Types of Anomalies

Different types of database anomalies are as follows:

1. Insertion Anomaly

The insertion anomaly occurs when a new record is inserted in the relation. In this anomaly, the user cannot insert a fact about an entity until he has an additional fact about another entity.

2. Deletion Anomaly

The deletion anomaly occurs when a record is deleted from the relation. In this anomaly, the deletion of facts about an entity automatically deletes the fact of another entity.

3. Modification Anomaly

The modification anomaly occurs when the record is updated in the relation. In this anomaly, the modification in the value of specific attribute requires modification in all records in which that value occurs.

5.14 Normalization

The process of producing a simpler and more reliable database structure is called **normalization**. It is used to create a suitable set of relations for storing data. This process works through different stages known as **normal forms**. These stages are 1NF, 2NF, 3NF and so on. Each normal form has certain requirements or condition. These conditions have to fulfilled to bring the database in that particular normal form. If a relation satisfies the conditions of a normal form, it is said to be in that normal form.

The task of database design starts with unnormalized set of relations. The process of normalization identifies and corrects the problems and complexities of database design. It produces a new set of relations. The new design is as free of processing problems as possible.

5.14.1 Purposes of Normalization

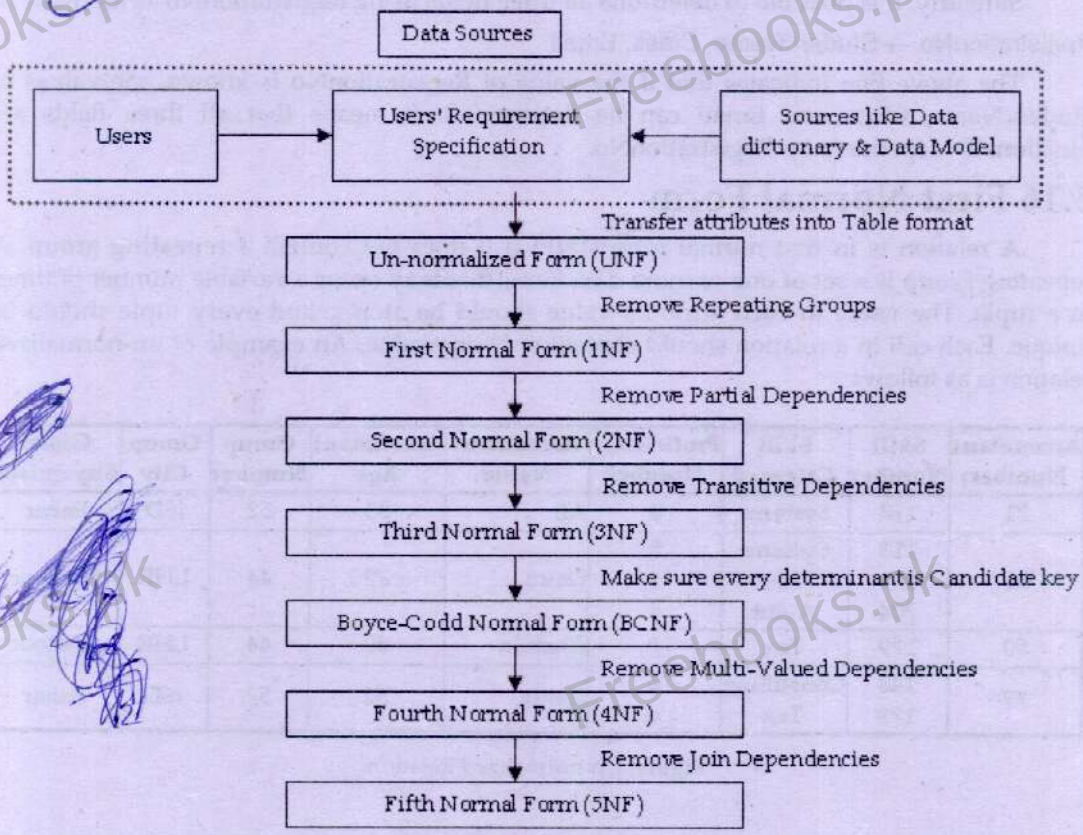
The purposes of normalization are as follows:

- It makes the database design efficient in performance.
- It reduces the amount of data if possible.
- It makes the database design free of update, insertion and deletion anomalies.
- It makes the design according to the rules of relational databases.
- It identifies relationship between entities.
- It makes a design that allows simple retrieval of data.
- It simplifies data maitenance and reduces the need to restructure data.

5.14.2 Characteristics of Normalized Database

A normalized database should have the following characteristics:

- Each relation must have a key field.
- All fields must contain atomic data.
- There must be no repeating fields.
- Each table must contain information about a single entity.
- Each field in a relation must depend on key fields.
- All non-key fields must be mutually independent.



5.15 Functional Dependency

Functional dependency is a relationship between attributes. It means that if the value of one attribute is known, it is possible to obtain the value of another attribute. Suppose there is a relation STUDENT with following fields:

STUDENT (RegistrationNo, StudentName, Class, Email)

If value of RegistrationNo is known, it is possible to obtain the value of StudentName. It means that StudentName is functionally dependent on RegistrationNo. An attribute B is functionally dependent on attribute A if the value of A determines the value of B.

Functional dependency is written as follows:

RegistrationNo → StudentName

The above expression is read as "RegistrationNo determines StudeName" or "StudentName is functionally dependent on RegistrationNo". The attribute on the left side is called **determinant**.

If A and B are attributes or sets of attributes of relation R, B is functionally dependent on A if each value of A in R has exactly one associated value of B in R.

A particular value of RegistrationNo is related to only one value of StudentName. But StudentName may be related with multiple values of RegistrationNo. For example, the RegistrationNo 10 is related with only one value of StudentName. But StudentName "Usman" may be related with two or more RegistrationNo values because two or more students may have the name "Usman".

Similarly, it is possible to determine all three fields using RegistrationNo as follows:

RegistrationNo → StudentName, Class, Email

The above line indicates that if the value of RegistrationNo is known, the values of StudentName, Class and Email can be determined. It means that all three fields are functionally dependent on RegistrationNo.

5.16 First Normal Form

A relation is in first normal form (1NF) if it does not contain a **repeating group**. A repeating group is a set of one or more data items that may occur a variable number of times in a tuple. The value in each attribute value should be atomic and every tuple should be unique. Each cell in a relation should contain only one value. An example of un-normalized relation is as follows:

Accountant Number	Skill Number	Skill Category	Proficiency Number	Accountant Name	Accountant Age	Group Number	Group City	Group Supervisor
21	113	Systems	3	Ali	55	52	ISD	Babar
35	113 179 204	Systems Tax Audit	5 1 6	Daud	32	44	LHR	Ghafoor
50	179	Tax	2	Chohan	40	44	LHR	Ghafoor
77	148 179	Consulting Tax	6 6	Zahid	52	52	ISD	Babar

Figure: Un-normalized Relation

The above relation is un-normalized because it contains repeating groups of three attributes Skill Number, Skill Category and Proficiency Number. All three fields contain more than one value.

