CE0317-Database Management System

ASSIGNMENT-2

INSTRUCTIONS:

a. Neat presentation.
b. Include well labelled diagrams.
c. Write down differences in tabular format.
d. Mention and Highlight important text.
e. Use only blue and black ink for presentation

-:QUESTION/ANSWERS:-

Q-1). Consider the relation Treatment with the schema Treatment (doctorID, doctorName, PatientID, diagnosis) and functional dependencies : doctorID → doctorName and (doctorID, patientID) → diagnosis.
Describe different types of anomaly that can arise for this table with example records.

Ans

In a relational schema like the one described for the Treatment table :

Treatment (doctorID, doctorName, patientID, diagnosis) with the following functional dependencies :
- doctorID → doctorName
- (doctorID, patientID) → diagnosis

Several types of anomalies can arise if the table is not properly normalized. These anomalies include update anomalies, insert anomalies and delete anomalies.
Let's understand each anomalies with an example based on this schema :

i). Update Anomaly :-

- An update anomaly occurs when the same data has to be updated in multiple rows, leading to data inconsistency if not all instances are updated correctly.

- Let's understand it with an example :

Consider the following records in the Treatment Table :

doctorID	doctorName	patientID	diagnosis
101	Dr. Smith	1001	Flu
101	Dr. Smith	1002	Cold
101	Dr. Smith	1003	Headache

- Here the doctorName ("Dr Smith") is repeated for every patient. If Dr Smith changes their name ("Dr. John Smith") this change needs to be made in multiple rows. If the update is not applied consisently across all records, there should be inconsistency, where some rows have "Dr. Smith" and others have "Dr. John Smith".

ii). Insert Anomaly :-

- An insert anomaly occurs when certain data cannot be added to the table without having other data that may not yet be available or applicable.

- Let's understand it with an example :

In the treatment table, suppose a new doctor joins the hospital, and you want to add their details to the table. However, since there is no patient yet assigned to the doctor, you don't have a valid patientID or diagnosis to enter. Due to the table's design, you can't insert a row for the new doctor without also including a patient, which creates an insert anomaly.

doctorID	doctorName	patientID	diagnosis
102	Dr. Johnson	NULL	NULL

- This row may not be allowed because patientID and diagnosis fields might be required.

iii). Delete Anomaly :-

- A delete anomaly occurs when deleting a record unintentionally removes other useful information.

- Let's understand it with an example :

doctorID	doctorName	patientID	diagnosis
101	Dr. Smith	1001	Flu
101	Dr. Smith	1002	Cold
101	Dr. Smith	1003	Headache

- Suppose the last patient of Dr. Smith (patientID = 1003) is discharged, and you want to delete their treatment record. Deleting the row with patientID = 1003 would remove the association between doctorID = 101 and doctorName = Dr. Smith entirely if there are no other patients linked to Dr. Smith. This would cause a loss of information about Dr. Smith even though they are still employed and available for new patients.

iv). Redundancy (Duplication) :-

- Redundancy is when the same piece of information is stored multiple times in different rows, leading to increased storage requirements and potentional inconsistencies.

- Let's understand it with an example:

doctorID	doctorName	patientID	diagnosis
101	Dr. Smith	1001	Flu
101	Dr. Smith	1002	Cold

- The doctorName ("Dr. Smith") is duplicated multiple times, leading to redundancy. If doctorName needs to be updated, it would have to be done in every row where doctorID = 101. This increases the risk of inconsistencies and unnecessary duplication of data.

2. Normalize the below table User_Personal upto 3NF.

UserID	U_email	Fname	Lname	City	State	Zip
A12	mani@ymail.com	Manish	Jain	Bilaspur	Chatisgarh	458991
P045	pooja.g@gmail.com	Pooja	Magg	Kacch	Gujarat	832212
LA33	laule98@jj.com	Lauleen	Dhalla	Raipur	Chattisgardh	853578
CH99	cheki9j@ib.com	Chimal	Bedi	Trichy	Tamilnadu	632011
DA74	Danu58@g.com	Dany	James	Trichy	Tamilnadu	645018

Ans

- To normalize the given User_Personal table upto 3NF (Third normal form), we'll go through the following normalization steps : 1NF, 2NF and 3NF.

