U3 Functional Dependency and Normalization

Friday, October 29, 2021 14:23

1. Explain domain constraints.

a. Domain of possible values must be associated with every attribute
b. Standard domain types, such as number, character, and date/time types defined in SQL
c. Declaring an attribute to be of a particular domain acts as a constraint on the values that it can take
d. Domain constraints are the most elementary form of integrity constraint.
e. They are tested easily by the system whenever a new data item is entered into the database
f. Several attributes may have the same domain.
g. For example, the attributes customer-name and employee-name might have the same domain: the set of all person names
h. Whether customer-name and branch-name should have the same domain?
i. At the implementation level, both customer names and branch names are character strings.
j. At the conceptual, rather than the physical level, customer-name and branch-name should have distinct domains.
k. CREATE TABLE STUDENT(PRN NUMBER,

```
                    FNAME VARCHAR2(20),
                    LNAME VARCHAR2(20),
                    CLS VARCHAR2(10),
                    CONSTRAINT CH_CLS CHECK (CLS IN
                    ('FYBTECH','SYBTECH','TYBTECH','FINALBTEC H')))
INSERT INTO STUDENT VALUES (111, 'AAA', 'BBB', 'MTECH')
INSERT INTO STUDENT VALUES (111, 'AAA', 'BBB', 'TYBTECH')
```

l. Domain constraints can be applied by specifying data type and size of attribute implicitly
m. And by using Check constraint explicitly.
n. Domain constraints are checked at the time of inserting and updating record.
o. UPDATE STUDENT SET CLS='MTECH' WHERE PRN=111

2. Explain referential integrity with example.

3. Compare trigger and assertion

4. What is functional dependency?

5. Explain different types of functional dependency with example.

6. Explain 'Closure of set of functional dependency'.

7. Write note on 'Armstrong's Axioms'

a. Rules of Inference provide a simpler technique for reasoning about functional dependencies.
b. They can be used to find logically implied functional dependencies.
c. Reflexivity rule -If α is a set of attributes and $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$ holds
d. Augmentation rule - If $\alpha \rightarrow \beta$ holds and γ is a set of attributes, then $\gamma\alpha \rightarrow \gamma\beta$ holds.
e. Transitivity rule - If $\alpha \rightarrow \beta$ holds and $\beta \rightarrow \gamma$ holds, then $\alpha \rightarrow \gamma$ holds.
f. Armstrong's axioms are sound
g. They do not generate any incorrect functional dependencies.
h. They are complete.
i. For a given set F of functional dependencies, they allow us to generate all F+.
j. Although Armstrong's axioms are complete
k. It is tiresome to use them directly for the computation of F+.
l. To simplify matters further, we list additional rules.

i. Union rule - If α → β holds and α → γ holds, then α →βγ holds.
ii. Decomposition rule - If α →βγ holds, then α → β holds and α →γ holds.
iii. Pseudo transitivity rule - If α→β holds and γβ →δ holds, then αγ →δ holds.
iv. Self-determination: α → α
v. Composition : If A → B, C → D then AC → BD
vi. If A → B and C → D, then All (C – B) → BD

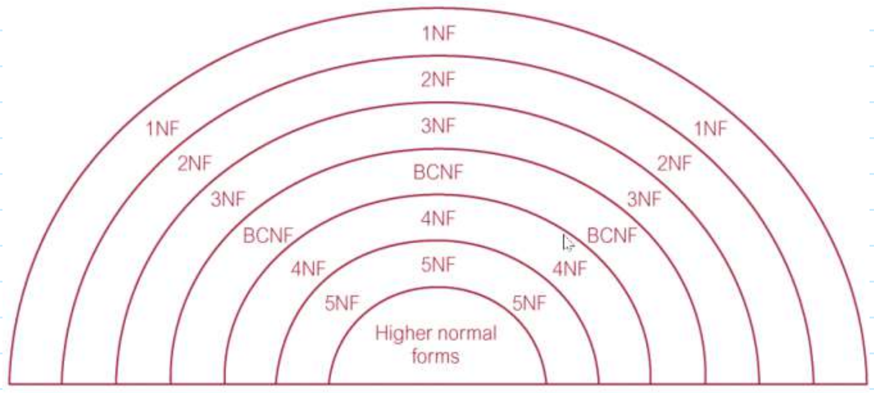
8. Explain Canonical Cover.

