

# INFO20003 Database Systems

---

Week 9

# Anomalies

- Update anomaly
- Deletion anomaly
- Insertion anomaly

# Update anomaly

- **Data inconsistency** due to data redundancy and partial update

CourseNumber	Tutor	Room	Seats
INFO20003	Farah	Alice Hoy 109	30
COMP10001	Farah	EDS 6	25
INFO30005	Patrick	Sidney Myer G09	20
COMP20005	Alan	Sidney Myer G09	20

- E.g. suppose the room Sidney Myer G09 now has 30 seats
- we will have to update all rows where room = Sidney Myer G09

# Deletion anomaly

- an **unintentional loss of certain attribute values** due to the deletion of other data for other attributes

CourseNumber	Tutor	Room	Seats
INFO20003	Farah	Alice Hoy 109	30
COMP10001	Farah	EDS 6	25
INFO30005	Patrick	Sidney Myer G09	20
COMP20005	Alan	Sidney Myer G09	20

- E.g. If we remove COMP10001 from the above table, the details of room EDS 6 are also deleted

# Insertion anomaly

- the **inability to add certain attributes** to a database due to absence of other attributes

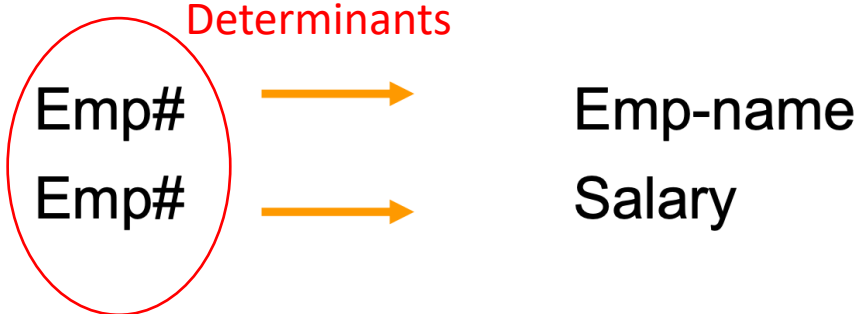
CourseNumber	Tutor	Room	Seats
INFO20003	Farah	Alice Hoy 109	30
COMP10001	Farah	EDS 6	25
INFO30005	Patrick	Sidney Myer G09	20
COMP20005	Alan	Sidney Myer G09	20

- E.g. a new room “NewAlice109” has been built but has not yet been timetabled for any course or members of staff

# Functional dependency (FD)

- Occurs when a set of attributes  $A_i$  determines attributes  $B_i$

- Written  $A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_n$

- e.g. 

The diagram illustrates functional dependencies. On the left, a red circle encloses the attributes 'Emp#' and 'Emp#'. Two orange arrows originate from this circle: one points to 'Emp-name' and the other points to 'Salary'. Above the arrows, the word 'Determinants' is written in red, indicating that the attributes inside the circle determine the attributes on the right.

- A relation  $R$  satisfies a FD if and only if the FD is true for every instance of  $R$

# Key and non-key attribute

- A key is a **minimal** set of attributes that can **functionally determine all other attributes** of R
- E.g. **Person (ssn, name, birthdate, address, age)**
  - Ssn is a key
  - {ssn, name} is not (it's a super key)

# Functional Dependency

- Partial Functional Dependency
- Transitive Functional Dependency



# Partial functional dependency

- arises when one or more non-key attributes are functionally **determined by a subset of the key**
- E.g.  $R(\underline{A}, \underline{B}, C, D)$ 
  - $\{A, B\}$  is a key
  - Partial functional dependency:  $B \rightarrow D$
  - To resolve this:
    - $R(\underline{A}, \underline{B(FK)}, C)$
    - $R2(\underline{B}, D)$