In order to convert this relation in first normal form, these repeating groups should be removed. The following relation is in first normal form:

Primary Key

Accountant Number	Skill Number	Skill Category	Proficiency	Accountant Name	Accountant Age	Group Number	Group City	Group Supervisor
21	113	Systems	3	Ali	55	52	ISD	Babar
35	113	Systems	5	Daud	32	44	LHR	Ghafoor
35	179	Tax	1	Daud	32	44	LHR	Ghafoor
35	204	Audit	6	Daud	32	44	LHR	Ghafoor
50	179	Tax	2	Chohan	40	44	LHR	Ghafoor
77	148	Consulting	6	Zahid	52	52	ISD	Babar
77	179	Tax	6	Zahid	52	52	ISD	Babar

Figure: First Normal Form (1NF)

5.16.1 Problems in 1NF

The relation in 1NF has certain problems which are as follows:

1. Updating Problem

Suppose the user wants to change the name of Accountant Number 35 to "M. Daud". He has to the name in all records in which Accountant number 35 appears. This process of updating can be very lengthy.

2. Inconsistent Data

The above table may contain inconsistent data. There are three records of Accountant Number 35. It is possible that there are two different names with Account Number 35 in two different records. The user can make this error during updating.

3. Addition Problem

Suppose the user wants to add another skill number in the table. It is not possible until an Accountant with that skill exists because both Skill Number and Accountant Number are used as primary key in the above table.

4. Deletion Problem

Suppose the user wants to delete the record of supervisor Ghafoor. If he deletes the whole record in which Ghafoor appears, the information about Accountants will also be lost.

5.17 Full Functional Dependency

In a relation R, attribute B of R is fully functionally dependent on an attribute or set of attributes A of R if B is functionally dependent on A but not functionally dependent on any other proper subset of A.

Suppose there is a relation MARKS as follows:

MARKS (RegistratioNo, Subject, Marks)

Assume that one student is studying many subjects. Both RegistrationNo and Subject are required to determine a particular marks. It can be written as follows:

RegistrationNo, Subject → Marks

Here, Marks is fully functionally dependent on both fields because it is not functionally dependent on either RegistrationNo or Subject alone.

5.18 Second Normal Form

A relation is in Second Normal Form (2NF) if it is in 1NF and if all of its nonkey attributes are fully functionally dependent on the whole key. It means that none of non-key attributes are related to a part of key.

The above relation in 1NF has some attributes which are not depending on the whole primary key. For example, Accountant Name, Accountant Age and group information is determined by Accountant Number and is not dependent on Skill. The following relation can be created in which all attributes are fully dependent on primary key Account Number.

Primary Key

Accountant Number	Accountant Name	Accountant Age	Group Number	Group City	Group Supervisor
21	Ali	55	52	ISD	Babar
35	Daud	32	44	LHR	Ghafoor
50	Chohan	40	44	LHR	Ghafoor
77	Zahid	52	52	ISD	Babar

Figure: Accountant Table in 2NF

Similarly, another relation Skill can be created in which all fields are fully dependent on the primary key as follows:

Primary Key

Skill Number	Skill Category
113	Systems
179	Tax
204	Audit
148	Consulting

Figure: Skill Table in 2NF

The attribute Proficiency in 1NF relation was fully dependent on the whole primary key. The Proficiency requires to know the accountant number and skill number. The third relation will be created as follows:

Primary Key

Accountant Number	Skill Number	Proficiency
21	113	3
35	113	5
35	179	1
35	204	6
50	179	2
77	148	6
77	179	6

Figure: Proficiency Table in 2NF

There are three relations in second normal form (2NF). The attributes of all relations are fully dependent on the primary keys.

5.18.1 Analysis of Second Normal Form (2NF)

The following analysis indicates whether the problems are eliminated in 2NF or not.

- **Updating Problem:** If the user needs to change the name of Accountant Number 35 to "M. Daud" in 1NF, he must change the name in every record in which Accountant number 35 appears. But in 2NF, the record of one accountant appears only once. The updating problem is eliminated in 2NF.
- **Inconsistent Data:** The record of one accountant appears only once in the database, the possibility of inconsistent data is automatically eliminated.
- **Addition Problem:** In 1NF, it was not possible to enter a new Skill Number until an Accountant with that skill existed. In 2NF, any number of skills can be added in Skill relation without an accountant with that skill. It eliminates the addition problem.
- **Deletion Problem:** In 2NF, if the record of Ghafoor is deleted, it does not delete any other record.

The analysis shows that the second normal form has solved all problems of 1NF.

5.19 Transitive Dependency

Transitive dependency is a condition in which an attribute is dependent on an attribute that is not part of the primary key.

Suppose A, B, and C are attributes of a relation. If A → B and B → C then C is transitively dependent on A via B provided that A is not functionally dependent on B or C.

Suppose there is a relation BOOK as follows:

BOOK (BookID, BookDescription, CategoryID, CategoryDescription)

We can write:

BookID → CategoryID, CategoryID → CategoryDescription

A transitive dependency occurs when one non-key attribute determines another non-key attribute. In above relation, BookID determines CategoryID and CategoryID determines CategoryDescription. CategoryDescription is transitively depedent on BookID attribute.

5.20 Third Normal Form

A relation is in third normal form if it is in 2NF and if no non-key attribute is dependent on another non-key attribute. It means that all non-key attributes are functionally dependent only on primary key. There should be no transitive dependency in a relation.

In order to convert a relation to 3NF:

- Remove all attributes from the 2NF record that depend on another non-key field
- Place them into a new relation with the other attribute as the primary key.

The Accountant table in 2NF contains some attributes which are depending on non-key attributes. For example, Group City and Group Supervisor are depending on a non-key field Group Number. A new relation can be created as follows:

Primary Key

Accountant Number	Accountant Name	Accountant Age	Group Number
21	Ali	55	52
35	Daud	32	44
50	Chohan	40	44
77	Zahid	52	52

Figure: Accountant Table in 3NF

The second table created form the Accountant table in 1NF is as follows:

Primary Key

Group Number	Group City	Group Supervisor
52	ISD	Babar
44	LHR	Ghafoor

Figure: Group table in 3NF

Both Accountant table and Group table contain the attribute Group Number. This attribute is used to join both tables.

The Skill table in 2NF contains no attribute, which is depending on a non-key attribute. It is already in third normal form and will be used without any further change.

Primary Key

Skill Number	Skill Category
113	Systems
179	Tax
204	Audit
148	Consulting

Figure: Skill Table in 3NF

The Proficiency table in 2NF also contains no attribute which is depending on non-key attribute. It is already in third normal form and will be used without any further change.

