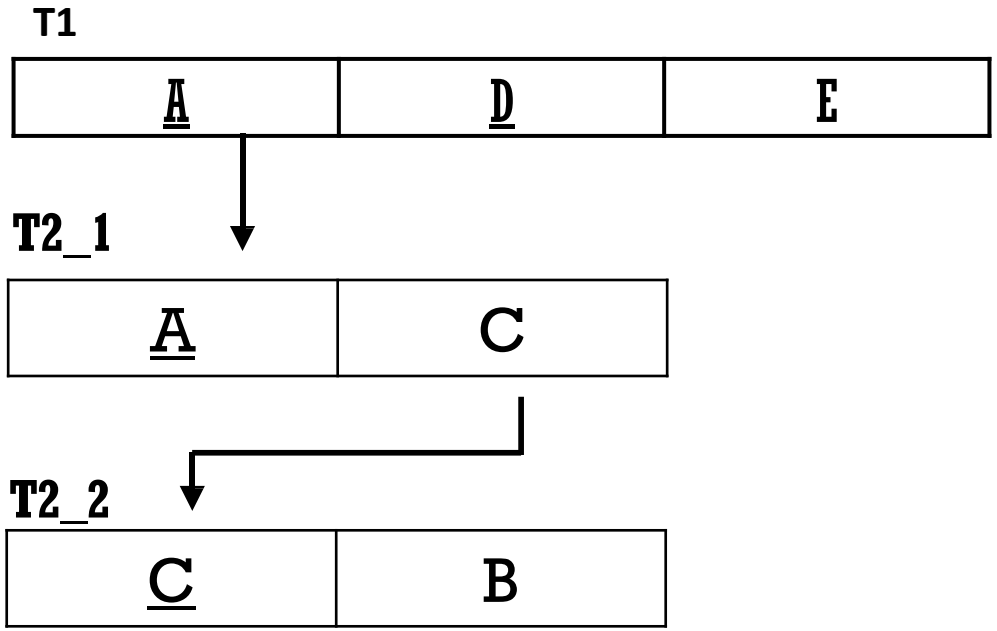
- 2NF: E is fully FD on AD
- 3NF: No transitive FDs (trivially true)

T2 is in …2NF

- 2NF: B and C fully FD on A (trivially true)
- !3NF: B is transitively FD on A [via C]

Decompose!

ANSWER (4)

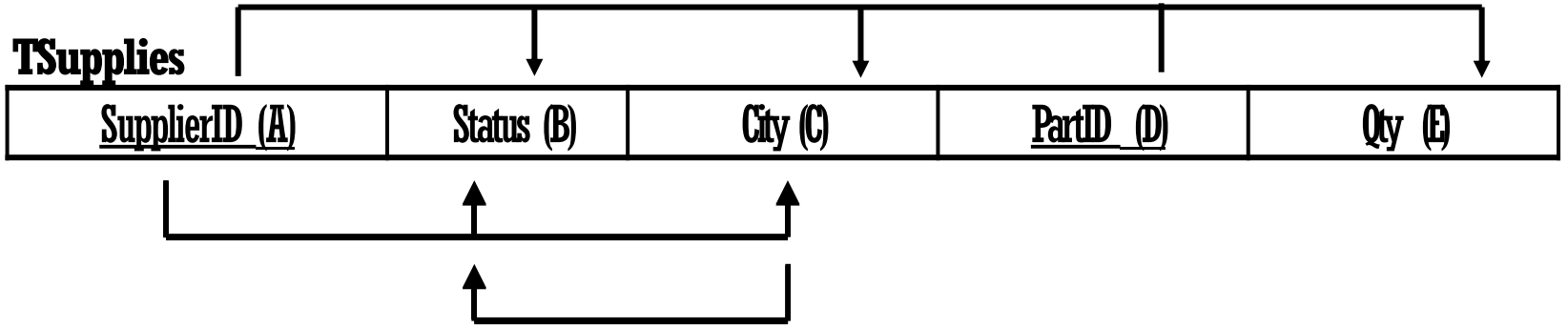


$T1(\underline{A}, \underline{D}, E)$
$T2\_1(\underline{A}, C)$
$T2\_2(\underline{C}, B)$
$AD \rightarrow E$
$A \rightarrow C$
$C \rightarrow B$