9. Find closure of set of functional dependency (F+) and canonical cover (Fc) from given set of functional dependencies (F) on schema (A, B, C) F={ A → BC, B → C, A → B, AB → C}

10. Explain different SQL statements used for authorization.

11. What is normalization?

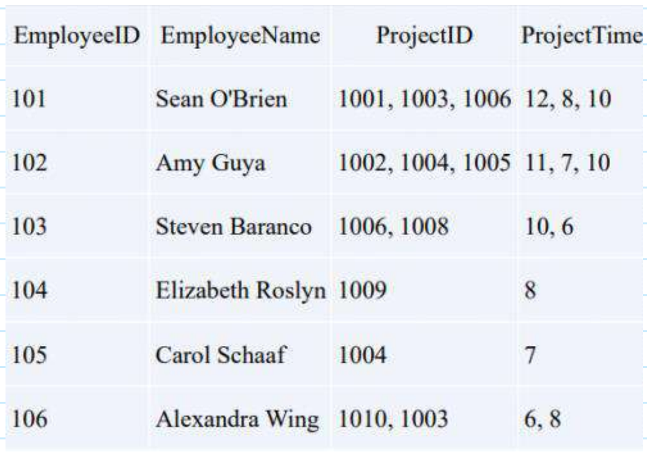
a. Database design theory includes design standards called Normal Forms.
b. The process of making data and tables match these standards is called normalizing data or data normalization.
c. By normalizing data, eliminate redundant information and
d. Organize tables to make it easier to manage the data and make future changes to the table and database structure.
e. This process removes the insertion, deletion, and modification anomalies.
f. In normalizing data, usually divide large tables into smaller, easier to maintain tables.
g. Then use the technique of adding foreign keys to enable connections between the tables.
h. Data normalization is part of the database design process
i. It is neither specific nor unique to any particular RDBMS.
j. There are first, second, third, Boyce-Codd, fourth, and fifth normal forms.
k. Each normal form represents an increasingly stringent set of rules.
l.



12. Explain 1NF with example.

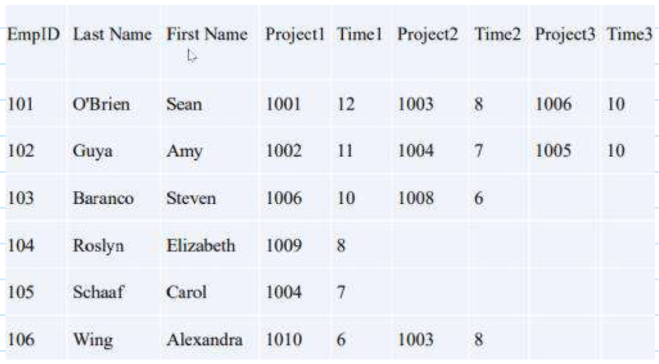
a. First Normal Form, imposes a very basic requirement on relations
b. It does not require additional information such as functional dependencies.
c. A relation in which intersection of each row and column contains one and only one value.
d. A table is in first normal form (1NF) if there are no repeating groups.
e. A repeating group is a set of logically related fields or values that occur multiple times in one record.
f. Under First Normal Form, all occurrences of a record type must contain the same number of fields.
g. Each attribute has only one value
h. All the attributes are atomic in nature
i. Atomic- the smallest level to which data may be broken down and remain meaningful.
j. E.g. NAME – FNAME, MNAME, LNAME, ADDRESS – STREET, CITY, PIN

k.



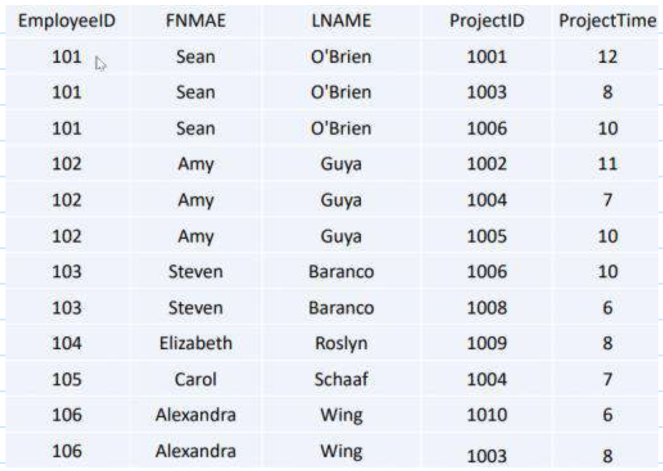
EmployeeID	EmployeeName	ProjectID	ProjectTime
101	Sean O'Brien	1001, 1003, 1006	12, 8, 10
102	Amy Guya	1002, 1004, 1005	11, 7, 10
103	Steven Baranco	1006, 1008	10, 6
104	Elizabeth Roslyn	1009	8
105	Carol Schaaf	1004	7
106	Alexandra Wing	1010, 1003	6, 8

l.