# Transitive functional dependency

- When a non-key attribute is **determined by another non-key attribute** (or by a combination of key and non-key attributes)
- E.g. R (A, B, C, D)
  - {A, B} is a key
  - Transitive functional dependency:  $C \rightarrow D$
  - To resolve this:
    - R(A, B, C (FK))
    - R2(C, D)

# Armstrong's Axioms

$A = \{A1, A2, \dots, An\}$

$B = \{B1, B2, \dots, Bn\}$

$C = \{C1, C2, \dots, Cn\}$

- **Reflexivity:**  $B \subseteq A \Rightarrow A \rightarrow B$

$ssn, name \rightarrow name$

- **Augmentation:**  $A \rightarrow B \Rightarrow AC \rightarrow BC$

$ssn, name, age \rightarrow name, age$

- **Transitivity:**  $A \rightarrow B \text{ and } B \rightarrow C \Rightarrow A \rightarrow C$

$ssn \rightarrow birthdate, birthdate \rightarrow age \Rightarrow ssn \rightarrow age.$

# Normalisation and normal forms

- Normalisation is a technique used to improve relations to **remove undesired redundancy** by **decomposing relations** and **eliminating anomalies**
- Performed in stages generally referred to as Normal Forms
- **First Normal Form** (1NF): all *repeating groups* are identified to be decomposed into new relations
- **Second Normal Form** (2NF): all the *partial dependencies* are resolved
- **Third Normal Form** (3NF): all the *transitive dependencies* are resolved

# 1NF

StudentCourse(StudentID, studentName, (CourseID, enrolmentDate, score))

<u>StudentID</u>	StudentName	<u>CourseID</u>	EnrolmentDate	Score
1	Aa	INFO20003	01/03/2024	90
		COMP10001	05/03/2024	89
		COMP30023	07/03/2024	95
2	Bb	INFO20003	28/02/2024	92
		COMP10001	01/03/2024	77

## Normalize into 1NF

- Student(StudentID, studentName)
- StudentCourse(StudentID, CourseID, EnrolmentDate, Score)

# Q1) Anomalies

1. Consider the relation Diagnosis with the schema Diagnosis (DoctorID, DocName, PatientID, DiagnosisClass) and the following functional dependencies:

$\text{DoctorID} \rightarrow \text{DocName}$

$\text{DoctorID, PatientID} \rightarrow \text{DiagnosisClass}$

Consider the following instance of Diagnosis:

DoctorID	DocName	PatientID	DiagnosisClass
D001	Alicia	P888	Flu
D002	John	P999	Lactose intolerance
D003	Jennifer	P000	Flu
D002	John	P111	Fever

Identify different anomalies that can arise from this schema using the above instance.

Give an example of each anomaly with respect to this case study

# Insertion anomaly

<b>DoctorID</b>	<b>DocName</b>	<b>PatientID</b>	<b>DiagnosisClass</b>
D001	Alicia	P888	Flu
D002	John	P999	Lactose intolerance
D003	Jennifer	P000	Flu
D002	John	P111	Fever

- Cannot insert data for a new doctor unless there is at least 1 patient associated with the doctor

# Deletion anomaly

DoctorID	DocName	PatientID	DiagnosisClass
D001	Alicia	P888	Flu
D002	John	P999	Lactose intolerance
D003	Jennifer	P000	Flu
D002	John	P111	Fever

- Deleting patient's data can result in the loss of doctor's data.
- E.g. if we delete P888 data from the table we lose record for the doctor named Alicia



# Update anomaly

DoctorID	DocName	PatientID	DiagnosisClass
D001	Alicia	P888	Flu
D002	John	P999	Lactose intolerance
D003	Jennifer	P000	Flu
D002	John	P111	Fever

- One doctor may be associated with more than one patient, an update anomaly may result if a doctor's name is changed for only one patient
- E.g. change the doctor's name from "John" to "John Miller"

## Q2) Normalisation

2. Consider a relation  $R(A, B, C, D)$  with the following FDs:

$$AB \rightarrow C, AC \rightarrow B, BC \rightarrow A, B \rightarrow D$$

The possible candidate keys of  $R$  are  $AB$ ,  $AC$ , and  $BC$ , since each of those combinations is sufficient to uniquely identify each record. Let's consider  $AB$  for instance. From  $AB \rightarrow C$  we see that  $AB$  uniquely identifies  $C$ , and since  $B$  alone uniquely identifies  $D$ ,  $AB$  together have covered  $CD$ , i.e. the entire set of attributes.