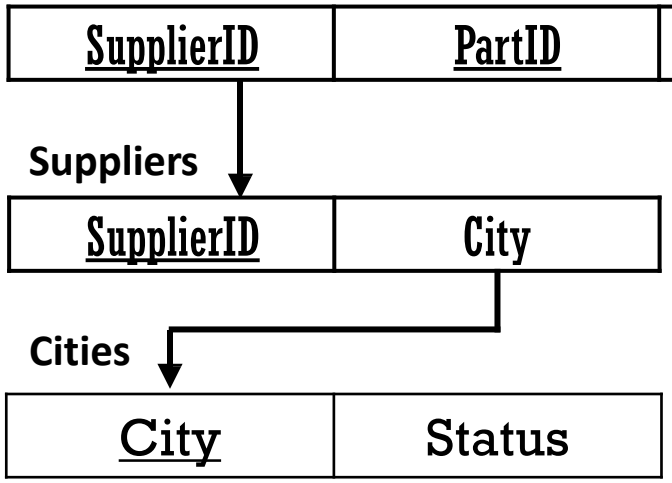
Database is in 3NF

- Why?

ANSWER (5)



Supplier_Parts



$\{SupplierID, PartID\} \rightarrow \{Qty\}$

$\{SupplierID\} \rightarrow \{City\}$

$\{City\} \rightarrow \{Status\}$

BOYCE-CODD NORMAL FORM (BCNF)

We say a relation *R* is in *BCNF* if whenever $X \rightarrow Y$ is a nontrivial FD that holds in *R*, *X* is a superkey

- Remember: *nontrivial* means *Y* is not contained in *X*
- Remember, a *superkey* is any superset of a key (not necessarily a proper superset)

BOYCE-CODD NORMAL FORM (BCNF)

It is an advance version of 3NF that's why it is also referred as 3NF. BCNF is stricter than 3NF.

A table complies with BCNF:

It is in 3NF and

for every functional dependency $X \rightarrow Y$, X should be the superkey of the table.

EXAMPLES: BCNF

Drinkers(name, addr, beersLiked, manf, favBeer)

FD's: F= {name → addr, favBeer;

beersLiked → manf}

Only key is {name, beersLiked}

In each FD, the left side is *not* a superkey

Anyone of these FD's shows *Drinkers* is not in BCNF

ANOTHER EXAMPLE

Beers(name, manf, manfAddr)

FD's: F={name → manf, manf → manfAddr}

Only key is {name}

Name → manf does not violate BCNF, but manf → manfAddr does

DECOMPOSITION INTO BCNF

Given: relation R with FD's F

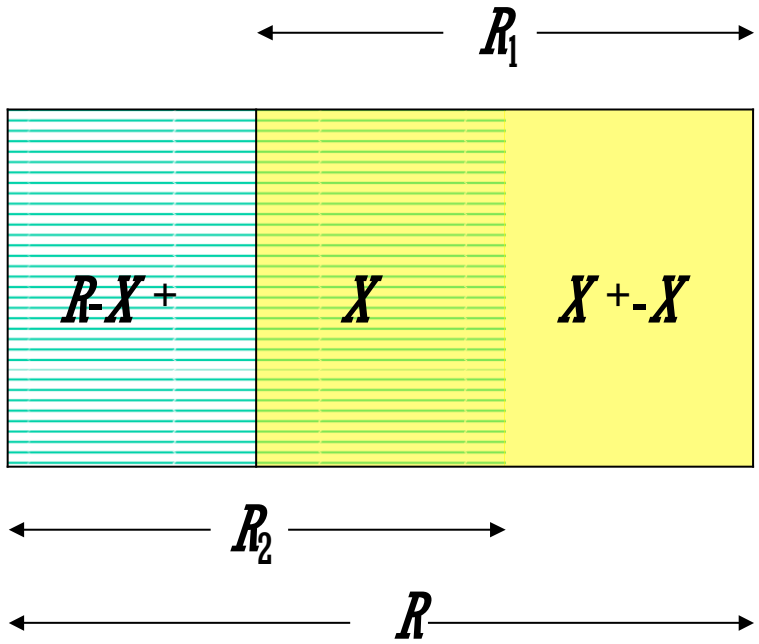
- Look among the given FD's for a BCNF violation $X \rightarrow Y$
  - If any FD following from F violates BCNF, then there will surely be an FD in F itself that violates BCNF
- Compute X^+
  - Not all attributes, or else X is a superkey