Primary Key

Accountant Number	Skill Number	Proficiency
21	113	3
35	113	5
35	179	1
35	204	6
50	179	2
77	148	6
77	179	6

Figure: Proficiency Table in 3NF

5.21 Boyce-Codd Normal Form (BCNF)

A relation is in **Boyce-Codd normal form** if and only if every determinant is a candidate key. It can be checked by identifying all determinants and then making sure that all these determinants are candidate keys. BCNF is a stronger form of third normal form. A relation in BCNF is also in third normal form. But a relation in 3NF may not be in BCNF.

Primary Key

ProjectID	PartID	QuantityUsed	PartName
101	P01	20	CD-R
112	P05	6	Zip disk
194	P01	12	CD-R
194	P02	1	Box floppy disks
194	P05	3	Zip disk

Figure: PartsAndProject Table

Assume that PartName is unique. It means that no two parts can have the same name. There are now two candidate keys (ProjectID, PartID) and (ProjectID, PartName).

There are following dependencies:

(ProjectID, PartID) → QtyUsed
PartID → PartName
PartName → PartID

This relation satisfies 2NF because there are no non-key attributes that are dependent on a subset of the primary key. PartName is not a non-key attribute, it is part of a candidate key. Therefore this relation is in 2NF.

There are no transitive dependencies so the relation is in 3NF. PartID and PartName are ignored by 3NF rule because they are both key attributes.

In order to convert this relation to BCNF, all functional dependencies must be removed which have a determinant that is not a candidate key. The result is as follows:

Primary Key

PartID	PartName
P01	CD-R
P02	Box floppy disks
P05	Zip disk

Primary Key

ProjectID	PartID	QuantityUsed
101	P01	20
112	P05	6
194	P01	12
194	P02	1
194	P05	3

Figure: Tables in BCNF

5.22 Fourth Normal Form

A relation is in 4NF if it is in BCNF and has no multi-valued dependencies.

5.22.1 Multi-Valued Dependencies

A multi-valued dependency exists when a relation has at least three attributes, two of them are multi-valued and their values depend on only the third attribute.

Suppose there is a relation R(A, B, C). A multi-valued dependency exists if:

- A determines multiple values of B
- A determines multiple values of C
- B and C are independent of each other.

For example, the following relation has multi-valued dependencies:

StudentID	Subject	Interest
100	Math	Reading
100	Accounting	Reading
100	Math	Tennis
100	Accounting	Tennis
150	Economics	Jogging

Figure: Table with multi-valued Dependencies

In the above example, a student can study many subjects and can have many interests. There are four records for StudentID 100. This is because we have to use all possible combination of Subject and Interest to make it clear that a student with any subject can have any interest. Otherwise, it would appear that Reading can only occur with Math and Tennis can occur only with Accounting.

Multi-valued dependencies are written as follows:

StudentID →→ Subject
StudentID →→ Interest

5.22.2 Anomalies in Multi-Valued Dependencies

Suppose StudentID 100 decides to take another course "Economics". It requires the addition of two tuples i.e. one with combination of Reading and one with combination of Tennis as follows:

100	Economics	Reading
100	Economics	Tennis

Figure: New tuples in table

Similarly, if a student wants to drop a subject, all tuples containing that particular subject have to be deleted.

In order to eliminate these anomalies, multi-valued dependency should be eliminated. It can be done by creating two relations. Each relation will contain data for only one multi-valued attribute. The resulting relations do not have anomalies.

StudentID	Subject
100	Math
100	Accounting
150	Economics

StudentID	Interest
100	Reading
100	Tennis
150	Jogging

Figure: Elimination of Multi-Valued Dependency

5.23 Lossless Join Dependency

A property of decomposition that ensures that no spurious tuples are generated when relations are reunited through a natural join operation.

A relation that is decomposed during normalization can be rejoined by using Joins to produce the original data. Sometimes, a relation is decomposed into more than two relations. Lossless join dependency ensures that if the decomposed relations are reunited, data is not lost and no additional tuple is generated. The real focus of lossless join dependency and fifth normal form are the cases where the relation is decomposed in more than two relations.

5.24 Fifth Normal Form

A relation is in fifth normal form if it has no join dependency. Fifth normal form (5NF) is also called **project-join normal form (PJNF)**. The first four normal forms are based on the concept of functional depdency. The fifth normal is based on the concept of join dependency.

5.25 Domain Key Normal Form

A relation is in DK/NF if every constraint on the relation is a logical consequence of the definition of keys and domains. It means that a relation is in DK/NF if enforcing key and domain restrictions causes all of the constraints to be met.

DK/NF is important because a relation in DK/NF has no modification anomalies. A relation with no modification anomalies must be in DK/NF. DK/NF involves only the concepts of key and domain. These concepts are fundamental in database environment. They are readily supported by DBMS products.

Suppose there is a relation STUDENT as follows:

STUDENT(SID, NAME, MAJOR, CREDITS)

Suppose a rules thates that SID should have prefix to indicate the type of student. The prefix 1 indicates freshman, 2 indicates sophomores and so on. It can be used to express a general constraint "if first digit of SID is 1 then CREDITS must be between 0 and 30 so on." The constraint must be expressible as domain constraint or key constraint to convert a relation in DKNF. The constraint can be expressed by splitting STUDENT trelation in four different relations. For example, the following relation can be used for freshman:

STD1(SID, NAME, MAJOR, CREDITS)

The above relation will have the following constraints:

- SID must begin with 1.
- CREDITS must be between 0 and 30.

Similarly, STD2 will be created for sophomores, STD3 for juniors and STD4 for seniors.

The basic concept of DKNF is simple but there is no proven method of converting a desing to this form. It remains an ideal rather than a state that can be achieved readily.

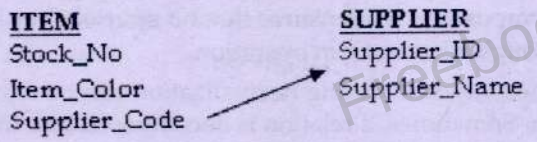
5.26 Problems in Relations

Different problems in the relations are as follows:

1. Synonym

A **synonym** is a type of problem that exists in relations. A synonym is created when two different names are used for the same information or attribute. The name of attribute must be same if it exists in two or more relations.

The following example displays two relations with synonym problem:



The above relations are inter-related. The ITEM relation contains information about the items that are supplied by the supplier. The SUPPLIER relation has an attribute Supplier_ID. The ITEM relation is referring to Supplier_ID with Supplier_Code that is wrong. It must also use Supplier_ID in order to refer to Supplier_ID attribute of SUPPLIER relation.

2. Homonym

A **homonym** is a type of problem that exists in relations. A homonym is created when same name is used for two different attributes. The following example displays two relations with homonym problem:

CUSTOMER	SUPPLIER
Company_Name	Company_Name

The attribute **Company_Name** is appearing in both relations. It may create confusion. The solution is that unique attribute names must be used in all relations to avoid confusion.

3. Redundancy

Redundancy means duplication of data in multiple files. It is a type of problem that exists in relations. It is created when the same information is unnecessarily stored in two ways or forms. The following example displays a relation with redundancy problem:

EMPLOYEE
Date_of_Birth
Age

The relation contains two attributes. The first attribute stores the date of birth of an employee. The second attribute stores the age of an employee. The age can be calculated by using the date of birth. It means that Age attribute is not necessary. It is creating redundancy in the relation and must be dropped from the relation.

