Module - II

Functional Dependencies and Normalization for Relational Database

Q1. Explain what constitutes a bad database design? Explain informal guidelines for designing a good relation schema.

Consider the following schema:
SUPPLIER(SNAME, SADDRESS, ITEM, PRICE)
There are several problems with this schema.

1. **Redundancy:** The address of the supplier is repeated once for each item supplied. The reason why is that the dependency between SNAME and (ITEM, PRICE) is multivalued.

2. **Update Anomalies:** As a consequence of the redundancy, we could update the address for a supplier in one tuple, while leaving it fixed in another. This will cause a possible inconsistency in the database.

3. **Insertion Anomalies:** The supplier address cannot be recorded if that supplier does not currently supply at least one item. NULL values can be used for ITEM and PRICE to that supplier, but SNAME and ITEM being the key of the relation, it will be impossible to take up tuple with NULL values in the key.

4. **Deletion Anomalies:** The inverse to problem 3 is that, deletion of all the items supplied by the supplier in turn lose track of their address.

Therefore, the following are four guidelines for the design of good relation schema

Guideline 1: Design a relation schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity types and relationship types into a single relation.

Guideline 2: Design the base relation schema so that no insertion, deletion or updation anomalies are present in the relations.

Guideline 3: Avoid placing attributes in a base relation whose values may frequently be NULL. If NULLs are unavoidable, make sure that they apply in exceptional cases only and do not apply to a majority of tuples in the relation.

Guideline 4: Design relation schema so that they can be joined with equality conditions on attributes that are (primary key, foreign key) pairs in a way that guarantees that no spurious tuples are generated.

Q2. Define functional dependency. Explain the properties of functional dependency.

A functional dependency, denoted by $X \longrightarrow Y$, between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuples that can form a relation state 'r' of R. The constraint is that, for any two tuples t_1 and t_2 in 'r' that have $t_1[X] = t_2[X]$, they must also have $t_1[Y] = t_2[Y]$.

That is, X functionally determines Y in a relation schema R if and only if whenever two tuples of $r(R)$ agree on their X-value, they must necessarily agree on their Y-value.

Properties

- If a constraint on R states that there cannot be more than one tuple with a given X-value in any relation instance $r(R)$ - that is, X is a candidate key of R - this implies that $X \longrightarrow Y$ for any subset of attributes Y of R.
- if $X \longrightarrow Y$ in R, this does not say whether or not $Y \longrightarrow X$
- A functional dependency $X \longrightarrow Y$ is said to be trivial if $Y \subseteq X$

Q3. Explain different Armstrong's inference rules

The set of rules that can be used to infer new functional dependencies from the given set of functional dependencies (F) are often called Armstrong's Axioms. These are:

1. **Reflexively:** If $Y \subseteq X \subseteq U$, where U is the universal set of attributes, then $X \longrightarrow Y$ is logically implied by F.
2. **Augmentation:** If $X \longrightarrow Y$ holds, and $Z \subseteq U$, then $XZ \longrightarrow YZ$
3. **Transitivity:** If $X \longrightarrow Y$ and $Y \longrightarrow Z$ holds, then $X \longrightarrow Z$ holds
4. **The Union Rule:** $X \longrightarrow Y, X \longrightarrow Z \models X \longrightarrow YZ$
5. **The Pseudotransitivity Rule:** $X \longrightarrow Y, WY \longrightarrow Z \models XW \longrightarrow Z$
6. **The Decomposition Rule:** If $X \longrightarrow Y$ holds and $Z \subseteq Y$, then $X \longrightarrow Z$ holds.

Q4. Problem on finding minimal cover

Find the minimal cover of the following set of functional dependencies.
$F \longrightarrow \{AB \longrightarrow CD, A \longrightarrow E, E \longrightarrow C\}$

Q5. What do you mean by normalization of relations.

Normalization of data can be considered as a process of analyzing the given relation schema based on their functional dependencies and primary keys to achieve the desirable properties of (1) minimizing redundancy and (2) minimizing the insertion, deletion and update anomalies. Unsatisfactory relation schema that do not meet certain conditions - the normal form test - are decomposed into smaller relation schema that meet the rests and hence possess the desirable properties. Thus the normalization procedure provides database designers with the following:

- A formal framework for analyzing relation schema based on their keys and on the functional dependencies among their attributes.
- A series of normal form tests that can be carried out on individual relation schema so that the relational database can be normalized to any desired degree.

The normal form of a relation refers to the highest normal form conditions that it meet and hence indicate the degree to which it has been normalized.

Q6. Explain 1NF, 2NF, 3NF and BCNF with suitable examples.

First Normal Form

The first normal form (1NF) states that the domain of an attribute must include only atomic (simple, indivisible) values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. Hence, 1NF disallows having a set of values, a tuple of values or a combination of both as an attribute value for a single tuple.

For example consider the relation

DEPARTMENT(Dname, $\underline{\text{Dnumber}}$, Dmgr_ssn, Dlocation)

As the attribute 'Dlocation' can have set of values, this is not an atomic value and it violates 1NF condition.

Remedy

Form a new relation for each multivalued attribute or nested relation.

Solution

Decompose the relation DEPARTMENT into DEPARTMENT AND DEPTLOC as
DEPARTMENT(Dname, $\underline{\text{Dnumber}}$, Dmgr_ssn)
DEPTLOC($\underline{\text{Dnumber, Dlocation}}$)

Second Normal Form

Second normal form (2NF) is based on the concept of full functional dependency. A functional dependency $X \longrightarrow Y$ is a full full functional dependency if removal of any attribute A from X means that the dependency does not hold any more; that is for any attribute $A \in X$ such that $(X - \{A\})$ does not functionally determine Y.