DECOMPOSE R USING $X \rightarrow Y$

Replace R by relations with schemas:

1. $R_1 = X^+$
2. $R_2 = R - (X^+ - X)$

Project given FD's F onto the two new relations



EXAMPLES: BCNF?

Let's take R = {A,B,C,D,E,G} and

F = {BC $\rightarrow$ D, CD $\rightarrow$ E}

Key is {A,B,C,G}

For example, we use FD: BC $\rightarrow$ D to decompose R into two relations R_1 and R_2

X = BC and X^+ = BCDE

R_1 = BCDE, R_2= ABCG

It means R_1 intersect R_2 = X

EXAMPLES: BCNF DECOMPOSITION

Drinkers(name, addr, beersLiked, manf, favBeer)

F= {name → addr, name → favBeers,
beersLiked → manf}

Pick BCNF violation name → addr

Close the left side:

{name}$^+$ = {name, addr, favBeer}

Decomposed relations:

1. Drinkers1(name, addr, favBeer)
2. Drinkers2(name, beersLiked, manf)

EXAMPLES: BCNF DECOMPOSITION

We are not done; we need to check Drinkers1 and Drinkers2 for BCNF

Projecting FD's is easy here

For Drinkers1(<u>name</u>, addr, favBeer), relevant FD's F = {name $\rightarrow$ addr, name $\rightarrow$ favBeer}

- Thus, {name} is the only key and Drinkers1 is in BCNF

EXAMPLES: BCNF DECOMPOSITION

For Drinkers2(name, beersLiked, manf),

the only FD is beersLiked → manf, and the only key is {name, beersLiked}

Violation of BCNF

$\{beersLiked\}^+ = \{beersLiked, manf\}$,

So we decompose *Drinkers2* into:

- Drinkers3(beersLiked, manf)
- Drinkers4(name, beersLiked)

EXAMPLES: BCNF DECOMPOSITION

The resulting decomposition of *Drinkers:*

1. Drinkers1(name, addr, favBeer)
2. Drinkers3(beersLiked, manf)
3. Drinkers4(name, beersLiked)

Notice: *Drinkers1* tells us about drinkers,

Drinkers3 tells us about beers, and

Drinkers4 tells us the relationship between drinkers and the beers they like

Compare with running example:

1. Drinkers(name, addr, phone)
2. Beers(name, manf)
3. Likes(drinker,beer)

EXERCISES: BCNF

Suppose there is a company wherein employees work in more than one department. They store the data like this:

Employees (Emp_id, Emp_Nationality, Emp_Dept, Dept_type, Dept_no_of_emp)

