Department of CSE/IT

Sessional Test-2 Solution

Course: B.Tech.

Session: 2017-18.

Subject: DBMS.

Max Marks: 50

Semester: V

Section: CS-1,2,3, IT-1,2

Sub. Code: NCS-502

Time: 2 Hour.

Faculty:-

1. Dr. Arun Yadav (CSE)
2. Mr. Ruchin Gupta (IT)
3. Ms. Vidushi (IT)

12/10/17

(Ruchin Gupta).

Reviewed by
Anupama
Anupama Sharma

SECTION - A

Q1. what do you mean by referencial integrity?

Ans: Referential integrity is a property of data which, when satisfied, requires every value of one attribute (column) of a relation (table) to exist as a value of another attribute in a different (or the same) relation.

Q.2 Distinguish between functional dependency and multi-valued dependency.

Ans: In relational database theory, a functional dependency is a constraint between two sets of attributes in a relation from a database. In other words, functional dependency is a constraint that describe the relationship b/w the attributes in a relation.

In database theory, a multi-valued dependency is a full constraint between two sets of attributes in a relation. In contrast to the functional dependency, the multi-valued dependency requires that certain tuples be present in a relation.

Q.3 what is union compatibility?

Ans: Two tables are said to be union compatibil~~ity~~ if the both the tables have same number of attributes (column) and corresponding attributes have same data type. corresponding attributes means first attribute of both relations, then second and so on.

Q.4 Explain insert and delete anomolies associated with RDBMS?

Ans: Tables that have redundant data which creates problems known as anomalies. So Data redundancy is a cause of an anomalies.

(I) Insert Anomaly: when we insert a record without having it stored on the related record.

(II) Delete Anomaly: when we delete some information and lose valuable related information at the same time.

Q.5 Differenciate between partial functional dependency and full functional dependency.

Ans: Full functional Dependency: Given a Relation R and functional Dependency $X \rightarrow Y$, Y is fully functional dependent on X and there should not be any $Z \rightarrow Y$, where Z is a proper subset of X.

Partial Dependency: if any proper subsets of the key determine any of the non-key attributes then there exist a partial dependency.

SECTION: B

(6) Consider the schema given below

EMPLOYEE (E-NAME, STREET, CITY)

WORKS-FOR (E-Name, COMPANY Name, SALARY)

COMPANY (COMPANY-Name, CITY)

MANAGES (E-NAME, MANAGER-NAME)

Write the SQL queries for the following.

(i) Find the name of employee works for infosys

(ii) Find the names and cities of residence of employees working for TCS

(iii) Find name, street and city of residence of employees working for infosys and earning more than 20,000.

(iv) Find the names of employees working in the same city where they live.

(v) Find the names of employees, who are not working for wipro.

Ans (i) SELECT E-NAME FROM works-for where company-Name = 'Infosys'.

(ii) Select E.Ename, city from works-for W, employee E where W.E-NAME = E.E-Name and company-Name = 'TCS'.

(iii) Select E.E-Name, street, city from works-for W, employee E where W.E-Name = E.E-Name and company-Name = 'Infosys' AND SALARY > 20,000.

(iv) select E.E-Name = W.E-Name AND W.COMPANY Name = C.COMPANY-Name and E.city = C.city

(v) select E-Name from employee MINUS select E-Name from works-for where company-Name = 'WIPRO'

(7) what do you mean by SQL JOIN? Explain its types with suitable Example.

Ans: An SQL Join clause combines columns from one or more tables in a relation database. it creates a set that can be saved as a table or used as it is. A Join is a means for combining columns from one or more tables by using values common to each.

TYPES OF JOIN:

(i) INNER JOIN: in this kind of a Join, we get all recording that matches the condition in both the tables, and records in both the tables that do not match are not reported.
In other words, INNER JOIN is based on the single fact that: ONLY the matching etries in BOTH the tables SHOULD be displayed.

(ii) OUTER JOIN: Outer join retrieves, Either the matched rows from one table and all rows in the other table, or all the tables all rows.