EmpID	Last Name	First Name	Project1	Time1	Project2	Time2	Project3	Time3
101	O'Brien	Sean	1001	12	1003	8	1006	10
102	Guya	Amy	1002	11	1004	7	1005	10
103	Baranco	Steven	1006	10	1008	6		
104	Roslyn	Elizabeth	1009	8				
105	Schaaf	Carol	1004	7				
106	Wing	Alexandra	1010	6	1003	8		

m.

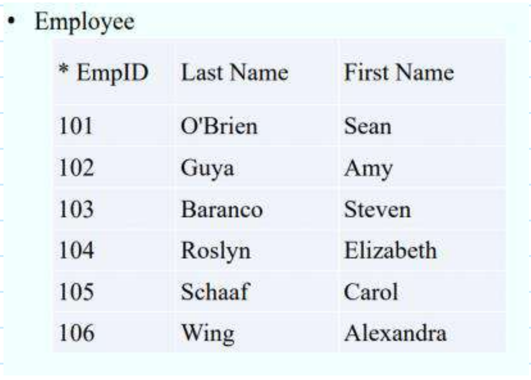


EmployeeID	FNMAE	LNAME	ProjectID	ProjectTime
101	Sean	O'Brien	1001	12
101	Sean	O'Brien	1003	8
101	Sean	O'Brien	1006	10
102	Amy	Guya	1002	11
102	Amy	Guya	1004	7
102	Amy	Guya	1005	10
103	Steven	Baranco	1006	10
103	Steven	Baranco	1008	6
104	Elizabeth	Roslyn	1009	8
105	Carol	Schaaf	1004	7
106	Alexandra	Wing	1010	6
106	Alexandra	Wing	1003	8

UNNORMALIZED FORM	1NF
PRN	PRN
SNAME	FNAME
	LNAME
SADDRESS	STREET
	CITY
	PIN
EXAMNO	EXAMNO
SUBJECT	SUBJECT
MARKS	MARKS

13. Explain 2NF with example.

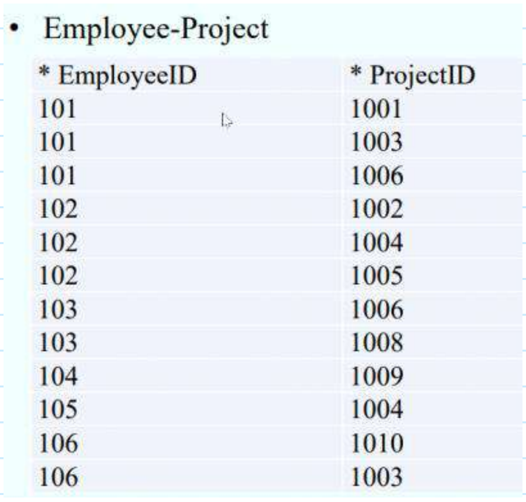
a. For any relation to be in Second Normal Form first of all, it has to be in First Normal Form and
b. Each non-key field should be dependent on entire primary key
c. No Partial Functional Dependency between non key attributes and key attributes
d. If there exists partial functional dependency between non-key attributes and key attributes,
e. it means that, there are too many fields in a single table.
f. Need to decompose it in multiple tables and
g. Link the tables using referential integrity
h.



- Employee

* EmpID	Last Name	First Name
101	O'Brien	Sean
102	Guya	Amy
103	Baranco	Steven
104	Roslyn	Elizabeth
105	Schaaf	Carol
106	Wing	Alexandra

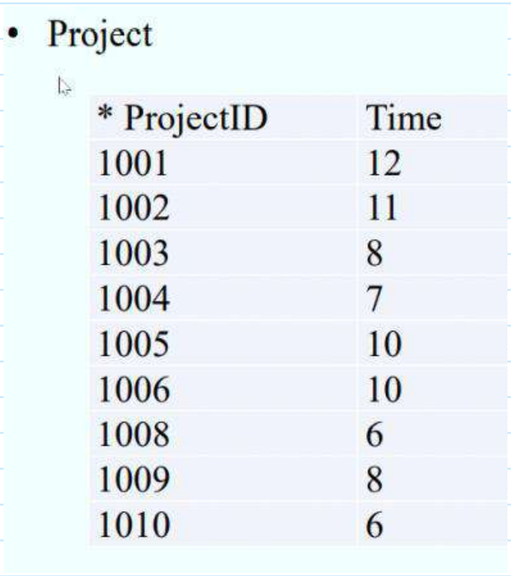
i.