List all the functional dependencies that violate 3NF. If any, decompose  $R$  accordingly. After decomposition, check if the resulting relations are in 3NF, if not decompose further.

## Q2) Normalisation

2. Consider a relation  $R(A, B, C, D)$  with the following FDs:

$$AB \rightarrow C, AC \rightarrow B, BC \rightarrow A, B \rightarrow D$$

The possible candidate keys of  $R$  are  $AB$ ,  $AC$ , and  $BC$ , since each of those combinations is sufficient to uniquely identify each record. Let's consider  $AB$  for instance. From  $AB \rightarrow C$  we see that  $AB$  uniquely identifies  $C$ , and since  $B$  alone uniquely identifies  $D$ ,  $AB$  together have covered  $CD$ , i.e. the entire set of attributes.

List all the functional dependencies that violate 3NF. If any, decompose  $R$  accordingly. After decomposition, check if the resulting relations are in 3NF, if not decompose further.

Taking  $\{A, B\}$  as the key  
 $B \rightarrow D$  is a partial functional dependency

$R_1(\underline{A}, \underline{B}, C)$

$R_2(\underline{B}, D)$

# Q3) Normalisation

3. Consider the following relation StaffPropertyInspection:

StaffPropertyInspection (propertyNo, pAddress, iDate, iTime, comments, staffNo, sName)

The FDs stated below hold for this relation:

propertyNo, iDate  $\rightarrow$  iTime, comments, staffNo, sName

propertyNo  $\rightarrow$  pAddress

staffNo  $\rightarrow$  sName

From these FDs, it is safe to assume that propertyNo and iDate can serve as a primary key. Your task is to normalise this relation to 3NF. Remember in order to achieve 3NF, you first need to achieve 1NF and 2NF.

# Q3)

StaffPropertyInspection (propertyNo, pAddress, iDate, iTime, comments, staffNo, sName)

The FDs stated below hold for this relation:

propertyNo, iDate  $\rightarrow$  iTime, comments, staffNo, sName

propertyNo  $\rightarrow$  pAddress

staffNo  $\rightarrow$  sName

- 1NF: already in 1NF



# Q3)

StaffPropertyInspection (propertyNo, pAddress, iDate, iTime, comments, staffNo, sName)

The FDs stated below hold for this relation:

propertyNo, iDate  $\rightarrow$  iTime, comments, staffNo, sName

propertyNo  $\rightarrow$  pAddress

$$\text{staffNo} \rightarrow \text{sName}$$

- 2NF: Property (propertyNo, pAddress)

FK  
PropertyInspection (propertyNo, iDate, iTime, comments, staffNo, sName)

# Q3)

StaffProperty Inspection (propertyNo, pAddress, iDate, iTime, comments, staffNo, sName)

The FDs stated below hold for this relation:

propertyNo, iDate  $\rightarrow$  iTime, comments, staffNo, sName

propertyNo  $\rightarrow$  pAddress

staffNo  $\rightarrow$  sName

• 3NF:    Property (propertyNo, pAddress)

Staff (staffNo, sName)

Property Inspection (propertyNo, <sup>FK</sup>iDate, iTime, comments, <sup>FK</sup>staffNo)

