ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Lesson 5
Database design

Part 2: Normalization

Objective

- ***Upon completion of this lesson, students will be able to:***

    1. Know why we need normalization in relational DB
    2. Identify normal forms such as 1st NF, 2nd NF, 3rd NF
    3. Know how to normalize a relational DB into 3NF

Outline

- Introduction
- Normal Forms
- Normalization

1. Introduction

1.1. Motivation

1.2. Full & Partial Dependency

1.3. Transitive Dependency

1.1. Motivation

- Designing DB: one of the most difficult tasks
- One simplest design approach is to use a big table and store all data
- But what's the problem with this?
  - Anomalies
  - Redundancies

1.1. Motivation: Insertion Anomalies

- PK: (student_id, subject_id)
- We can not insert a new subject if we do not have a student assigned to it yet
- We can not insert a null value into PK attributes

student_id	full_name	dob	subject_id	name	result
1234	David Beckham	12/21/1997	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4843	Data integration	B
1234	David Beckham	12/21/1997	IT4868	Web mining	C
1497	Tony Blair	03/01/1999	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4868	Web mining	B
1542	Margaret Thatcher	05/08/1997	IT2000	Introduction to ICT	C

1.1. Motivation: Update anomalies

- An instance where the same information must be updated in several different places
- If you update the name of subject "**Databases**", you need to update in two different places (not efficient)

student_id	full_name	dob	subject_id	name	result
1234	David Beckham	12/21/1997	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4843	Data integration	B
1234	David Beckham	12/21/1997	IT4868	Web mining	C
1497	Tony Blair	03/01/1999	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4868	Web mining	B
1542	Margaret Thatcher	05/08/1997	IT2000	Introduction to ICT	C

1.1. Motivation: Deletion Anomalies

- Where deleting one piece of data inadvertently causes other data to be lost
- If we delete student **Margaret Thatcher**, then we will lose information about subject "**Introduction to ICT**"

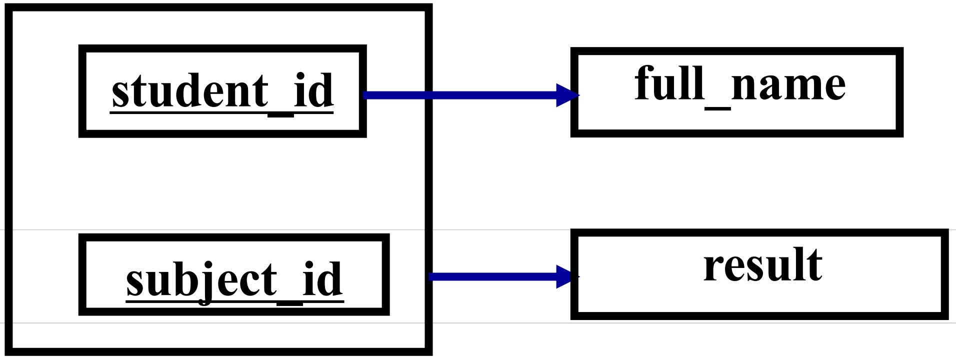
<u>student_id</u>	full_name	dob	<u>subject_id</u>	name	result
1234	David Beckham	12/21/1997	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4843	Data integration	B
1234	David Beckham	12/21/1997	IT4868	Web mining	C
1497	Tony Blair	03/01/1999	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4868	Web mining	B
1542	Margaret Thatcher	05/08/1997	IT2000	Introduction to ICT	C

1.1. Motivation

- Normalization is the process of removing **anomalies** and **redundancies** from DB

1.2. Full & Partial Dependency

<u>student_id</u>	full_name	dob	<u>subject_id</u>	name	result
1234	David Beckham	12/21/1997	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4843	Data integration	B
1234	David Beckham	12/21/1997	IT4868	Web mining	C
1497	Tony Blair	03/01/1999	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4868	Web mining	B
1542	Margaret Thatcher	05/08/1997	IT2000	Introduction to ICT	C



Key: {<u>student_id, subject_id</u>}

Full Key Dependency:

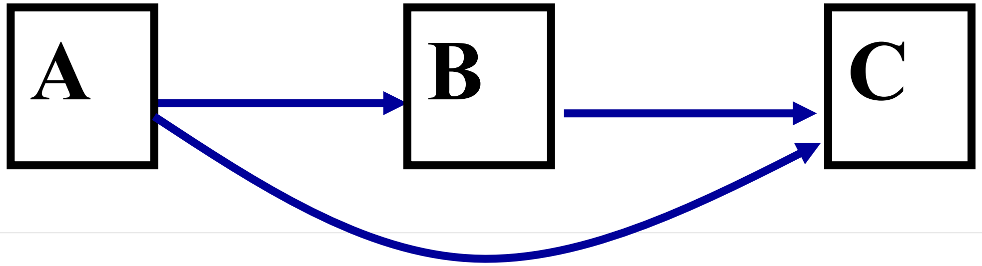
{student_id, subject_id} → result

Partial Key Dependency:

student_id → full_name

1.3. Transitive dependency

- If $A \rightarrow B$ and $B \rightarrow C$
  - Attribute A must be the determinant of C.
  - Attribute A transitively determines attribute C or
  - C is transitively dependent on A



2. Normal Forms

2.1. Introduction

2.2. 1st Normal Form

2.3. 2nd Normal Form

2.4. 3rd Normal Form

2.1. Introduction

- Each form was designed to eliminate one or more of the anomalies: First NF; Second NF; Third NF
- Unnormalized Form (UNF)
    - A table that contains one or more repeating groups. I.e., its cell may contain multiple values

Multi Value
Or repeating groups

<u>student_id</u>	full_name	dob	subject_id	name	result
1234	David Beckham	12/21/1997	IT3090, IT4868	Databases, Web mining	A, C
1238	Theresa May	08/06/1998	IT4843, IT4868	Data integration, Web mining	B, B
1497	Tony Blair	03/01/1999	IT3090	Databases	A
1542	Margaret Thatcher	05/08/1997	IT2000	Introduction to ICT	C

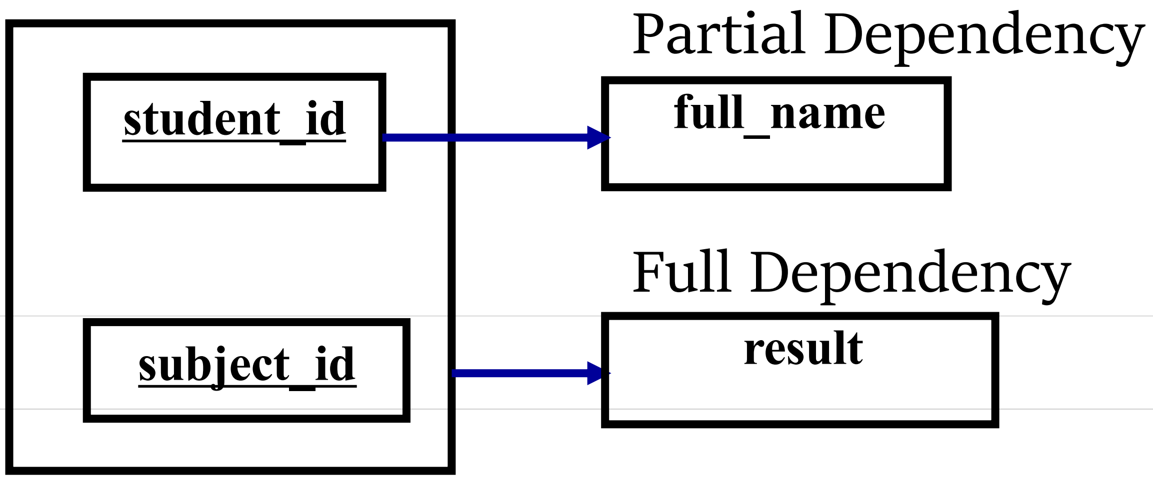
2.2. First Normal Form (1NF)

- A cell in a relation contains one and only one value.
  - Disallows composite attributes, multivalued attributes or nested relations

<u>student_id</u>	full_name	dob	<u>subject_id</u>	name	result
1234	David Beckham	12/21/1997	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4843	Data integration	B
1234	David Beckham	12/21/1997	IT4868	Web mining	C
1497	Tony Blair	03/01/1999	IT3090	Databases	A
1238	Theresa May	08/06/1998	IT4868	Web mining	B
1542	Margaret Thatcher	05/08/1997	IT2000	Introduction to ICT	C

2.3. Second Normal Form (2NF)

- Based on the concept of full functional dependency
- A prime attribute
  - It is an attribute that is member of some candidate key
- 2NF relation is
  - in 1NF and every non-prime attribute is fully functionally dependent on the primary key



2.4. Third Normal Form (3NF)

- A relation that is in 2NF
  - No non-prime attribute is transitively dependent on the primary key.
  - I.e, all non-prime attributes are fully & directly dependent on the PK.