- Employee-Project

* EmployeeID	* ProjectID
101	1001
101	1003
101	1006
102	1002
102	1004
102	1005
103	1006
103	1008
104	1009
105	1004
106	1010
106	1003

j.



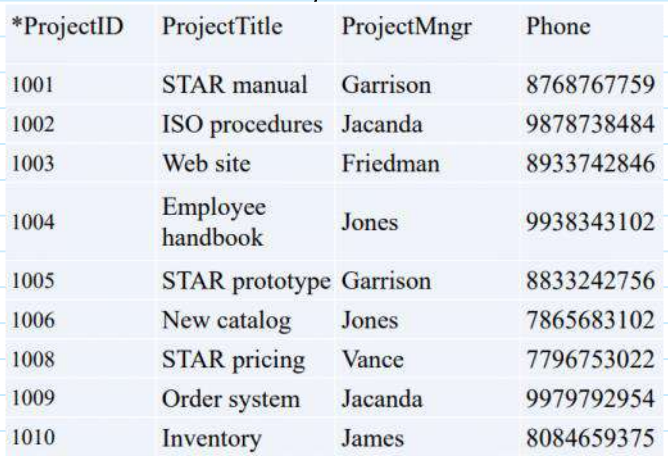
- Project

* ProjectID	Time
1001	12
1002	11
1003	8
1004	7
1005	10
1006	10
1008	6
1009	8
1010	6

1NF	2NF

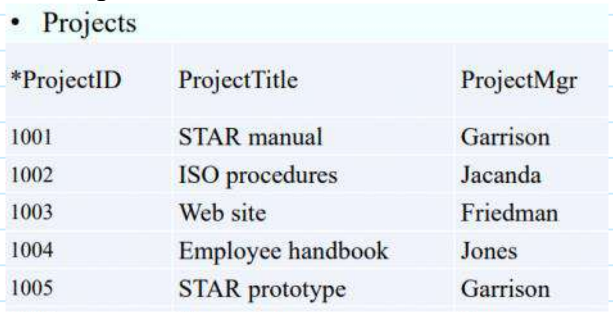
PRN, FNAME, LNAME, STREET, CITY, PIN, EXAMNO, SUBJECT, MARKS	PRN, FNAME, LNAME, STREET, CITY, PIN
	PRN, EXAMNO, SUBJECT, MARKS

14. Explain 3NF with example.

a. For any relation to be in Third Normal Form – First of all, it has to be in Second Normal Form and
b. Non-key attributes should not be transitively dependent on the primary key.
c. E.g. A is candidate key, B and C are not candidate keys
    i. A → B B → C A → C
d. A Transitive Functional Dependency is a type of functional dependency in which,
e. the value in a non-key field is determined by the value in another non-key field and that field is not a candidate key
f. 

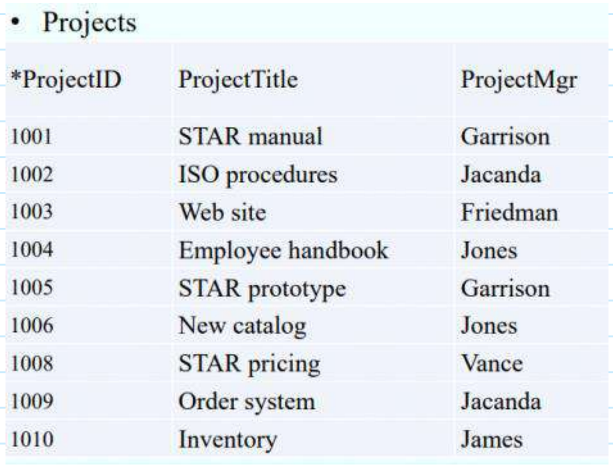
*ProjectID	ProjectTitle	ProjectMngr	Phone
1001	STAR manual	Garrison	8768767759
1002	ISO procedures	Jacanda	9878738484
1003	Web site	Friedman	8933742846
1004	Employee handbook	Jones	9938343102
1005	STAR prototype	Garrison	8833242756
1006	New catalog	Jones	7865683102
1008	STAR pricing	Vance	7796753022
1009	Order system	Jacanda	9979792954
1010	Inventory	James	8084659375

    i. The phone number is repeated each time a manager name is repeated.
    ii. This is because the phone number is only a second cousin to the project number.
    iii. It's dependent on the manager,
    iv. which is dependent on the project number
    v. (a transitive dependency)
    vi. The ProjectMngr field is not a candidate key because
    vii. the same person manages more than one project.
    viii. Again, the solution is to remove the field with repeating data to a separate table.

g. 