# Q4)

4. The following Report table is used by a publishing house to keep track of the editing and design of books by a number of authors:

report_no	editor	dept_no	dept_name	dept_addr	author_id	auth_name	auth_addr
4216	woolf	15	design	argus1	53	mantel	cs-tor
4216	woolf	15	design	argus1	44	bolton	mathrev
4216	woolf	15	design	argus1	71	koenig	mathrev
5789	koenig	27	analysis	argus2	26	fry	folkstone
5789	koenig	27	analysis	argus2	38	umar	prise
5789	koenig	27	analysis	argus2	71	koenig	mathrev

By looking at the data, we see that functional dependencies in the Report table are the following:

report\_no  $\rightarrow$  editor, dept\_no  
dept\_no  $\rightarrow$  dept\_name, dept\_addr  
author\_id  $\rightarrow$  auth\_name, author\_addr

The candidate key for this relation is (report\_no, author\_id) since we need these two attributes to uniquely identify each record. Thus we have:

Report (report\_no, editor, dept\_no, dept\_name, dept\_addr, author\_id, auth\_name, auth\_addr)

- Is the Report table in 2NF? If not, put the table in 2NF.
- Are there any insert, update or delete anomalies with these 2NF relations?



# Q4a)

Report (report\_no, editor, dept\_no, dept\_name, dept\_addr, author\_id, auth\_name, auth\_addr)

report\_no → editor, dept\_no

dept\_no → dept\_name, dept\_addr

author\_id → auth\_name, auth\_addr

## Partial Dependencies

Author (author\_id, auth\_name, auth\_addr)

This is a new relation → Report (report\_no, dept\_no, dept\_name, dept\_addr, editor)

This is the original Report → ReportAuthor (<sup>FK</sup>report\_no, <sup>FK</sup>author\_id)

## Q4b)

- Will we still have anomalies? Yes!
- transitive dependency: dept\_no  $\rightarrow$  dept\_name, dept\_addr
- Deletion Anomaly: deletion of a record from report table may delete information about a department
- Insertion Anomaly: cannot insert a new department until we have a report for it

Author (author\_id, auth\_name, auth\_addr)

Department (dept\_no, dept\_name, dept\_addr)

Report (report\_no, <sup>FK</sup>dept\_no, editor)

ReportAuthor (<sup>FK</sup>report\_no, <sup>FK</sup>author\_id)

# Q5)

5. Consider the following relation:

Class (courseNumber, roomNumber, instructorName, studentNumber,  
workshopNumber, grade, tutor)

The following functional dependencies hold for this relation:

$\text{workshopNumber} \rightarrow \text{tutor}$

$\text{studentNumber}, \text{courseNumber} \rightarrow \text{grade}, \text{workshopNumber}$

$\text{courseNumber} \rightarrow \text{roomNumber}, \text{instructorName}$

Normalise this relation into 3NF.

# Q5)

Class (courseNumber, roomNumber, instructorName, studentNumber,  
workshopNumber, grade, tutor)

workshopNumber  $\rightarrow$  tutor

studentNumber, courseNumber  $\rightarrow$  grade, workshopNumber

courseNumber  $\rightarrow$  roomNumber, instructorName

- Already in 1NF
- 2NF: partial dependencies

Course (courseNumber, roomNumber, instructorName)

FK

Class (courseNumber, studentNumber, workshopNumber, grade, tutor)

# Q5)

Class (courseNumber, roomNumber, instructorName, studentNumber,  
workshopNumber, grade, tutor)

workshopNumber → tutor

studentNumber, courseNumber → grade, workshopNumber

courseNumber → roomNumber, instructorName

- 3NF: Transitive dependencies

Workshop (workshopNumber, tutor)

Class (<sup>FK</sup>courseNumber, studentNumber, <sup>FK</sup>workshopNumber, grade)

# Q5)

Course (courseNumber, roomNumber, instructorName)

Workshop (workshopNumber, tutor)

Class (<sup>FK</sup>courseNumber, <sup>FK</sup>studentNumber, workshopNumber, grade)