LEFT OUTER JOIN: This Join returns all the rows from the left table in conjunction with the matching rows from right table. if there is no column matched in the right table, it returns NULL values.

RIGHT OUTER JOIN: This JOIN returns all rows from right table in conjunction with the matching rows from the left table. if there are no column matching in the left table, it returns NULL values.

NATURAL JOIN: it is based on the two conditions

(I) the Join is made on all the columns with the same name for equality.

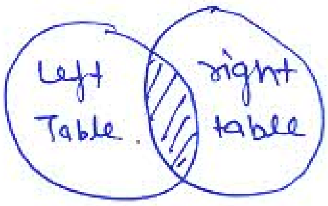
(II) Removes duplicate columns from the result.

CROSS JOIN

it is the cartesion product of the two tables involved. The result of a CROSS JOIN will not make sence in most of the situations.

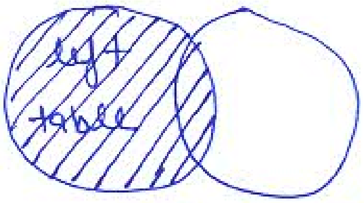
SELF JOIN: it is not different ~~from~~ form of JOIN, rather it is a JOIN of table of itself.

Example.



INNER JOIN

FULL JOIN



LEFT JOIN

RIGHT JOIN

Q.8. What are prime and non-prime attributes? For the relation R(A, B, C, D). Find prime and non-prime full and partial functional dependencies of following FDs.

$C \to D, C \to A, B \to C$

Ans: Prime attributes are the attributes of the candidate key which define uniqueness.

Non-prime attributes are ~~not~~ attributes that are not part of candidate key are called non-prime attribute.

$C \to D, C \to A, B \to C$

candidate key

$[B]^+ = \{BC\}$

$= \{C, B, D\} \quad C \to D$

$= \{A, B, C, D\} \quad C \to A$

B is candidate key of Relation R.

prime attribute = B

non-prime attribute = A, C, D

Full functional dependency:- $B \to C$

Partial functional dependency → Nil

Q.9 What do you mean by loss-less join decomposition? Determine whether the following decomposition of SP(S#, Sname, Scity, Status, P#, Pname, Price, Qty) is loss-less-join decomposition or not.

Decomposition:

CS (Scity, Status)
SUPP (S#, Sname, Scity)
PART (P#, Pname, Price)
SPN (S#, P#, Qty)

S# → Sname, Scity
Scity → Status
P# → Pname, Price
{S#, P#} → Qty

Ans: Since S# → Scity and Scity → Status, thus S# → Status

therefore S# → Sname, Scity, Status

Step 1 and Step 2: Make a matrix of size 4×8 and initialize it.

	S# 0	Sname	Scity 2	Status	P# 4	Pname 5	Price	City 6
CS 0	b_{00}	b_{01}	a_2	a_3	b_{04}	b_{05}	b_{06}	b_{07}
SUPP 1	a_0	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}	b_{17}
PART 2	b_{20}	b_{21}	b_{22}	b_{23}	a_4	a_5	a_6	b_{27}
SPN 3	a_0	b_{31}	b_{32}	b_{33}	a_4	b_{35}	b_{36}	a_7

Step. 3

Applying the FD Scity → Status, row 0 and 1 match on the value of Scity so force these two rows to match on the value of Status. Thus replace b_{13} in row by a_3.

		S# 0	Sname 1	Scity 2	Status 3	P# 4	Pname 5	Price 6	Qty 7
CS	0	b_{00}	b_{01}	a_2	a_3	b_{04}	b_{05}	b_{06}	b_{07}
SUPP	1	a_0	a_1	a_2	a_3	b_{14}	b_{15}	b_{16}	b_{17}
PART	2	b_{20}	b_{21}	b_{22}	b_{23}	a_4	a_5	a_6	b_{17}
SPN	3	a_0	b_{31}	b_{32}	b_{33}	a_4	b_{35}	b_{36}	a_7

Now, applying the FD P# → Pname, Price, row 2 and 3 match on the value of P#, so force these two rows to match on the value of Pname and Price. Thus replace b_{35} in row 3 by a_5 and replace b_{36} in row 3 by a_6.