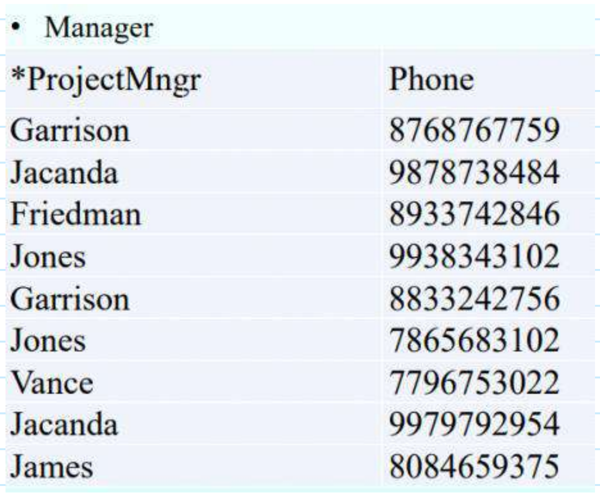
• Projects

*ProjectID	ProjectTitle	ProjectMgr
1001	STAR manual	Garrison
1002	ISO procedures	Jacanda
1003	Web site	Friedman
1004	Employee handbook	Jones
1005	STAR prototype	Garrison

g. 

- Projects

*ProjectID	ProjectTitle	ProjectMgr
1001	STAR manual	Garrison
1002	ISO procedures	Jacanda
1003	Web site	Friedman
1004	Employee handbook	Jones
1005	STAR prototype	Garrison
1006	New catalog	Jones
1008	STAR pricing	Vance
1009	Order system	Jacanda
1010	Inventory	James

h. 

- Manager

*ProjectMngr	Phone
Garrison	8768767759
Jacanda	9878738484
Friedman	8933742846
Jones	9938343102
Garrison	8833242756
Jones	7865683102
Vance	7796753022
Jacanda	9979792954
James	8084659375

2NF	3NF
PRN, FNAME, LNAME, STREET, CITY, PIN	PRN, FNAME, LNAME, STREET, CITY
PRN, EXAMNO, SUBJECT, MARKS	CITY, PIN
	PRN, EXAMNO, SUBJECT, MARKS

i. s

15. Explain BCNF with example.
16. Explain 4NF with example
17. Explain 5NF with example
18. Compare Normalization with Denormalization.

19. Explain Fourth Normal Form (4NF). Convert the following relationship into 4NF

a.

Movie	Movie	Producer
Once Upon a Time	Julie Garland	Alfred Brown
Once Upon a Time	Mickey Rooney	Alfred Brown
Once Upon a Time	Julie Garland	Muriel
Once Upon a Time	Mickey Rooney	Muriel
Moonlight	Humphrey Bogart	Alfred Brown
Moonlight	Julie Garland	Alfred Brown

20. Explain Fifth Normal Form (5NF). Convert the following relationship into 5NF.

a.

Buyer	Product	Company
Chris	jeans	Levi
Chris	jeans	Wrangler
Chris	shirts	Levi
Lori	jeans	Levi

21. Apply different applicable normal forms on the schema given in unnormalized form. Grade_report (StudNo, StudName, Branch, Mentor, CourseNo, Ctitle, InstructName, InstructLocation, Grade)
22. Apply different applicable normal forms on the schema given in unnormalized form. video(title,director,serial) customer(name,addr,memberno) hire(memberno,serial,date)
23. Apply different applicable normal forms on the schema given in unnormalized form. Employee (Employee_ID, Employee_Name, City, Department_No, Department_Name, Salary_Slip_no, Salary))

Q2. Explain Referential Integrity with Example

Ans :

Referential Integrity

- Value that appear in one relation for a given set of attributes
- also appears for a certain set of attributesin another relation.
- This condition is called referential integrity.
- Foreign keys can be specified as part of the SQL create table statement by using the foreign key Clause

create table account

(...

foreign key (branch-name) references branch(name) on delete cascade

on update cascade,

…)

