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
  - Otherwise → data inconsistency

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
  - Any 2 records with the same A values must have the same B values
- Written $A_1, A_2, \ldots, A_n \rightarrow B_1, B_2, \ldots, B_n$
- e.g.

Determinants

Emp# → Emp-name

Emp# → Salary

- 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 (<u>ssn</u>, 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 (<u>A</u>, <u>B</u>, C, D)
  - {A, B} is a key
  - Partial functional dependency: B → D

  - To resolve this:
    - R(<u>A</u>, <u>B (FK)</u>, C)
    - R2(<u>B</u>, 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→ D

  - To resolve this:
    - R(A, B, C (FK))
    - R2(C, D)

Armstrong's Axioms

A = {A1, A2, ..., An}   B = {B1, B2, ..., Bn}   C = {C1, C2, ..., Cn}

- Reflexivity: $B \subseteq A \Longrightarrow A \rightarrow B$ $\text{ssn, name} \longrightarrow \text{name}$
- Augmentation: $A \rightarrow B \Longrightarrow AC \rightarrow BC$ $\text{ssn, name, age} \longrightarrow \text{name, age}$
- Transitivity: $A \rightarrow B \text{ and } B \rightarrow C \Longrightarrow A \rightarrow C$ $\text{ssn} \longrightarrow \text{birthdate, birthdate} \longrightarrow \text{age} \Longrightarrow \text{ssn} \longrightarrow \text{age.}$

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

Repeating groups: multiple values in a single cell / a “sub-table” (like below)

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

StudentID	StudentName	CourseID	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:

   DoctorID → DocName

   DoctorID, PatientID → 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

DoctorID	DocName	PatientID	DiagnosisClass
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’s diagnosis 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.

When choosing PK to be AB,
B → D is a partial functional dependency as B is a key attribute
R1 (<u>A</u>, <u>B</u>, C)
R2 (<u>B</u>, 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 → iTime, comments, staffNo, sName
propertyNo → pAddress
staffNo → 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 → iTime, comments, staffNo, sName
propertyNo → pAddress
staffNo → sName

- 2NF:

Property (<u>propertyNo</u>, pAddress)

FK
PropertyInspection (<u>propertyNo</u>, <u>iDate</u>, iTime, comments, staffNo, sName)

Q3)

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

The FDs stated below hold for this relation:

propertyNo, iDate → iTime, comments, staffNo, sName
propertyNo → pAddress
staffNo → sName

- 3NF: Property (<u>propertyNo</u>, pAddress)

  Staff (<u>staffNo</u>, sName)

  PropertyInspection (<u>propertyNo</u> (FK), <u>iDate</u>, iTime, comments, staffNo (FK))

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 → editor, dept_no
dept_no → dept_name, dept_addr
author_id → 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 (<u>report_no</u>, editor, dept_no, dept_name, dept_addr, <u>author_id</u>, auth_name, auth_addr)

a. Is the Report table in 2NF? If not, put the table in 2NF.
b. 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, author_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 (report_no FK, author_id FK)

Q4b)

- Will we still have anomalies? Yes!
- transitive dependency: dept_no → 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)

FK
Report (report_no, dept_no, editor)

FK FK
ReportAuthor (report_no, author_id)

Q5)

5. Consider the following relation:

   Class (<u>courseNumber</u>, roomNumber, instructorName, <u>studentNumber</u>, workshopNumber, grade, tutor)

The following functional dependencies hold for this relation:

workshopNumber → tutor
studentNumber, courseNumber → grade, workshopNumber
courseNumber → roomNumber, instructorName

Normalise this relation into 3NF.

Q5)

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

workshopNumber → tutor
studentNumber, courseNumber → grade, workshopNumber
courseNumber → roomNumber, instructorName

- Already in 1NF
- 2NF: partial dependencies

Course (courseNumber, roomNumber, instructorName)

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

Q5)

Class (<u>courseNumber</u>, roomNumber, instructorName, <u>studentNumber</u>, workshopNumber, grade, tutor)

workshopNumber → tutor
studentNumber, courseNumber → grade, workshopNumber
courseNumber → roomNumber, instructorName

- 3NF: Transitive dependencies

Workshop (<u>workshopNumber</u>, tutor)

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

Q5)

Course (courseNumber, roomNumber, instructorName)

Workshop (workshopNumber, tutor)

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