Step-1 : 1NF (First Normal Form) :-

- 1NF requires that all values in the table be atomic. In the given table, the data is already in atomic form, meaning each field contains a single value. Therefore, it is already in 1NF.

Step-2 : 2NF (Second Normal Form) :-

- 2NF requires that the table be in 1NF and that all non-key attributes are fully functionally dependent on the primary key. In this case, the canditate key is UserID. However, we can see that City, State, and Zip are dependent on each other rather than directly on the UserID. This violates 2NF, as these columns (City, State, Zip) depend on each other.

· Decompose the table into two tables to satisfy 2NF:

i). User Table :-

· Attributes that are dependent on UserID

UserID	U_email	Fname	Lname	City
A12	mani@ymail.com	Manish	Jain	Bilaspur
P045	pooja.g@gmail.com	Pooja	Magg	Kachh
LA33	lavle98@jj.com	Lauleen	Dhalla	Raipur
CH99	chekiqj@in.com	Chimal	Bedi	Trichy
DA74	danu58@g.com	Dany	James	Trichy

ii). Location Table :-

· Attributes that are location-specific (City, State, Zip) which are related to geographical location.

City	State	Zip
Bilaspur	Chattisgardh	458991
Kacch	Gujarat	832212
Raipur	Chattisgardh	853578
Trichy	Tamil nadu	632011
Trichy	Tamilnadu	645018

- Now, the User table has no partial dependency, and all non-key attributes depend fully on the primary key UserID. The Location table stores location-specific details and avoids redundancy.

Step-3 : 3NF (Third Normal Form) :-

- 3NF requires that the table be in 2NF, and all attributes should be functionally dependent only on the primary key. In other words, there should be no transitive dependencies.
- In this Location table, there is a dependency between City and the combination of State and Zip. Therefore, to achieve 3NF, we need to further decompose the Location table.

Decomposition for 3NF :

- We will break the Location table into two seperate tables : one for City-State and another for State-Zip.

Final 3NF Tables :

i). User :-

UserID	U_email	Fname	Lname	City
A12	mani@ymail.com	Manish	Jain	Bilaspur
P045	poojag@gmail.com	Pooja	Magg	Kachh
LA33	laule98@jj.com	Lauleen	Dhalla	Raipur
CH99	chekiaj@ih.com	Chimal	Bedi	Trichy
DA74	danu58@g.com	Dany	James	Trichy

ii). City_State :-

City	State
Bilaspur	Chattisgarh
Kacch	Gujarat
Raipur	Chattisgarh
Trichy	Tamilnadu

iii). State_Zip :-

State	Zip
Chattisgarh	458991
Gujarat	832212
Chattisgarh	853578
Tamilnadu	632011
Tamilnadu	645018

3.

i). Suppose, a relational schema $R(A, B, C, D, E)$ and set of functional dependencies : $F \{A \to BC, CD \to E, B \to D, E \to A\}$. Compute $CD+$, $E+$ (Closure of attribute set CD, attribute E respectively).

ii). Suppose, a relational schema $R(A, B, C, D, E, F)$ and set of functional dependencies : $F \{A \to BC, BC \to AD, D \to E, CF \to B\}$ Compute $BCF+$, $CD+$, $D+$.

iii). Suppose, a relational schema $R(A, B, C, D, E, F, G, H)$ and set of functional dependencies : $F \{A \to BC, E \to C, AH \to D, CD \to E, D \to AEH, DH \to BC\}$ Compute $AE+$. Is BCD H valid or not ?