4. Mutual Exclusiveness of Data

The data that does not have overlapping information is known as **mutually exclusive data**. The mutual exclusiveness of data creates problem in some cases. It creates problem for the attributes whose values can be specified as "Yes/No" form. Sometimes, two or more such attributes cannot be **true** or **false** at the same time for one entity. The following example displays a relation with this problem:

EMPLOYEE
Married
Single

The above two attributes cannot be **true** or **false** at the same. The problem occurs if "Yes" is selected in both attributes. The problem can be solved by using a larger categorical attribute. The above relation can have an attribute "MARITAL_STATUS". The possible values in this attribute can be "M" and "S" where "M" indicates "Married" and "S" indicates "Single".

EMPLOYEE
Marital_Status

Normalization Project 1

The following table depicts the set of attributes found in a database:

StdID	StdName	SocietyID	SocName	SupID	Supervisor	Position
123	Masood	001	Urdu	1	Mr. Waseem	Chairman
		003	Maths	2	Ms. Chauhdry	Member
132	Khalida	001	Urdu	1	Mr. Waseem	Member
142	Javed	002	English	1	Mr. Waseem	Member
		005	Physics	3	Mr. Liaqat	Chairman
		008	Biology	4	Miss Yasmeen	Member

Figure: Un-normalized Relation **Student_in_Society**

Solution

The above table is in un-normalized form. We will apply first three normal forms on the above relation. The solution is as follows:

First Normal Form

The **Student_in_Society** table is in un-normalized form as **SocietyID** is a multi-valued attribute. The above relation can be converted into 1NF by storing the details of the repeating groups in a separate table. This will result in the following table structures.

Student (StdID, StdName)
Student_in_Society (StdID, SocietyID, SocietyName, SupID, Superviser, Position)

StdID	StdName
123	Masood
132	Khalida
142	Javed

Student relation

StdID	SocietyID	SocName	SupID	Supervisor	Position
123	001	Urdu	1	Mr. Waseem	Chairman
123	003	Maths	2	Ms. Chauhdry	Member
132	001	Urdu	1	Mr. Waseem	Member
142	002	English	1	Mr. Waseem	Member
142	005	Physics	3	Mr. Liaqat	Chairman
142	008	Biology	4	Miss Yasmeen	Member

Student_in_Society relation

Figure: Student relation and **Student_in_Society** in 1NF

The above relations are in 1NF. However, many data anomalies still exist. Suppose that a new supervisor Mr. Khan replaces Mr. Waseem to become society teacher of Urdu Society. Two rows in **Student_in_Society** table need to be updated which is a modification anomaly. The relation also has an insertion anomaly. It is not possible to store information about a new society as no student has joined it. A deletion anomaly exists when the last member of a society quits. The society information will be permanently removed from the database.

Second Normal Form

The partial dependencies are removed from the table to convert a table to 2NF. The functional dependencies for the **Student** table are as follows:

StdID, SocietyID →Position: Full functionally dependency
SocietyID → SocName: Partial dependency as SocietyID is a part of primary key only
SocietyID → SupID: Partial dependency as SocietyID is a part of primary key only

The **Student_in_Society** table can be converted to 2NF by extracting **SocietyName, SupID, Superviser** to a separate table **Society**. These three attributes are fully functionally dependent on **SocietyID** which will be used as primary key in **Society** table as follows:

Student (StdID,StdName)
Society (SocietyID, SocietyName, SupID, Superviser)
Student_in_Society (StdID, SocietyID, Position)

StdID	StdName
123	Masood
132	Khalida
142	Javed

Student relation

SocietyID	SocName	SupID	Supervisor
001	Urdu	1	Mr. Waseem
002	English	1	Mr. Waseem
003	Maths	2	Ms. Chauhdry
005	Physics	3	Mr. Liaqat
008	Biology	4	Miss Yasmeen

Society relation

StdID	SocietyID	Position
123	001	Chairman
123	003	Member
132	001	Member
142	002	Member
142	005	Chairman
142	008	Member

Student_in_Society relation

Figure: Student, Society and **Student_in_Society** relations in 2NF

The above tables are in 2NF but are not able to solve all anomalies. The modification anomaly still exists in **Society** table. Suppose Mr. Waseem resigns and a new teacher Mr. Khan replaces him in all societies. Mr. Khan will use the same SupID of Mr. Waseem. Two rows instead of one need to be updated to reflect this change in **Society** table.

Third Normal Form

A table is in 3NF if it is in 2NF and it exhibits no transitive dependencies. In the **Society** table, **SocietyID → SupID** and **SupID → Supervisor** and thus **SocietyID → Supervisor**. It is a kind of transitive dependency. The attributes that contribute to transitive dependencies are extracted to separate table(s) to convert a table to 3NF. The **Society** table can be converted to 3NF by extracting **Supervisor** to a new table **Society_Teacher**. The attribute **Supervisor** is fully functionally dependent on **SupID**. It is copied to **Society_Teacher** table to be used as primary key. This will result in the following table structures.

Student (StdID,StdName)
Society (SocietyID, SocietyName, SupID)
Student_in_Society (StdID, SocietyID, Position)
Society_Teacher (SupID, Superviser)

StdID	StdName
123	Masood
132	Khalida
142	Javed

Student relation

SocietyID	SocName	SupID
001	Urdu	1
002	English	1
003	Maths	2
005	Physics	3
008	Biology	4

Society relation

StdID	SocietyID	Position
123	001	Chairman
123	003	Member
132	001	Member
142	002	Member
142	005	Chairman
142	008	Member

Student relation

SupID	Supervisor
1	Mr. Waseem
2	Ms. Chauhdry
3	Mr. Liaqat
4	Miss Yaseem

Society_Teacher relation

Figure: Student, Society, Student_in_Society and **Society_Teacher** relations in 3NF

Normalization Project 2

The following table depicts the set of attributes found in a University database:

StudentID	StudentName	CourseID	CourseLength	UnitCode	UnitName	Lecturer
001	Usman	A203	3	U45 U87	Databases II Programming	Ashraf Ghafoor
003	Nadeem	A104	4	U86 U45 U25	Algorithms Databases II Business I	Naveed Ashraf Raoof
007	Abdullah	A203	3	U12 U46	Business II Databases I	Abid Zahid
010	Adnan	A323	2	U12 U86	Business II Algorithms	Abid Naveed

Figure: Un-normalized Relation

Note that a student attends one course and can take any units during the course. A unit may be presented as part of any course and is always given by one particular lecturer.

Solution

The above table contains un-normalized table. We will apply first three normal forms on the above relation. The solution is as follows:

First Normal Form

To convert the above relation into first normal form (1NF), we need to remove all repeating groups. The set of attributes which repeat for each value of StudentID are UnitCode, UnitName, and Lecturer. In order to remove the repeating groups, we can use two approaches:

First Approach

In the first approach, as used in the previous example, we enter data in the empty columns of rows that contain the repeating data. It removes repeating groups and now the relation contains atomic value in each field. In this approach, we get the problem of data redundancy that is solved in the next steps of normalization.