F = {Emp_id → Emp_Nationality; Emp_Dept → Dept_type, Dept_no_of_emp}

Only key is {Emp_id, Emp_dept}

BCNF– MOTIVATION

There is one structure of FD's that causes trouble when we decompose

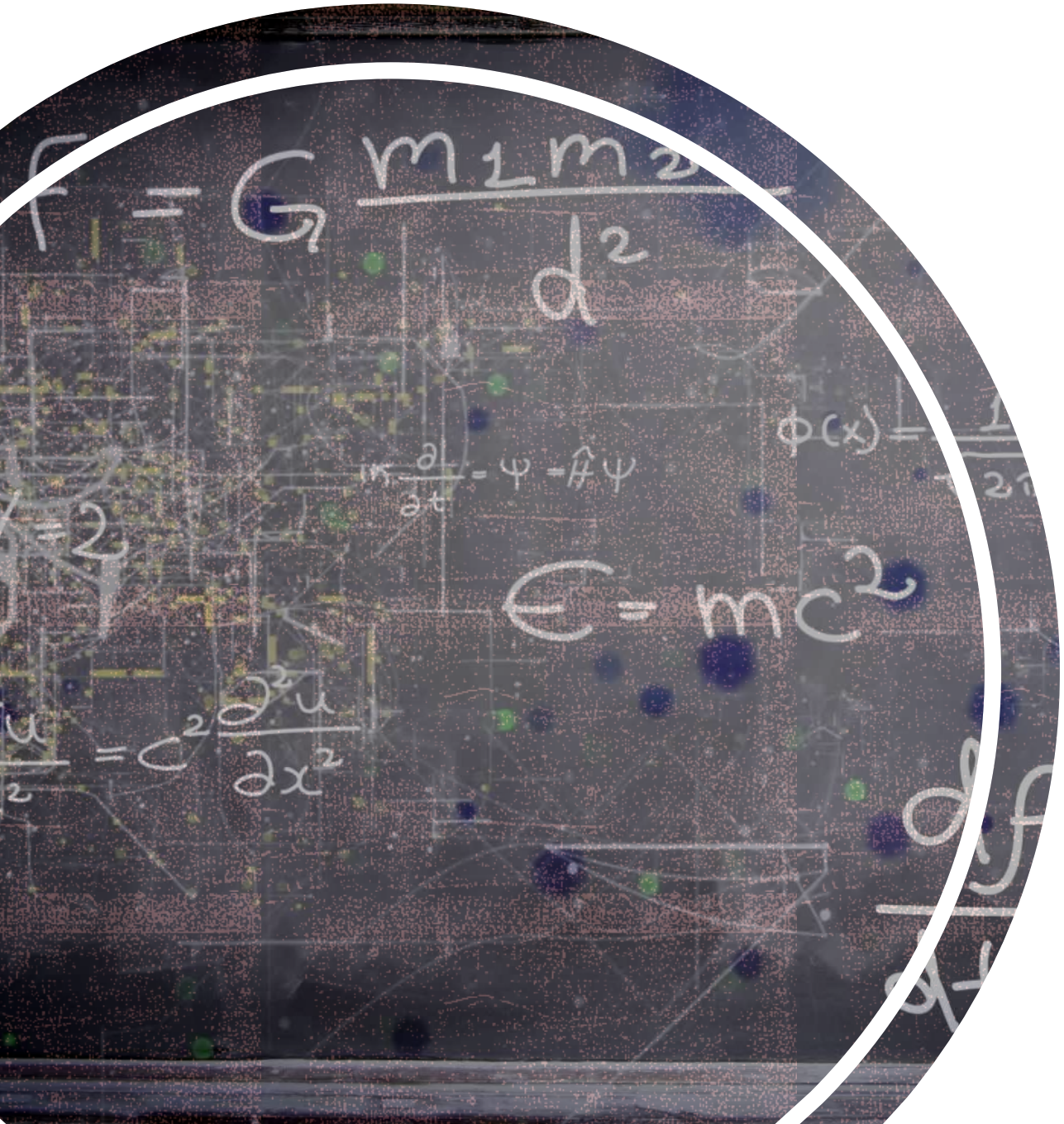
$AB \rightarrow C$ and $C \rightarrow B$

Example:

A = street address; B = city; C = post code

There are two keys, $\{A,B\}$ and $\{A,C\}$

$C \rightarrow B$ is a BCNF violation, so we must decompose into AC, BC



WE CANNOT ENFORCE FD'S

The problem is that if we use *AC* and *BC* as our database schema, we cannot enforce the FD $AB \rightarrow C$ by checking FD's in these decomposed relations

