CLASS 03/05/2021

COURSE: ADVANCED DBMS

BOOKS: 1) NAVATHE

2) KORTH

PRACTICAL: 1) PL/SQL -- Evan Bayross

2) JSP

NORMALIZATION

DEFINITION: The PROCESS of decomposition of a universal relation into multiple relations obeying certain rules such that the different anomalies like insertion, deletion and update anomaly are avoided.

Anomalies:

- insertion anomaly
- deletion anomaly
- update anomaly

Insertion Anomaly

emp_project(empno,empname,sal,projno,pname)

primay key: empno,projno

Assumption: One employee can work in multiple project and a project can have multiple employees.

Suppose a new employee comes to the company and there is no project assigned to him.

As projno is NULL so we can not insert this record. The employee physically exist but logically we are facing problem inserting his record. This situation is called insertion anomaly.

Deletion Anomaly

emp_project(empno,empname,sal,projno,pname)

Suppose one employee is working on a single project and the project is terminated and the project record is removed from the database. The employee we mentioned will also be deleted technically from the database. But the employee is still working in the company and we cannot keep his record. This kind of situation is called deletion anomaly.

Update Anomaly

emp_project(empno,empname,sal,projno,pname,location)

Suppose we have updated the project location of a project to a different value. So all record related to that project should be updated accordingly. This updation is needed due to data redundancy. If for some reason some of the record contains previous value then there will be inconsistent data. This kind of situation is called update anomaly.

Functional dependency:

It expresses the relationship in between different sets of attributes of a relation.

If X and Y are sets of attributes of a realtion r defined by the relation schema R and for every pair of tuple t1 and t2 of r if t1[X]=t2[X] then t1[Y]=t2[Y] also the we say Y is functionally dependent on X of X determines Y and is expressed as

X→Y

FD Rules of inferences

i) Reflexivity: If X and Y are sets of attributes of a relation schema R and Y is a subset of X then

X→Y.

e.g. empno ename→ename

ii) Augmentation rule: If X, Y and Z are sets of attributes of a relation schema R and X→ Y holds then we can say XZ→YZ

ex: empno→ename holds then

empno sal→ename sal

iii) Transitivity Rule: If X, Y and Z are sets of attributes of a relation schema R ; X→ Y and Y→z holds then X→Z holds.

if

empno→desig

design→sal

then

empno→sal

Rule i,ii,iii are called Armstong rule of inference.

iv) Addition rule:

If X, Y and Z are sets of attributes of a relation schema R ; X→ Y and X→Z holds then X→YZ holds.

Proof:

Given X→Y a)

X→ Z holds b)

Augment X on both sides of a)

X→XY c)

Augment Y on both sides of b)

XY→YZ d)

Using transitivity rule from c) and d)

X→YZ (proved)

v) Decompostion rule:

If X, Y and Z are sets of attributes of a relation schema R ; X→YZ holds then X→ Y and X→Z holds.

Proof: Given X→YZ a)

From reflexivity rule YZ→Y b)

From transitivity rule

X→Y (proved)

vi) Pseudo transitivity rule:

If X, Y , Z and W are sets of attributes of a relation schema R ; If X→Y and WY→Z hold then WX→Z holds.

Proof:

Given X→Y a)

WY→Z b)

Augment W on both sides of a)

WX→WY c)

Using transitivity rule in b) and c)

WX→Z (proved)

Closure of a set of attributes

R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies

F = { {A, B}→{C}, {A}→{D, E}, {B}→{F}, {F}→{G, H}, {D}→{I, J} }

Find (AB)+ ?

(AB)+={AB}

={ABC} AB→C

={ABCDE} A→DE

={ABCDEF} B→F

={ABCDEFGH} F→GH

={ABCDEFGHIJ} D→IJ

Deriving a functional dependency from a set of FD

Let us consider F is a set of FDs. Then another functional dependency X→ Y can be derived or inferred from F, expressed as F ł X→Y if Y is a subset of X†.

Cover of FD set

A FD set F covers another FD set G if every FD of the form X→Y of G can be derived from F.

Equivalence of two FD sets

Two FD sets F and G are equivalent if “F covers G” and “G covers F”.

1) F={A→C,AC→D,E→AD,E→H}

G={A→CD , E→AH} Are F and G equivalent?

F and G are equivalent if F covers G and G covers F.

F covers G

i) F ǂ A→CD

(A)+={A}

={AC} A→C

={ACD} AC→D

CD is a subset of (A)+. so F ǂ A→CD

ii) F ǂ E→AH.

(E)+={E}

={EAD} E→AD

={EADH} E→H

={EADHC} A→C

so AH is a subset of (E)+ . So F ǂ E→AH.

so F covers G

G covers F

G={A→CD , E→AH}

i) G ǂ A→C

(A)+={A}

={ACD} FD Rule A→CD

C is a subset of (A)+. So G ǂ A→C

ii) G ǂ AC→D

(AC)+={AC}

={ACD} A→CD

D is subset of (AC)+. Hence G ǂ AC→D

iii) G ǂ E→AD

(E)+={E}

={EAH} E→AH

={EAHCD} A→CD

AD is a subset of (E)+ . Hence G ǂ E→AD

iv)) G ǂ E→H

(E)+ ={EAHCD}

H is a subset of (E)+

Hence G covers F

Hence F and G are equivalent.

Minimal Cover

Normal forms defined by primary key

- **1NF**
- **2NF**
- **3NF**
- **General form 2NF**
- **General form of 3 NF**
- **BCNF**

1NF

A relation in 1NF if every cell of it contains atomic values. So 1NF rules out possibility of repeating group of attributes or multi-valued attributes.

2NF

A relation is in 2NF if

- It is in 1NF
- Every non-prime attribute should be fully functionally dependent on the primary key.

prime attribute: The attributes of a relation which belong to primary key.

non-prime attribute:The attributes of a relation which does not belong to primary key.

emp-proj
(empno,ename,desig,charge,projno,projname,hours)

Partial and full fucntional dependency: Suppose X→Y is a FD. Also Z is a subset of X. and Z→Y also holds then Y is partially dependent on X.

If no such Z exists then Y is fully dependent on X.

empno projno → ename

empno→ename

so ename is partially dependent on (empno,projno)

3NF

A relation in 3NF

- It is in 2NF
- Every non prime attribute should not be transitively dependent of primary key or part or there is no mutual dependency in between any two non prime attribute.

empno→desig

design→charge

then we can say empno→charge. So this is not in 3NF.