- create table suppliers1(
- idnumber,name varchar2(50),CONSTRAINTFK_SP1 foreign key(id) references consumers(p_id) on delete cascade)

on delete cascade

- if a delete of a tuple in branch results in this referential-integrity constraint being violated, the system does not reject the delete.
- Instead, the delete "cascades" to the account relation, deleting the tuple that refers to the branch that was deleted
- Similarly, the system does not reject an update to field referenced by the constraint
- instead, the system updates the field branch-name in referencing tuples in account to new value
- actions other than cascade, if the constraint is violated

- The referencing field (branch-name) can be set to null by using set null in place of cascade
- create table suppliers2 (id number,name varchar2(50), CONSTRAINT FK_SP2 foreign key(id) references consumers(p_id) on delete set null)

Q3. Compare trigger and assertion

Ans :

Assertions

• An assertion is a predicate expressing a condition that we wish the database always to satisfy.
• Domain constraints and referential-integrity constraints are special forms of assertions.
• The sum of all loan amounts for each branch must be less than the sum of all account balances at the branch.
• Every saving account should have minimum balance of 1000.

• When an assertion is created, the system tests it for validity.
• If the assertion is valid, then any future modification to the database is allowed only if it does not cause that assertion to be violated.
• This testing may introduce a significant amount of overhead if complex assertions have been made.
• Hence, assertions should be used with great care.
• The high overhead of testing and maintaining assertions has led some system developers to omit support for general assertions,
• or to provide specialized forms of assertions that are easier to test

Triggers

• A trigger is a statement that the system executes automatically as a side effect of a modification to the database.
• Things necessary to design a trigger mechanism:
1] Specify when a trigger is to be executed – An event that causes the trigger to be checked and a condition that must be satisfied for trigger execution to proceed.
2] Specify the actions to be taken when the trigger executes

• The model of triggers is referred to as the event condition-action model for triggers.

• The database stores triggers just as if they were regular data, so that they are persistent and are accessible to all database operations.
• Once enter a trigger into the database, the database system takes the responsibility of executing it whenever the specified event occurs and the corresponding condition is satisfied
• Triggers are useful mechanisms for alerting humans or for starting certain tasks automatically when certain conditions are met.

• Example

• Instead of allowing negative account balances,
• the bank deals with overdrafts by setting the account balance to zero
• and creating a loan in the amount of the overdraft.
• The bank gives this loan a loan number identical to the account number of the overdrawn account.
• For this example, the condition for executing the trigger is an update to the account relation that results in a negative balance value.

Q4. WHAT IS FUNCTIONAL DEPENDENCY?

ANS :

Functional Dependency

• Functional dependencies play a key role in differentiating good database designs from bad database designs.

• A functional dependency is a type of constraint that is a generalization of the notion of key • Functional dependencies are the constraints on the set of legal relations.

• A functional dependency occurs when a set of one or more attributes in a relation uniquely determines other attributes.

• This can be written as A →B

• A functionally determines B

• B is functionally dependent upon A

• Let R be the relation, and

• let X and Y be the arbitrary subset of the set of attributes of R.

• Then Y is functionally dependent on X.

• X →Y

• X functionally determines Y

• If and only if each X value in R has associated with it precisely one Y value in R

• In other words, whenever two tuples of R agree on their X value, they also agree on their Y value.

Use of Functional Dependency

• To group the attributes together in a relation.

– The list of attributes satisfying common set of functional dependencies can be grouped together

• To select key attributes of a relation.

– Set of one or more attributes which functionally determines all other attributes in a relation can be used as a key.

Q5. Explain different types of functional dependency with example

Ans :

Types of Functional Dependencies

• Full Functional Dependency

• Partial Functional Dependency

• Transitive Functional Dependency

Full Functional Dependency

- $X \subseteq R$ and $Y \subseteq R$
- $X \rightarrow Y$
- All the attributes of X are required to determine attributes of Y.
- Subset of X is not capable of determining attributes of Y.
- E.g.
- STUDENT (PRN, SNAME, SUBJECT, MARKS)
- PRN, SUBJECT → MARKS
- Both PRN and SUBJECT are needed to determine MARKS.
- Only PRN or SUBJECT can’t determine MARKS.
- It is Full Functional Dependency

Partial Functional Dependency

- $X \subseteq R$ and $Y \subseteq R$
- $X \rightarrow Y$
- All the attributes of X are not required to determine attributes of Y.
- Subset of X is capable of determining attributes of Y.