Second Approach

One approach to do so is to split the relation in two relations to eliminate the repeating groups. We remove the repeating group by placing the repeating data along with a copy of original key attributes in a separate relation. A primary key is identified for the new relation. This approach produces relations with less redundancy. This example uses the second approach as follows:

Primary Key

StudentID	StudentName	CourseID	CourseLength
001	Usman	A203	3
003	Nadeem	A104	4
007	Abdullah	A203	3
010	Adnan	A323	2

Primary Key

StudentID	UnitCode	UnitName	Lecturer
001	U45	Databases II	Ashraf
001	U87	Programming	Ghafoor
003	U86	Algorithms	Naveed
003	U45	Databases II	Ashraf
003	U25	Business I	Raoof
007	U12	Business II	Abid
007	U46	Databases I	Zahid
010	U12	Business II	Abid
010	U86	Algorithms	Naveed

Figure: Student table and Unit table in First Normal Form

Second Normal Form

To convert the above relation into second normal form (2NF), we need to remove all partial dependencies. The Student relation does not have a composite primary key and, therefore, cannot contain partial dependencies. So, the Student is already in second normal form. The Unit relation has the following dependencies:

UnitCode → UnitName
UnitCode → Lecturer
StudentID, UnitCode → UnitName, Lecturer

Lecturer is only determined by UnitCode. Therefore, a partial dependency exists between UnitCode and UnitName and Lecturer. Removing the partial dependencies from Unit produces the following relations:

Primary Key

UnitCode	UnitName	Lecturer
U45	Databases II	Ashraf
U87	Programming	Ghafoor
U86	Algorithms	Naveed
U25	Business I	Raoof
U12	Business II	Abid
U46	Databases I	Zahid

Unit Table

Primary Key

StudentID	UnitCode
001	U45
001	U87
003	U86
003	U45
003	U25
007	U12
007	U46
010	U12
010	U86

Take Table

Figure: Tables in Second Normal Form (2NF)

Third Normal Form

To convert the above relation to third normal form (3NF), we need to remove all transitive dependencies. The Student table contains the following functional dependencies:

StudentID → StudentName, CourseCode, CourseLength
CourseCode → CourseLength

Therefore, the following transitive dependency exists:

StudentID → CourseCode → CourseLength

Removing this transitive dependency from Student produces the following relations:

Primary Key

StudentID	StudentName	CourseID
001	Usman	A203
003	Nadeem	A104
007	Abdullah	A203
010	Adnan	A323

Student Table

Primary Key

CourseID	CourseLength
A203	3
A104	4
A323	2

Course Table

Figure: Tables in Third Normal Form (3NF)

The Take relation contains no non-key attributes and so contains no transitive dependencies. The Unit relation contains the following dependencies:

UnitCode → UnitName, Lecturer

Therefore, there are no transitive dependencies in Unit. The final relations in third normal form are as follows:

Primary Key

StudentID	StudentName	CourseID
001	Usman	A203
003	Nadeem	A104
007	Abdullah	A203
010	Adnan	A323

Student Table

Primary Key

CourseID	CourseLength
A203	3
A104	4
A323	2

Course Table

Primary Key

UnitCode	UnitName	Lecturer
U45	Databases II	Ashraf
U87	Programming	Ghafoor
U86	Algorithms	Naveed
U25	Business I	Raoof
U12	Business II	Abid
U46	Databases I	Zahid

Unit Table

Primary Key

StudentID	UnitCode
001	U45
001	U87
003	U86
003	U45
003	U25
007	U12
007	U46
010	U12
010	U86

Take Table

Figure: Final tables in Third Normal Form (3NF)

Short Questions

Q.1. Contrast the following:

a. Candidate key and primary key
b. Functional dependency and transitive dependency
c. Foreign key and primary key

a. **Candidate Key and Primary Key:** A primary key is an attribute or combination of attributes that uniquely identifies a row in a relation. When a relation has more than one such attribute or combination of attributes, each is called a candidate key.

b. **Functional Dependency and Transitive Dependency:** A functional dependency is a relationship between any two attributes or two sets of attributes. A transitive dependency is a functional dependency between two or more non-key attributes.

c. **Foreign Key and Primary Key:** A primary key uniquely identifies each row in a relation. a foreign key is an attribute or set of attributes in a relation that reference a primary key in another table.

Q.2. What is the difference between a candidate key and a superkey?

A superkey may not be minimal. A candidate key is a minimal superkey. A superkey is minimal if it does not remain unique by removing a column.

Q.3. What is the relationship between the referential integrity rule and foreign keys?

Referential integrity involves the values that can be used in foreign keys. A foreign key can store a value matching a candidate key value in some row of the table containing the associated candidate key or a null value.

Q.4. What is a primary key? Are duplicate primary keys allowed? Why or why not?

A primary key is a field or set of fields that can uniquely identify a row of a table. Duplicate key values are not allowed because primary keys must uniquely identify a row.

Q.5. What is a foreign key? Why are foreign keys used in a relational database? Are duplicate foreign key values allowed? Why or why not?

A foreign key is a field or set of fields stored in one table that also exists as a primary key value in another table. Foreign keys are used to represent relationships among entities that are represented as tables. Duplicate foreign keys are not allowed within the same table because they would redundantly represent the same relationship. Duplicate foreign keys may exist in different tables because they would represent different relationships.

Q.6. What are the difference between partial dependency and transitive dependency?

A partial dependency exists when an attribute is dependent on only a part of primary key. It is associated with 1NF. Transitive dependency is a condition in which an attribute is dependent on another attribute that is not part of primary key. It usually requires the decomposition of the table containing the transitive dependency.

Q.7. What restrictions must be placed on a table to consider it a relation?

- Rows contain data about an entity
- Columns contain data about attributes of the entity
- Cells of the table hold a single value
- All entries in a column are of the same kind
- Each column has a unique name
- The order of the columns in unimportant
- The order of the rows in unimportant
- No two rows may be identical

Q.8. Define relation, tuple, attribute, file, record, field, table, row and column.

- A relation is a two-dimensional table that has the characteristics.
- A tuple is one row of a table.
- An attribute is one column in a table.
- A file is a term use by many people to a table.
- A record is the same as row of a table
- A field is the same as a column of a table.
- A table is a relation. Generally these terms are interchangeable.
- A row contains all data about one instance of an entity represented in a table.
- A column contains all values assigned to an attribute in a table.

Q.9. What is a view? Discuss the difference between a view and a base relation.

View is the dynamic result of one or more relational operations operating on the base relations to produce another relation. Base relation exists as a set of data in database. A view does not contain any data. It is defined as a query on one or more base relations. The query on view is translated into a query on the associated base relations.

Q.10. Describe the purpose of normalizing data.