3. Normalization

3.1. Properties of relational decompositions

3.2. An algorithm decomposes a universal relation into 3NF

3.3. Some examples

3.1. Properties of relational decompositions

- A single universal relation schema $R = \{A_1, A_2, ..., A_n\}$ that includes all the attributes of the DB
- F is a set of FDs holds on R
- Using the FDs, the algorithms decompose the universal relation schema R into a set of relation schemas $D = \{R_1, R_2, ..., R_m\}$; D is called **a decomposition of R**

3.1. Properties of relational decompositions

- Attribute preservation
  - Each attribute in *R* will appear in at least one relation schema R_i in the decomposition so that no attributes are *lost*
- Dependency preservation
  - Each FD $X \rightarrow Y$ specified in F either appeared directly in one of the R_i in the decomposition D or could be inferred from the dependencies that appear in some R_i.
- Lossless join
  - $r = \Pi_{R1}(r) \bowtie \Pi_{R2}(r) \bowtie \ldots \bowtie \Pi_{Rm}(r)$

3.1. Properties of relational decompositions

- An example
  - Suppose we have a relation:

    Learn(<u>student_id</u>, full_name, dob, <u>subject_id</u>, name, result)
  - We split it into two relations:

    Student(<u>student_id</u>, full_name, dob)

    Subject(<u>subject_id</u>, name)
  - This decomposition does not warrant:
    - Attribute preservation: Lost information about "result"
    - Dependency preservation condition, for instance, (student_id, subject_id) $\rightarrow$ result is loss.
    - Lossless join property, i.e., we can join these two relations

3.2. An algorithm decomposes a universal relation into 3NF

- **Input**: A universal relation R and a set of FDs F on the attributes of R.
  - Find a minimal cover G for F
  - For each left-hand-side X of a FD that appears in G, create a relation schema in D with attributes $\{X \cup \{A_1\} \cup \{A_2\} \dots \cup \{A_k\}\}$, where $X \rightarrow A_1$, $X \rightarrow A_2$, ..., $X \rightarrow A_k$ are the only dependencies in G with X as the left-hand-side (X is the key of this relation);
  - Place any remaining attributes (that have not been placed in any relation) in a single relation schema to ensure the attribute preservation property.

3.2. An algorithm decomposes a universal relation into 3NF

- 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.
- Eliminate redundant relations from the resulting set of relations in the relational database schema.
  - A relation R is considered redundant if R is a projection of another relation S in the schema; alternately, R is subsumed by S

3.3. Some examples

- **Example 1:**
  - Given R = {A, B, C, D, E, F, G}, F = {$A \rightarrow B$; $ABCD \rightarrow E$; $EF \rightarrow G$; $ACDF \rightarrow EG$}
  - A minimal cover of F is G = {$A \rightarrow B$, $ACD \rightarrow E$, $EF \rightarrow G$}
  - Find a minimal key: K = ACDF
  - We have R_1(AB), R_2(ACDE), R_3(EFG)
  - Since K is not a subset of R_i, we have a new relation R_4(ACDF)
  - In conclusion, we have a decomposition D = {R_1, R_2, R_3, R_4}

3.3. Some examples

- **Example 2:**
  - Given R(student_id, name, birthday, advisor, department, semester, course, grade)
  - F = { student_id → (name, birthday); advisor → department; (student_id, semester, course) → (grade, advisor, department)}
  - We denote like this: student_id (A), name (B), birthday (C), advisor (D), department (E), semester (F), course (G), grade (H)
  - F is rewritten as {A →BC; D →E; AFG →HDE}
  - A minimal cover of F is G = {A→B; A →C; D →E; AFG →DH}
  - Find a minimal key: K = AFG
  - We have R_1(ABC), R_2(DE), R_3(AFGDH)
  - Since K is a subset of R_3, we have a decomposition D = {R_1, R_2, R_3} or {R1(student_id, name, birthday), R_2 (advisor, department), R_3 (student_id, semester, course, advisor, grade)}

Remark

- Motivation of normalization
- Full & Partial Dependency
- Transitive dependency
- 1NF, 2 NF, 3 NF
- Properties of relational decompositions
- An algorithm decomposes a universal relation into 3NF

Summary

1. **Introduction**
   - Normalization is the process of removing anomalies and redundancies from DB
   - Full & Partial Dependency
   - Transitive dependency
2. **Normal Forms**
   - 1NF, 2NF, 3NF
3. **Normalization**
   - Properties of relational decompositions
   - An algorithm decomposes a universal relation into 3NF
   - Some examples

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attention!

soict.hust.edu.vn/ fb.com/groups/soict

References

- Raghu Ramakrishnan and Johannes Gehrke, Database Management Systems, 3rd edition, Mc Graw Hill, 2003.
- Elmasri and Navathe, Fundamentals of Database Systems, 6th edition, Addison-Wesley, 2011.