Example with *A* = street, *B* = city, and *C* = post code on the next slide

AN UNENFORCEABLE FD

street	post
Campusvej	5230
Vestergade	5000

city	post
Odense	5230
Odense	5000

Join tuples with equal post codes

street	city	post
Campusvej	Odense	5230
Vestergade	Odense	5000

No FD's were violated in the decomposed relations and FD street, city → post holds for the database as a whole

AN UNENFORCEABLE FD

street	post
Hjallesevej	5230
Hjallesevej	5000

city	post
Odense	5230
Odense	5000

Join tuples with equal post codes

street	city	post
Hjallesevej	Odense	5230
Hjallesevej	Odense	5000

Although no FD's were violated in the decomposed relations, FD street, city → post is violated by the database as a whole.

ANOTHER UNENFORCEABLE FD

Departures(time, track, train)

F={time, track → train; train → track}

Two keys, {time,track} and {time,train}

train → track is a BCNF violation, so we must decompose it into

Departures1(time, train)
Departures2(track,train)

ANOTHER UNENFORCEABLE FD

time	train
19:08	ICL54
19:16	IC852

tracktrain	
4	ICL54
3	IC852

Join tuples with equal train code

time	track	train
19:08	4	ICL54
19:16	3	IC852

No FD's were violated in the decomposed relations,
FD time, track → train holds for the database as a whole

ANOTHER UNENFORCEABLE FD

time	train
19:08	ICL54
19:08	IC 42

Tracktrain	Train
4	ICL54
4	IC 42

Join tuples with equal train code

time	track	train
19:08	4	ICL54
19:08	4	IC 42

Although no FD's were violated in the decomposed relations, FD time, track → train is violated by the database as a whole.

EXAMPLES: DECOMPOSITION INTO BCNF

1. Let's take R(ABCDE), and FD's

F = {A $\rightarrow$ BC, C $\rightarrow$ DE}

2. Let's take R(ABCD) and FD's

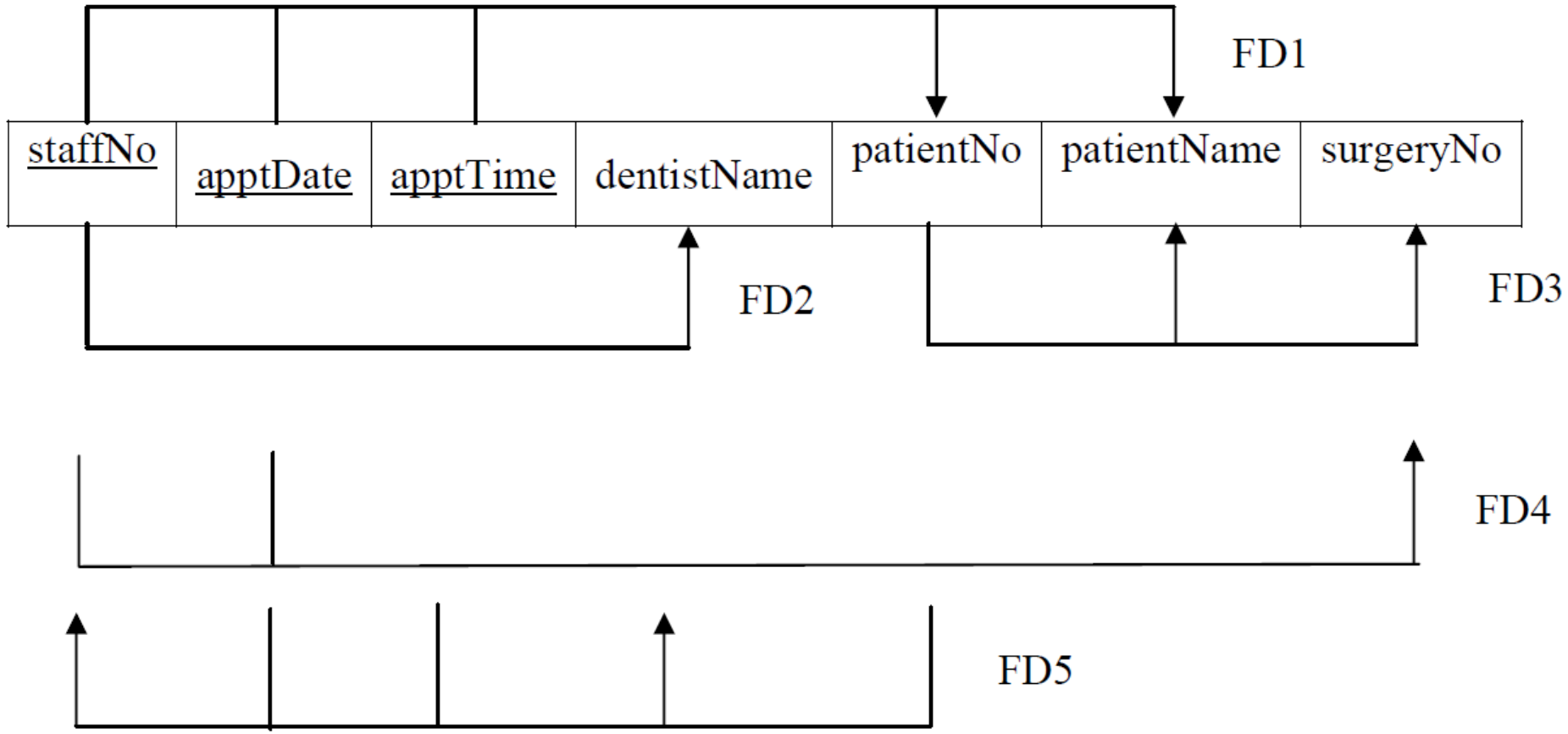
F = {AB $\rightarrow$ C, B $\rightarrow$ D; C $\rightarrow$ A}