- E.g.
- STUDENT (PRN, SNAME, SUBJECT, MARKS)
- PRN → SNAME
- PRN, SUBJECT → SNAME Partial
- PRN, SUBJECT → MARKS Full
- Only PRN is sufficient to determine SNAME.
- SUBJECT is not necessary to determine SNAME.
- It is Partial Functional Dependency

Transitive Functional Dependency

- $X \subseteq R$, $Y \subseteq R$, $Z \subseteq R$
- $X \rightarrow Y$ and $Y \rightarrow Z$
- It results into $X \rightarrow Z$
- Example
- STUDENT(PRN, SNAME, SCITY, PIN)
- PRN → SCITY
- SCITY → PIN
- PRN → PIN

Q6. Explain 'Closure of set of functional dependency'

Ans :

Closure of a Set of Functional Dependencies

• The set of all Functional Dependencies (FDs) that are implied by a given set S (or F) of Functional Dependencies is called the closure of S

• It is denoted by S+ (or F+)

• It is not sufficient to consider the given set of functional dependencies.

• It is needed to consider all functional dependencies that hold.

• Given a relation schema R = (A, B, C, G, H, I) and the set of functional dependencies

• $A \rightarrow B$

• $A \rightarrow C$

• $CG \rightarrow H$

- $CG \rightarrow I$
- $B \rightarrow H$
- The functional dependency $A \rightarrow H$ is logically implied.

- Suppose, t1 and t2 are tuples such that
- t1[A] = t2[A]
- Since we are given that $A \rightarrow B$
- It follows from the definition of functional dependency that
- t1[B] = t2[B]
- Then, since we are given that $B \rightarrow H$,
- it follows from the definition of functional dependency that
- t1[H] = t2[H]
- Given S (or F), we can compute S+ (or F+) directly from the formal definition of functional dependency.
- If F is large, this process would be lengthy and difficult.
- Armstrong’s Axioms, or Rules of Inference provide a simpler technique for reasoning about functional dependencies.

• They can be used to find logically implied functional dependencies.

Closure of Attribute Sets

• To test whether a set α is a superkey,

• we must devise an algorithm for computing the set of attributes functionally determined by α.

• One way of doing this is to compute F+

• Take all functional dependencies with α as the left hand side, and take the union of the right-hand sides of all such dependencies.

• However, doing so can be expensive, since F+ can be large

• Compute (AG)+ with the functional dependencies {A → B, A → C, CG → H, CG → I, B → H}.

• We start with result = AG.

• A → B causes us to include B in result.

• we observe that A → B is in F, A ⊆ result (which is AG), so result := result ∪ B.

• A→ C causes result to become ABCG.

• CG→H causes result to become ABCGH.

• CG→I causes result to become ABCGHI.

• So, CG is a Superkey

Q8. Explain Canonical Cover.

Ans :

Canonical Cover

• Whenever user performs Insert or Update on a relation, It is necessary ensure that insert/update does not violate any functional dependencies.

• All the functional dependencies in F are satisfied in the new database state.

• The system must roll back the update if it violates any functional dependencies in the set F.

• Closure of set of Function Dependencies add extra FDs in the set.

• It is cumbersome to check and ensure all the functional dependencies hold at every insert and update operation execution.

• Need an alternative to achieve the same task.

• Need to find minimal set of functional dependencies to reduce overload.

• That too without compromising or loosing any valid functional dependency. A canonical cover Fc for F is a set of dependencies such that

• F logically implies all the functional dependencies in Fc,

• and Fc logically implies all the functional dependencies in F.

• Furthermore, Fc must have the following properties:

– No functional dependency in Fc contains an extraneous attribute.

– All the determinants are unique (Each left side of a functional dependency in Fc is unique).

Q10. Explain different SQL statements used for authorization.

Ans :

Authorization

• Read authorization – allows reading, but not modification, of data.

• Insert authorization – allows insertion of new data, but not modification of existing data.

• Update authorization – allows modification, but not deletion, of data.

• Delete authorization – allows deletion of data.

Authorization in SQL

- Granting Privileges
- grant on to
- grant select on account to U1, U2, U3
- grant update (amount) on loan to U1, U2, U3
- grant references (branch-name) on branch to U1

- **Roles**
    - Roles can be created as follows
    - create role teller
    - Roles can then be granted privileges just as the users can:
    - grant select on account to teller
    - grant teller to john
    - create role manager
    - grant teller to manager
    - grant manager to mary
    - The Privilege to Grant Privileges
    - grant select on branch to U1 with grant option
    - revoke on from [restrict | cascade]

• restrict - system returns an error if there are any cascading revokes

• cascade - cascaded revoke of privileges

Q15. Explain BCNF with example.