		S# 0	Sname 1	Scity 2	Status 3	P# 4	Pname 5	Price 6	Qty 7
CS	0	b_{00}	b_{01}	a_2	a_3	b_{04}	b_{05}	b_{06}	b_{07}
SUPP	1	a_0	a_1	a_2	a_3	b_{14}	b_{15}	b_{16}	b_{17}
PART	2	b_{20}	b_{21}	b_{22}	b_{23}	a_4	a_5	a_6	b_{17}
SPN	3	a_0	b_{31}	b_{32}	b_{33}	a_4	a_5	a_6	a_7

Now applying the FD. S# → Sname, Scity, Status. row 1 and 3 match on the value of S#, so force these two rows to match on the value of Sname, Scity and Status. Thus, replace b_{31} in row 3 by a_1 replace b_{32} by a_2 and replace b_{33} in row 3 by a_3.

		S# 0	Sname 1	Scity 2	Status 3	P# 4	Pname 5	Price 6	Qty 7
CS	0	b_{00}	b_{01}	a_2	a_3	b_{04}	b_{05}	b_{06}	b_{07}
SUPP	1	a_0	a_1	a_2	a_3	b_{14}	b_{15}	b_{16}	b_{17}
PART	2	b_{20}	b_{21}	b_{22}	b_{23}	a_4	a_5	a_6	b_{17}
SPN	3	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7

Step: 4 The row 3 contains only 'a' values: therefore the above decomposition is a loss-less-join decomposition.

Determine whether a given decomposition $(R_1, R_2, \ldots R_n)$ of schema R is a loss-less-join decomposition or not.

A project R_i is said to form a Primary key foreign key relationship with another project R_j in the decomposition (where $R_i \neq R_j$) if $R_i \cap R_j \rightarrow R_i$ or $R_i \cap R_j \rightarrow R_j$.

Q.(10) Compute the closure of the following set F of functional dependencies for relation schema R = (A, B, C, D, E). $A \rightarrow BC$, $CD \rightarrow E$, $B \rightarrow D$, $E \rightarrow A$. List the candidate keys for R. Find the canonical cover F_c.

Ans: compute the closure of the following set F of functional dependencies for relation $R = (A, B, C, D, E)$

$A \to BC$, $CD \to E$, $B \to D$, $E \to A$

$A \to BC$, we can conclude $A \to B$, $A \to C$

Since $A \to B$ and $B \to D$, $A \to D$ (Decomposition transitive)

Since $A \to CD$ and $CD \to E$, $A \to E$ (union Decomposition)

$A \to A$ (reflexive)

$A \to ABCDE$

Since $E \to A$, $E \to ABCDE$

Since $CD \to E$, $CD \to ABCDE$ (transitive)

Since $B \to D$ & $BC \to CD$, $BC \to ABCDE$

Also $C \to C$, $D \to D$, $BD \to D$

Therefore any functional dependency with A, E, BC or CD on the left hand side of the arrow is in F^+. no matter which other attributes appear in the FD. Allow $*$ to represent any set of attributes in R, then F^+ is $BD \to B$, $BD \to D$, $C \to C$, $D \to D$, $BD \to BD$, $B \to D$, $B \to B$, $B \to BD$, and all FDs of the form $A* \to \alpha$, $BC* \to \alpha$, $CD* \to \alpha$, $E* \to \alpha$, where α is any subset of $\{A, B, C, D, E\}$. The candidate keys are A, BC, CD, and E. The given set of FDs F is: $A \to BC$, $CD \to E$, $B \to D$, $E \to A$

the left side of each FD in F is unique. Also none of the attribute in the left side or right side of any of the FDs is extraneous. Therefore the canonical cover F_c is equal to F.

SECTION - C

(11) R (A, B, C, D, E, F) and set of functional dependencies A→BCDEF, BC→ADEF, B→F, D→E. What is the key of R. is the relation 3NF. if yes then explain your Answer if not, then decompose it into the appropriate 3NF relations. show clearly all the steps involved.