The process of producing a simpler and more reliable database structure is called **normalization**. It is used to create a suitable set of relations for storing data. The process of normalization identifies and corrects the problems and complexities of database design. It produces a new set of relations. The new design is as free of processing problems as possible.

Q.11. How is functional dependency associated with the process of normalization?

Normalization is a technique to analyze relations based on primary key and functional dependencies. Normalization is often performed as a series of tests on a relation to determine whether it satisfies or violates the requirements of a given normal form. Three normal forms were initially proposed which are called 1NF, 2NF and 3NF. All of these normal forms are based on the functional dependencies among the attributes of a relation.

Q.12. What is well-structured Relation? Why are well-structured relations important in logical database design?

A well-structured relation is a relation that contains a minimum amount of redundancy. It allows users to insert, modify and delete rows in a table without errors or inconsistency. Well-structured relations are important because they promote database integrity.

Q.13. Which three data anomalies are likely to be the result of data redundancy? How can such anomalies be eliminated?

The most common anomalies considered when data redundancy exists are update anomalies, addition anomalies and deletion anomalies. All these can easily be avoided through normalization. Data redundancy produces data integrity problems.

Q.14. What is a deletion anomaly? Give an example.

A deletion anomaly is a case where deletion of facts about one entity instance deletes facts about another entity instance. The user can lose facts about two entities with one deletion.

Q.15. What is an insertion anomaly? Give an example.

An insertion anomaly is a case where the user cannot insert a fact about one entity until he has an additional fact about another entity.

Q.16. What is the cause of modification anomalies?

Poor database design causes the modification anomaly. A good database design avoids modification anomalies by eliminating excessive redundancies.

Q.17. Describe the characteristics of an unnormalized table. How such table is converted to first normal form (1NF)?

A table in unnormalized form contains one or more repeating groups. It is converted to first normal form by removing the repeating group to a new relation along with a copy of original key attribute(s). The repeating group can be removed by entering appropriate data in the empty columns of rows containing the repeating data.

Q.18. What is a normal form?

A normal form is a rule about allowable dependencies. Each normal form removes certain kinds of redundancies.

Q.19. What does 1NF prohibit?

1NF prohibits repeating groups in tables. A table not in 1NF is unnormalized or non-normalized.

Q.20. What is a key column?

A column is a key column if it is a candidate key or part of a candidate key.

Q.21. What is a non-key column?

A column is a non-key column if it is not a key column.

Q.22. Define functional dependency. Give an example of two attributes with functional dependency and give an example of two attributes with no functional dependency.

A functional dependency is a relationship between attributes such that given the value of one attribute it is possible to determine the value of the other attribute. Example of functional dependency: Name--->Phone#. Example of attributes that are not functionally dependent: Age and Address.

Q.23. Which normal forms rely on the definition of functional dependency?

Second and third normal forms are based on a concept called functional dependency—a one-to-one correspondence between two field values. Second normal form ensures that every field in a table is functionally dependent on the primary key. Third normal form ensures that no non-key field is functionally dependent on any other non-key field.

Q.24. What is a view? How it is related to data independence?

A view is a virtual table that does not really exist in its own. It is derived from one or more base table. There is no stored file that represents the view. A definition of view is stored in data dictionary. The view can insulate users from the effects of restructuring and growth in database. It provides logical data independence. .

Q.25. Define determinant.

A determinant is an attribute whose value enables us to obtain the value(s) of other related attributes. It appears on the left side of a functional dependency. Thus, in A--->B, the determinant is A.

Q.26. What is removed when a relation is converted to the first normal form?

All repeating groups are removed and the primary keys are identified when a relation is converted to the first normal form.

Q.27. What is removed when a relation is converted from 1NF to 2NF?

All the partially dependent attributes are removed and placed in another relation when a relation is converted from 1NF to 2NF.

Q.28. What is difference between normal form and normalization?

Normal form is a state of a particular relation regarding functional dependencies while the normalization is a process of producing a simpler and more reliable database structure.

Q.29. What kinds of functional dependencies are not allowed in 2NF?

Functional dependencies in which part of a key determines a nonkey column are not allowed in 2NF.

Q.30. List three conditions that can be applied to determine that a relation in first normal form is also in Second Normal Form.

Three conditions that imply a relation is in second normal form:

1. The primary key consists of a simple attribute.
2. No non-key attributes exist in the relation.
3. Every non-key attribute is functionally dependent on the full set of primary key attributes.

Q.31. What is removed when a relation is converted from 2NF to 3NF?

Any transitive dependencies, nonkey attributes dependent on other nonkey attributes, are removed when a relation is converted from 2NF to 3NF.

Q.32. What kinds of functional dependencies are not allowed in 3NF?

Functional dependencies in which a nonkey column determine another nonkey column are not allowed in 3NF.

Q.33. What kinds of functional dependencies are not allowed in BCNF?

Functional dependencies in which the determinant is not a candidate key are not allowed in BCNF..

Q.34. Define second normal form. Give an example of a relation in 1NF but not in 2NF. Transform the relation into relations in 2NF.

A relation is in second normal form if all its non-key attributes are dependent on the entire key. A table that is in second normal form cannot have a partial dependency.

Assume: COURSE (Dept_Code, Course_Number, Course_Title, Credits, Department_Name)

The table is in first normal form because it has no repeating attributes. Since the key is composite, it is not in second normal form if Dept_Code → Department_Name. Only part of primary key determines an attribute. To put the table in second normal form use the following tables.

COURSE (Dept_Code, Course_Number, Course_Title, Credits)

DEPARTMENT (Dept_Code, Department_Name)

Q.35. Define third normal form. Give an example of a relation in 2NF but not in 3NF. Transform the relation into relations in 3NF.

A relation is in third normal form if it is in second normal form and has no transitive dependencies. All determinants must be keys.

Assume: STUDENT (SIDU, Stu_Name, P_O_Box, Major, Hours_Required)

The table is in first normal form because it has no repeating attributes. The table is in second normal form because it has a single attribute key. If you assume Major → Hours_Required them Major is a determinate but not the key therefore you have a transitive dependency. To put the table in second normal form use the following tables.

STUDENT (SID, Stu_Name, P_O_Box, Major)

DEPARTMENT (Major, Hours_Required)

Q.36. What are the special cases covered by BCNF but not by 3NF?

BCNF covers two special cases not covered by 3NF: (1) part of a key determines part of a key and (2) a nonkey column determines part of a key.

Q.37. The special cases covered by BCNF but not by 3NF are significant?

The special cases are not significant because they rarely occur.

Q.38. Is a relation with no non-key attribute is always in third normal form?

This is a special case of 3NF. A 3NF relation typically involves a functional dependency between two non-key attributes, and if no such non-key attributes are present then the relation must be 3NF.

Q.39. Can foreign keys contain null values? Explain why via an example.