<u>Ans</u>

i).

a). Compute $CD+$ (Closure of the attribute set CD)

- We are given the following functional dependencies on relation $R(A, B, C, D, E)$ $R(A, B, C, D, E)$ $R(A, B, C, D, E)$:

  i). $A \to BCA$ \ to $BCA \to BC$
  ii). $CD \to ECD$ \ to $ECD \to E$
  iii). $B \to DB$ \ to $DB \to D$
  iv). $E \to AE$ \ to $AE \to A$

- To compute the closure $CD+$, we need to iteratively apply the functional dependencies that are applicable based on the attributes we already know.

  i). Start with CDCDCD
  ii). From $CD \to ECD$ \ to $ECD \to E$, we can add EEE to the closure. Now, $CD+ = \{CD, E\} + = \backslash \{C, D, E\backslash\}$ $CD+ = \{C, D, E\}$.
  iii). From $E \to AE$ \to $AE \to A$, we can add AAA to the closure. Now, $CD+ = \{C, D, E, A\} + = \backslash C, D, E, A\}$ $CD+ = \{C, D, E, A\}$.

b). Compute $E+$ (closure of the attribute set E)

- Start with EEE.

i). From E → AE \to AE → A, we can add AAA to the closure. Now, E+ = {E, A} + = \{E, A} E+ = {E, A}.
ii). From A → BCA \ to BCA → BC, we can add BBB and CCC to the closure.
iii). From D → BD \ to DB → D, we can add DDD to the closure.

- So, E+ = {A, B, C, D, E} + = \{A, B, C, D, E \} E+ = {A, B, C, D, E}

ii).

a). Compute BCF + (closure of the attribute set BCF)

- We are given the following functional dependencies on relation R(A, B, C, D, E, F) R(A, B, C, D, E, F) R(A, B, C, D, E, F):

i). A → BCA \to BCA → BC
ii). BC → ADBC \to ADBC → AD
iii). D → ED \to ED → E
iv). CF → BCF \to BCF → B

- Start with BCF+

i). From CF → BCF, we add BBB, so no change.
ii). from BC → ADBC, we can add AAA and DDD.
iii). From A → BCA, we add BBB and CCC, so no change.
iv). From D → ED, we add EEE.

- So, BCF + = {A, B, C, D, E, F} + = \{A, B, C, D, E, F \} BCF + = {A, B, C, D, E, F}.

b). Compute CD+ (closure of the attribute set CD)

- Start with CD+

i). From BC → ADBC, we need both BBB and CCC to add AAA and DDD, but we only have CCC, so this dependency is not applicable yet.

ii). From D→ED , we can add EEE.
• So, CD+ = {C, D, E} + = \{C, D, E\} CD+ = {C, D, E}.

c). Compute D+ (closure of the attribute set D).

• Start with DDD
i). From D→ED , we can add EEE.
• So, D+ = {D, E} + = \{D, E\} D+ = {D, E}.

iii).
a). Compute AE+ (closure of the attribute set AE)

• Start with AEAEAE.
i). From A→BCA \to BCA → BC, we can add BBB and CCC.
ii). From E → CE \to CE→C , we can add CCC (already in the set), so no change.
iii). From D → AEHD \to AEHD → AEH , we need DDD , but we don't have it yet.
iv). From CD → ECD \to ECD → E , we need DDD , but we don't have it yet.
• No further functional dependencies are applicable. Therefore :

AE += {A, B, C, E} += \{A, B, C, E\} AE += {A, B, C, E}

b). Is BCDH valid?

• To check if BCDH is valid :

i). Start with BCDH
ii). From DH → BCDH \to BCDH → BC , we can add BBB and CCC , so no change.
iii). From D → AEHD \to AEHD → AEH , we can add A, E, HA, E, HA, E, H.
iv). From A→BCA \to BCA → BC, we can add BBB and CC , so no change.

• Therefore, $BCDH^+ = \{A, B, C, D, E, H\}$

• Since, $BCDH^+$ does not cover the entire set of attributes, BCDH is not a superkey for this relation and is not valid.