Definition

A relation schema R is in 2NF if every non prime attribute A in R is fully functional dependent on the primary key of R.

General Definition

A relation schema R is in second normal form (2NF) if every non-prime attribute A in R is not partially dependent on any key of R.

Test

For relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key.

For example, consider a relation schema
EMP_PROJ(ssn, pnumber, hours, ename, pname, plocaton)
and functional dependencies
FD1: $\{ssn, pnumber\} \longrightarrow hours$
FD2: $ssn \longrightarrow ename$
FD3: $pnumber \longrightarrow \{pname, plocation\}$

The EMP_PROJ relation is in 1NF but not in 2NF. The non prime attribute 'ename' violates 2NF because of FD2, as do the non prime attribute 'pname' and 'plocation' because of FD3.

Remedy

Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attribute that are fully functionally dependent on it.

Solution

To make the above relation to tbe 2NF normalized so that it is fully functional dependent on the primary key, decompose the EMP_PROJ into three relation shcema EP1, EP2, EP3 as:

EP1(ssn, pnumber, hours)
EP2(ssn, ename)
EP3(pnumber, pname, plocation)

Third Normal Form

The third normal form (3NF) is based on the concept of transitive dependency. A functional dependency $X \longrightarrow Y$ in a relation schema R is a transitive dependency if there is a set of attribute Z that is neither a candidate key nor a subset of nay key of R and both $X \longrightarrow Z$ and $Z \longrightarrow Y$ hold.

Definition

A relation schema R is in 3NF if it satisfies 2NF and no non prime attribute of R is transitively dependent on primary key.

General Definition

A relation schema R is in third normal form (3NF) if, whenever a non-trivial functional dependency $X \longrightarrow A$ hold in R either (a) X is a superkey of R or (b) A is a prime attribute of R.

Test

Relation should not have a non-key attribute functionally determined by another non-key attribute (or by a set of non-key attributes). That is, there should be no transitive dependency of a non-key attribute on the primary key.

For example consider the relation schema
EMP_DEPT(ename, <u>ssn</u>, bdate, address, dumber, dname, dmgr_ssn)
and functional dependencies
FD1: $ssn \longrightarrow \{ename, bdate, address, dnumber\}$
FD2: $dnumber \longrightarrow \{dname, dmgr\_ssn\}$

The relation schema EMP_DEPT is in 1NF and 2NF as no values is repeating with respect to key and no partial dependencies on a key exist. However, EMP_DEPT is not in 3NF because the dependency $ssn \longrightarrow dmgr\_ssn$ is transitive through 'dnumber' in EMP_DEPT. That is, both the dependencies $ssn \longrightarrow dnumber$ and $dnumber \longrightarrow dmgr\_ssn$ hold and 'dnumber' is neither a key itself nor a subset of the key of EMP_DEPT.

Remedy

Decompose and set up a relation that includes the non-key attribute(s) that functionally determine(s) other non-key attribute(s).

Solution

To make the above relation to be 3NF normalized, decompose the EMP_DEPT relation into two 3NF relation schemas ED1 and ED2 as follows:

ED1(ename, <u>ssn</u>, bdate, address, dnumber)
ED2(<u>dnumber</u>, dname, dmgr_ssn)

Boye-Codd Normal Form (BCNF)

Definition

A relation schema R is in BCNF if whenever a non-trivial functional dependency $X \longrightarrow A$ holds in R, then X is a superkey of R.
Consider an example relation schema

TEACH(student, course, instructor)
and the following functional dependencies
FD1: $\{student, course\} \longrightarrow instructor$
FD2: $instructor \longrightarrow course$

{student, course} is a candidate key for this relation. Hence this relation is in 3NF but not in BCNF. Decomposition of this relation schema into one of the three different possible pairs can make this relation into BCNF normalized:

1. {<u>student, instructor</u>} and {<u>student, course</u>}
2. {course, <u>instructor</u>} and {<u>course, student</u>}

3. {<u>instructor</u>, course} and {<u>instructor, student</u>}

Q7. Explain multivalued dependencies and fourth normal form.

Multivalued dependencies are a consequence of first normal form (1NF) which disallows an attribute in a tuple to have a set of values. If we have two or more multivalued independent attributes in the same relation schema, this get into a problem of having to repeat every value of one attribute with every value of the other attribute to keep the relation state consistent and to maintain the Independence among the attributes involved. This constraint is specified by a multivalued dependency.

For example, consider the relation EMP below. A tuple in this EMP relation represent the fat that an employee whose name is 'ename' works on the project whose name is 'pname' and has a dependent whose name is 'dname'. An employee may work on several projects and may have severa dependents, and the employee's projects and dependents are independent of one another. To keep the relation state consistent, a separate tuple need to be represent for every combination of an employee's dependent and an employee's project.

EMP

ename	pname	dname
smith	X	John
smith	Y	Anna
smith	X	Anna
smith	Y	John

Formal definition of multivalued dependency

A multivalued dependency $X \twoheadrightarrow Y$ specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation state 'r' of R. If two tuples t_1 and t_2 exists in 'r' such that $t_1[X] = t_2[X]$, then two tuples t_3 and t_4 should also exist in 'r' with the following properties

$t_3[X] = t_4[X] = t_1[X] = t_2[X]$
$t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$
$t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$
where $Z = R - (X \cup Y)$

Fourth Normal Form

Definition

A relation schema R is in 4NF with respect to a set of dependencies F (that includes functional dependencies and multivalued dependencies) if, for every non-trivial multivalued dependency $X \twoheadrightarrow Y$ in F^+, X is a superkey of R.