EXERCISE

The table shown in Figure below is susceptible to update anomalies. Provide examples of insertion, deletion, and modification anomalies. Decomposition step by step to achieve BCNF if it has not yet achieved BCNF.

Staff No	Dentist Name	patient No	Patient Name	Appointment		Surgery No
				Date	time	
S1011	Tony Smith	P100	Gillian White	12-Aug-03	10.00	S10
S1011	Tony smith	P105	Jill Bell	13-Aug-03	12.00	S15
S1024	Helen Pearson	P108	Ian Mackay	12-Sep-03	10.00	S10
S1024	Helen Pearson	P108	Ian Mackay	14-Sep-03	10.00	S10
S1032	Robin Plevin	P105	Jill Bell	14-Oct-03	16.30	S15
S1032	Robin Plevin	P110	John Walker	15-Oct-03	18.00	S13

EXERCISE


FD1
staffNo
apptDate
apptTime
dentistName
patientNo
patientName
surgeryNo
FD2
FD3
FD4
FD5

MULTIVALUED DEPENDENCIES

A multivalued dependency (MVD) has the from
X ↠ Y, where X and Y are sets of attributes in a relation R.

X ↠ Y means that whenever two rows in R agree on all the attributes of X, then we can swap their Y components and get two rows that are also in R

X	Y	Z
a	b1	c1
a	b2	c2
a	b2	c1
a	b1	c2
…	…	…

MVD EXAMPLES

User (uid, gid, place)

- uid ↠ gid
- uid ↠ place
  - Intuition: given uid, gid and place are "independent"
- uid, gid ↠ place
  - Trivial: LHS ∪ RHS = all attributes of R
- uid, gid ↠ uid
  - Trivial: LHS ⊇ RHS

COMPLETE MVD + FD RULES

MVD complementation:	If $X \twoheadrightarrow Y$, then $X \twoheadrightarrow attrs\ R - X - Y$
MVD augmentation:	If $X \twoheadrightarrow Y$ and $V \subseteq W$, then $XW \twoheadrightarrow YV$
MVD transitivity:	If $X \twoheadrightarrow Y$ and $Y \twoheadrightarrow Z$, then $X \twoheadrightarrow Z - Y$

Replication (FD is MVD):

If $X \rightarrow Y$, then $X \twoheadrightarrow Y$

Coalescence:

If $X \twoheadrightarrow Y$ and $Z \subseteq Y$ and there is some W disjoint from Y such that $W \rightarrow Z$, then $X \rightarrow Z$

Given a set of FD's and MVD's $\mathcal{D}$, does another dependency d (FD or MVD) follow from $\mathcal{D}$?