Yes foreign keys should be allowed to have null values because there are situations in which such kind of information is not available as follows:

- Employees that do not belong to a specific department, yet;
- Products that have not been put on the shelves, yet;
- Students that have not chosen some options, yet;

Q.40. Define BCNF. Give an example of a relation in 3NF but not in BCNF. Transform the relation into relations in BCNF.

A relation is in BCNF if every determinant is a candidate key. Consider this relation:

FAC-OFFICE (FID, Department, Building, Office).

Assume faculty members in the same department have offices in same building. The key is FID that determines Department, Building and Office. Department is not a candidate key and it determines Building. These relations are in BCNF:

FACULTY (FID, Department, Office)

DEPARTMENT-LOCATIONS (Department, Building)

Q.41. Define fourth normal form. Give an example of a relation in BCNF but not in 4NF. Transform the relation into relations in 4NF.

A relation is in 4NF if every determinant is a candidate key and it has no multi-valued dependencies. The following relation is in BCNF but not 4NF:

EMPLOYEE-HISTORY (Name, Project, PublicServiceActivity).

Project can be multi-valued because an employee could have worked on many projects. PublicServiceActivity can also be multi-valued. But Project and PublicServiceActivity are unrelated. These relations are in 4NF:

EMPLOYEE-WORK-HIST (Name, Project).

EMPLOYEE-SERVICE-HIST (Name, PublicServiceActivity)

Q.42. What is the relationship between BCNF and 3NF? Is BCNF a stricter normal form than 3NF? Briefly explain your answer.

Ans: BCNF is the revised 3NF definition. Yes BCNF is stricter than 3NF in that every table in BCNF is in 3NF but not every table in 3NF is in BCNF.

Q.43. Why is the BCNF definition preferred to the 3NF definition?

Ans: BCNF is the preferred definition because it is a simpler definition and provides the basis for the simple synthesis algorithm.

Q.44. How many columns does an MVD involve?

An MVD involves three columns.

Q.45. What is a multivalued dependency (MVD)?

A multivalued dependency is a relationship that can be derived from other relationships.

Q.46. What is the relationship between MVDs and FDs?

MVDs are generalizations of FDs. Every FD is an MVD, but not every MVD is an FD.

Q.47. What is minimal normal form a relation must satisfy? Define this normal form.

The minimal normal form is 1NF. A relation in which the intersection of each row and column contains one and only one value.

Q.48. What is meant by 'union compatible'? Which operations require tables to be union compatible?

The tables are 'union compatible' if they have the same degree (number of attributes) and the domain of the corresponding attributes is same. Union, intersection and difference operations require tables to be union compatible.

Q.49. Why is it important to study the operators of relational algebra?

Knowledge of operators can improve query formulation skills. The operators are fundamental ways to retrieve data from a relational database. The study of operators of relational algebra is important to become proficient in query formulation.

Q.50. Why are the restrict and project operators widely used?

These operators are widely used because users often want to see a subset rather than an entire table. These operators are also popular because they are easy to understand.

Q.51. Why is the join operator so important for retrieving useful information?

It is important for retrieving useful information because information resides in different tables and combining tables is important. The join operator is used to combine tables to retrieve information.

Q.52. What happens to unmatched rows with the join operator?

Unmatched rows are removed from the result of the join operator. Unmatched rows are included in the result of the full outer join operator.

Q.53. State the difference between the following:

a. Union & Intersection
b. Product & join
c. Selection and projection

a. The **Union** operation on two tables returns all rows from both the tables, but it does not repeat the duplicate rows. The Intersection operation on two table returns rows that exist in both tables involved.

b. A **product** of two table matches every row from the first table to each and every row in the second table. The **join** operation on two tables joins rows from two tables based on a common attribute, and the rows with same common attribute value are joined to each other.

c. The **Selection** operation on a table returns rows that satisfy the supplied criteria. The **projection** operation returns the attributes specified from a table.

Q.54. What is the closure property of the relational algebra?

The operators of the relational algebra are closed over relations, in the same sense as that in which those of arithmetic are closed over numbers. In arithmetic, plus, minus, times and divide each operates on a pair of numbers and when invoked yields a number. It is this so-called closure property that allows us to use an invocation of an operator as an operand of another invocation. By repeatedly doing that we can build numerical expressions of arbitrary complexity. In relational algebra, each operator operates on one or more relations and when invoked yields a relation, thus allowing relational expressions of arbitrary complexity to be written.

Q.55. An organization works on a number of Projects, each identified by ProjectNo. There are number of employees identified by SIN who work on these projects. An employee can be working on more than one project at one time. The number of hours worked are also recorded. A preliminary table has been constructed as follows

SIN	ProjectNo	EmployeeName	ProjectName	EmpHours

Assuming the relation to be in 1NF,

a. What should be the Primary Key? Is it Composite?
b. Write down the Partial dependencies.

c. Split the relation such that the resulting relations are in 3NF.

a) **(SIN, ProjectNo)**

b) **SIN → EmpoyeeName, ProjectNo → ProjectName**

c)

SIN	ProjectNo	EmpHours

SIN	EmployeeName

ProjectNo	ProjectName

Q.56. A table contains sample data for parts and for vendors who supply parts. The part numbers uniquely identify parts and that vendor names uniquely identify vendors.

Part_No.	Description	Vendor_Name	Address	Unit_Cost
1234	Logic chip	Ejaz	Peoples colony	10.00
		Naeem	Medina town	8.00
5678	Memory chip	Ali Raza	Peoples colony	3.00
		Anjum	Raza Abad	2.00
		Nasir	Saeed colony	5.00

a. Convert this table to a relation (named PART SUPPLIER) in first normal form. Illustrate the relation with the sample data in the table.

b. List the functional dependencies in PART SUPPLIER and identify a candidate key.

c. For the relation PART SUPPLIER, identify each of the following: an insert anomaly a delete anomaly and a modification anomaly.

d. Draw a relational schema for PART SUPPLIER and show the functional dependencies.

e. In what normal form is this relation?

a. PART SUPPLIER

Part_No	Description	Vendor_Name	Address	Unit_Cost
1234	Logic Chip	Ejaz	Peoples colony	10.00
1234	Logic Chip	Naeem	Medina town	8.00
5678	Memory Chip	Ali Raza	Peoples colony	3.00
5678	Memory Chip	Anjum	Raza Abad	2.00
5678	Memory Chip	Nasir	Medina town	5.00

b.

Part_No → Description
Vendor_Name → Address
Part_No, Vendor_Name → Unit_Cost

c.