Ans :

Boyce-Codd Normal Form (BCNF)

• For any relation to be in Boyce-Codd Normal Form

• First of all, it has to be in Third Normal Form and

• All determinants must be candidate keys.

• Determinant - determines the value in another field

• Candidate keys- qualify for designation as primary key.

• This normal form applies to situations of overlapping candidate keys

• Boyce-Codd normal form (BCNF) can be thought of as a "new" third normal form.

- It was introduced to cover situations that the "old" third normal form did not address.
- More than one candidate keys are present.
- Two or more attributes form candidate keys.
- Overlapping Candidate Key – One of the attribute is common in two candidate keys.
- A superkey is a set of one or more attributes that, taken collectively, allow us to identify uniquely an entity in the entity set.
- superkeys for which no proper subset is a superkey.
- Such minimal superkeys are called candidate keys.
- several distinct sets of attributes could serve as a candidate key.
- primary key denote a candidate key that is chosen by the database designer as the principal means of identifying entities within an entity set.

• Comparison 3NF and BCNF

- Every relation in BCNF is also in 3NF.
- However, relation in 3NF may not be in BCNF.

• For a functional dependency A →B,

• BCNF insists that for this dependency to remain in a relation, A must be a candidate key.

• It is not necessary in 3NF

• BCNF is more stringent than 3NF

Q16. EXPLAIN 4NF WITH EXAMPLE

ANS :

Fourth Normal Form (4NF)

• For r any relation to be in Fourth Normal Form

• First of all, it has to be in Boyce-Codd Normal Form and

• there should not be multi-valued functional dependencies.

• A multi-valued functional dependency occurs when,

• for each value in field A

• there is a set of values for field B

• and a set of values for field C

• but fields B and C are not related.

• Although BCNF removes anomalies due to functional dependencies,

• another type of dependency called a multi-valued dependency (MVD) can also cause data redundancy.

• Possible existence of MVDs in a relation is due to 1NF and can result in data redundancy.

• Look for repeated or null values in non-key fields.

• A multi-valued dependency occurs when the table contains fields that are not logically related

• 4NF

•PRN,CERTIFICATE_COURSE, WORKSHOP

• PRN → CERTIFICATE_COURSE

• PRN → WORKSHOP

• PRN, CERTIFICATE_COURSE

• PRN, WORKSHOP

Q17. EXPLAIN 5NF WITH EXAMPLE

ANS :

Fifth Normal Form (5NF)

• For any relation to be in Fifth Normal Form

• First of all, it has to be in Fourth Normal Form and

• there should not be cyclic functional dependencies. (Join Dependency)

• It will not have lossless decomposition into smaller tables.

• It contains a multi-field primary key consisting of three or more fields.

• A cyclic functional dependency can occur only when you have a multi-field primary key consisting of three or more fields.

• For example, let's say your primary key consists of fields A, B, and C.

• A cyclic dependency would arise if the values in those fields were related in pairs of A and B, B and C, and A and C

• Fifth normal form is also called projection-join normal form.

• A projection is a new table holding a subset of fields from an original table.

• When properly formed projections are joined, they must result in the same set of data that was contained in the original table.

• 5NF

• EmpName, EmpSkills, EmpJob

• EmpName, EmpSkills

• EmpSkills, EmpJob

• EmpName, EmpJob

Q18. Compare Normalization with Denormalization.

Ans :

Normalization

• Database design theory includes design standards called Normal Forms.

• The process of making data and tables match these standards is called normalizing data or data normalization.

• By normalizing data, eliminate redundant information and

• Organize tables to make it easier to manage the data and make future changes to the table and database structure.

• This process removes the insertion, deletion, and modification anomalies.

• In normalizing data, usually divide large tables into smaller, easier to maintain tables.

• Then use the technique of adding foreign keys to enable connections between the tables.

• Data normalization is part of the database design process

• It is neither specific nor unique to any particular RDBMS

• There are first, second, third, Boyce-Codd, fourth, and fifth normal forms.

• Each normal form represents an increasingly stringent set of rules.

Denormalization

• Denormalization is a strategy used on a previously normalized database to increase performance.

- Try to improve the read performance of a database at the expense of losing some write performance
- by adding redundant copies of data or by grouping data.
- To improve the performance or scalability in DBMS
- Denormalization is one of the database optimization technique in which we add redundant data to one or more tables.
- This can help us avoid costly joins in a relational database.
- In a traditional normalized database,
- we store data in separate logical tables and attempt to minimize redundant data.
- We may strive to have only one copy of each piece of data in database.