Procedure

- Start with the "if-part" of d, and treat them as "seed" tuples in a relation
- Apply the given dependencies in $\mathcal{D}$ repeatedly
  - If we apply an FD, we infer equality of two symbols
  - If we apply an MVD, we infer more tuples

An elegant solution: chase

If we infer the "then-part" of d, we have a proof

Otherwise, if nothing more can be inferred, we have a counterexample

PROOF BY CHASE

In R(A, B, C, D), does A ↠ B and B ↠ C imply that
A ↠ C?

Have:

A	B	C	D
a	b_1	c_1	d_1
a	b_2	c_2	d_2

$A \twoheadrightarrow B$

a	b_2	c_1	d_1
a	b_1	c_2	d_2

$B \twoheadrightarrow C$

a	b_2	c_1	d_2
a	b_2	c_2	d_1

$B \twoheadrightarrow C$

a	b_1	c_2	d_1
a	b_1	c_1	d_2

Need:

A	B	C	D	
a	b_1	c_2	d_1	✌
a	b_2	c_1	d_2	✌

ANOTHER PROOF BY CHASE

In R(A, B, C, D), does A → B and B → C imply that A → C?

A → B	$b_1 = b_2$
B → C	$c_1 = c_2$

Have:

A	B	C	D
a	b_1	c_1	d_1
a	b_2	c_2	d_2

Need: $c_1 = c_2$ ✌

In general, with both MVD's and FD's, chase can generate both new tuples and new equalities

COUNTEREXAMPLE BY CHASE

- In R(A, B, C, D), does $A \twoheadrightarrow BC$ and $CD \rightarrow B$ imply that $A \rightarrow B$?

Have:

A	B	C	D
a	b_1	c_1	d_1
a	b_2	c_2	d_2

Need: $b_1 = b_2$ 👎

$A \twoheadrightarrow BC$

a	b_2	c_2	d_1
a	b_1	c_1	d_2

Counterexample!