Ans. R (A, B, C, D, E, F)

A → BCDEF, BC → ADEF, B → F, D → E

A^+ = ABCDEF

BC^+ = BCADEF

B^+ = BF

D^+ = DE

since closer of A and BC covers all the attributes of the relation R. [A^+] and [BC^+] will be the key of R.

Prime Attributes → A, B, C

Non. Prime Attributes → D, E, F

R is in 1st NF

For 2nd NF!

B → F is a partially functional dependency therefore R is not in 2nd NF.

To reduce it into 2nd NF we can decompose the relation R as:

R_1 = (B, F)

R_2 = (A, B, C, D, E)

F.D in R_1 will be $B \rightarrow F$

and f.d in R_2 will be $A \rightarrow BCDE$

$BC \rightarrow ADE$

$D \rightarrow E$

Now R_1 and R_2 are in 2^{nd} NF.

For 3^{rd} N.F condition

(I) No transitivity

(II) Non-trivial fds

(III) if there is f.d $x \rightarrow b$, then x should be superkey or b must be a prime attribute

(iv) R will be in 2^{nd} N.F

R_1 is in 3.N.F as it satisfy all the conditions of 3NF

R_2 is not in 3.N.F as in $D \rightarrow E$ is not the key and E is non-prime attribute.

Therefore R_2 can be decomposed as

$R_3(A, B, C)$ $R_4(D, E)$

$A \rightarrow BC$ $D \rightarrow E$

$BC \rightarrow A$

Now R_3 and R_4 are in 3N.F

the final result of the decomposing R and reducing into 3NF.

$R_1(B, F)$. $R_3(A, B, C)$ $R_4(D, E)$

(12) consider the following relational algebra expression to answer the following.

Employee (ename, street, city)

Works (ename, cname, salary)

Company (cname, city)

Manages (ename, mname)

(i) Find the names of employees who work for First Bank Corporation.

(ii) Find the names and cities of residence of all employees who work for First Bank Corporation.

(iii) Find the names, street, and cities of residence of all employees who work for first bank corporation and earn more than $10,000 per annum.

(iv) Find the names of all employees in this database who live in the same city as the company for which they work.

(v) Find the names of employees who live in the same city and on the same street as do their managers.

(vi) Find the names of all employees in this database who do not work for First Bank corporation.

(vii) Assume the companies may be located in several cities. Find all companies located in every city in which small bank corporation is located.

Ans(i) $\Pi_{ename}(\sigma_{cname='Firstcorp.bank'}(WORKS))$

(ii) $\Pi_{ename,city}(Employee) \bowtie_{Employee.ename=works.ename} \Pi_{ename}(\sigma_{cname='FirstBank Corporation'}(WORKS))$

(iii)

$T_1 \leftarrow \Pi_{ename}(\sigma_{ename='First Bank'}(works)) \cap \Pi_{ename}(\sigma_{salary>\$10000}(WORKS))$

$\Pi_{ename,street,city}((Employee) \bowtie_{Employee.ename=works.ename} T_1)$

(iv) $T_1 \leftarrow (\Pi_{city}(Company) \bowtie_{Works.cname=Company.cname} \Pi_{ename}(WORKS))$

$Result \leftarrow \Pi_{cname}((Employee) \bowtie_{Employee.city=T_1.city} (T_1))$

(v) $T_1 \leftarrow \Pi_{city,street}(Employee) \bowtie_{Manages.ename=Employee.ena} \Pi_{ename}(Manages)$

$Result \leftarrow \Pi_{ename}\left((Employee) \bowtie_{Employee.city = T_1.city} (T_1) \cap (Employee) \bowtie_{Employee.street = T_1.street} (T_1)\right)$

(vi) $\left(\Pi_{ename}(Employee) - \Pi_{ename}\left(\sigma_{cname = 'FirstBankCorporation'}(works)\right)\right)$

(vii) $\Pi_{cname}\left((\Pi_{city}(company)) \div \Pi_{city}\left(\Pi_{cname = 'FirstBankCor}(company)\right)\right)$