4NF

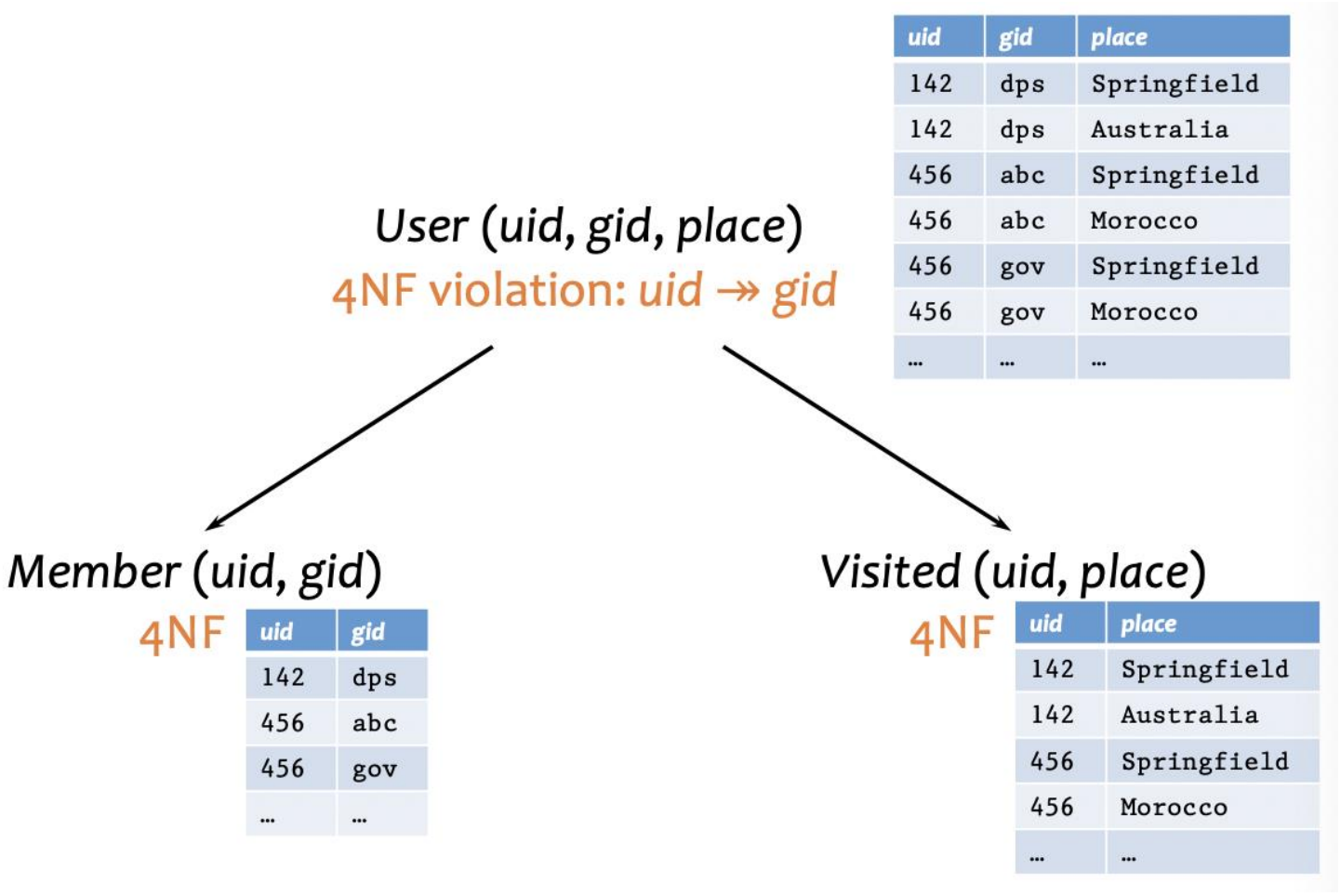
A relation R is in Fourth Normal Form (4NF) if

- For every non-trivial MVD $X \twoheadrightarrow Y$ in R, X is a superkey
- That is, all FD's and MVD's follow from "key $\rightarrow$ other attributes" (i.e., no MVD's and no FD's besides key functional dependencies)
- 4NF is stronger than BCNF, because every FD is also an MVD

4NF DECOMPOSITION ALGORITHM

- Find a 4NF violation: A non-trivial MVD $X \twoheadrightarrow Y$ in R where X is not a superkey
- Decompose R into R_1 and R_2, where
  - R1 has attributes X Y
  - R2 has attributes X Z (where Z contains R attributes not in X or Y)
- Repeat until all relations are in 4NF
- Almost identical to BCNF decomposition algorithm
- Any decomposition on a 4NF violation is lossless

4NF Decomposition Example



uid	gid	place
142	dps	Springfield
142	dps	Australia
456	abc	Springfield
456	abc	Morocco
456	gov	Springfield
456	gov	Morocco
…	…	…

uid	gid
142	dps
456	abc
456	gov
…	…

uid	place
142	Springfield
142	Australia
456	Springfield
456	Morocco
…	…

SUMMARY

- Philosophy behind BCNF, 4NF: Data should depend on the key, the whole key, and nothing but the key!
  - You could have multiple keys though
- Other normal forms
  - 3NF: More relaxed than BCNF; will not remove redundancy if doing so makes FDs harder to enforce
  - 2NF: Slightly more relaxed than 3NF
  - 1NF: All column values must be atomic

SUMMARY

- Normalization is the theory and process by which to evaluate and improve relational database design
  - Makes the schema informative
  - Minimizes information duplication
  - Avoids modification anomalies
  - Disallows spurious tuples

- Make sure all your relations are *at least* 3NF!
  - Higher normal forms exist
  - We may reduce during physical design

Thank you for your attention!