Insert anomaly: It is not possible to insert a new vendor before including a part number.
Delete anomaly: If part information is deleted, information about a vendor who supplies that part is also lost.
Modification anomaly: If a vendor address changes, all records for that vendor has to be modified.

d.

e. 1NF

Q.57. Examine the Patient Medication Form for Civil Hospital as follows:

Civil Hospital
Patient Medication Form

Patient Number: 9876

Full Name: Ali Ahmad **WardNumber: W11**
BedNumber: 87 **Ward Name: Fatima**

Drug Number	Name	Description	Dosage	Method of Admin	Units per Day	Start Date	Finish Date
10223	Morphine	Pain killer	10mg/ml	Oral	50	24/03/01	25/04/02
10334	Tetracycline	Antibiotic	0.5mg/ml	IV	10	24/03/01	17/04/01
10223	Morphine	Pain killer	10mg/ml	Oral	10	25/04/02	02/05/03

a. Identify the functional dependencies represented by the data shown in the form in
b. Describe and illustrate the process of normalizing the data shown in Figure to first (1NF), second (2NF), third (3NF), and BCNF
c. Identify the primary, alternate, and foreign keys in your BCNF relations

a)

Patient No → Full Name

Ward No → Ward Name

Drug No → Name, Description, Dosage, Method of Admin

Patient No, Drug No, Start Date → Units per Day, Finish date

Functional dependencies for Bed No are unclear. If Bed No is unique number for entire hospital then Bed No → Ward No. Bed No is concerned with allocation of patients on waiting list to beds.

b)

First Normal Form

Patient No, Drug No, Start Date, Full Name, Ward No, Ward Name, Bed No, Name, Description, Dosage, Method of Admin, Units per Day, Finish Date

Second Normal Form

Patient No, Drug No, Start Date, Ward No, Ward Name, Bed No, Units per Day, Finish Date

Drug No, Name, Description, Dosage, Method of Admin

Patient No, Full Name

Third Normal Form/BCNF

Patient No, Drug No, Start Date, Ward No, Bed No, Units per Day, Finish Date

Drug No, Name, Description, Dosage, Method of Admin

Patient No, Full Name

Ward No, Ward Name

c)

Patient No(FK), Drug No(FK), Start Date, Ward No(FK), Bed No, Units per Day, Finish Date

Drug No, Name, Description, Dosage, Method of Admin

Patient No, Full Name

WARD (Ward No, Ward Name)

The primary keys underlined.

Q.58. Consider the following relation definition and sample data:

ProjectID	EmployeeName	EmployeeSalary
100a	Adnan	65000
100a	Salman	5100
100b	Salman	51000
200a	Adnan	64000
200b	adnan	64000
200c	Amir	28000
200c	Salman	51000
200d	Amir	28000

PROJECT (ProjectID, EmployeeName, EmployeeSalary)

where ProjectID is the name of a work project

EmployeeName is the name of an employee who works on that project

EmployeeSalary is the salary of the employee whose name is EmployeeName

Assuming that all of the functional dependencies and constraints are apparent in data, which of the following statements is true?

A. **ProjectID → EmployeeName**
FALSE: There are multiple Employee Names for each project. (see project 100A)

B. **ProjectID → EmployeeSalary**
FALSE: There are multiple Employee Salaries for each project. (see project 100a)

C. **(ProjectID, EmployeeName) → EmployeeSalary**
FALSE: Each employee's salary is always the same, regardless of the project. (see salman)

D. **EmployeeName → EmployeeSalary**
TRUE: An Employee Name always has the same Salary. (see salman)

E. **EmployeeSalary → ProjectID**
FALSE: There are multiple ProjectIDs for a given Salary. (see Salary 51000)

F. **EmployeeSalary → (ProjectID, EmployeeName)**
FALSE: There are multiple ProjectID and EmployeeName combinations for a given Salary (see Salary 51000)

Answer these questions:

G. **What is the key of PROJECT?**

ProjectID and EmployeeName (Composite Key)

H. **Are all non-key attributes (if any) dependent on the whole key?**
No: EmployeeSalary is non-key attribute and it is dependent on EmployeeName only.

I. **In what normal form is PROJECT?**

Project is in first normal form because it has no multi-valued attributes. It is not in second normal form because it has a partial dependency. Key: ProjectID+EmployeeName but EmployeeName → EmployeeSalary.

J. **Describe two modification anomalies from which PROJECT suffers.**
Insertion Anomaly: It is not possible to add an Employee until Employee is assigned to a Project. Likewise, you cannot add a Project until and Employee is assigned to the Project.

Update Anomaly: If you want to change Salman's Salary you need to change three rows of data in order to change one Employee's salary.

Deletion Anomaly: If Amir did not work on Project 200C and worked in Project 200D only, deletion of ProjectC would delete the fact that Amir's salary was 28000.

K. **Is ProjectID a determinant?**
No

L. **Is EmployeeName a determinant?**
YES: EmployeeName → EmployeeSalary

M. **Is (ProjectID, EmployeeName) a determinant?**
No

N. **Is EmployeeSalary a determinant?**

No: In this case, it appears that it can be a determinate because no two people have same salary. Using logic however, one can assume that there is no business rule in a firm that says two people cannot have the same salary.

O. **Does this relation contain a partial dependency? If so, what is it?**

YES, the relation does contain a partial dependency. The key is ProjectID+EmployeeName but EmployeeName → EmployeeSalary.

P. **Redesign this relation to eliminate the modification anomalies.**
PROJECT (ProjectID, EmployeeName)
EMPLOYEE (EmployeeName, EmployeeSalary)

Q.59. Consider the following relation definition and sample data:

PROJECT-HOURS Relation

EmployeeName	ProjectId	TaskID	Phone	Total Hours
SALMAN	100A	B-1	788792	12
SALMAN	100A	P-1	788792	12
SALMAN	200B	B-1	788792	12
SALMAN	200B	P-1	788792	12
ADNAN	100A	C-1	788988	26
ADNAN	200A	C-1	788988	26
ADNAN	200D	C-1	788988	26

PROJECT-HOURS (EmployeeName, ProjectID, TaskID, Phone, TotalHours)
where EmployeeName is the name of an employee
ProjectID is the name of a project
TaskID is the name standard work task
Phone is the employee's telephone number
TotalHours is the hours worked by the employee on this project
Assuming that all of the functional dependencies and constraints are apparent in this data, which of the following statements is true?

A. **EmployeeName → ProjectID**
NO: There are multiple ProjectIDs for each EmployeeName

B. **EmployeeName →→ ProjectID**
YES: Each EmployeeName has three or more ProjectIDs

C. **EmployeeName → TaskID**
NO: There are multiple TaskIDs for each EmployeeName

D. **EmployeeName →→ TaskID**
YES: Salman has multiple (2) TaskIDs

E. **EmployeeName →: Phone**
YES: Each EmployeeName has exactly one Phone value

F. **EmployeeName →: TotalHours**
YES: Each EmployeeName has exactly one TotalHours value

G. (EmployeeName, ProjectID) → TotalHours

YES: It is true based on fourth assumption. Looking at data only, it is more probable that EmployeeName → TotalHours. It is because TotalHours is same for EmployeeName regardless of ProjectID.

H. (EmployeeName, Phone) → TaskID

NO: There are multiple TaskIDs for a given EmployeeName, Phone combination

I. ProjectID → TaskID

NO: There are multiple TaskIDs for a given ProjectID

J. TaskID → ProjectID

NO: There are multiple ProjectIDs for a given TaskID

Answer these questions:

K. What are all of the determinants?

EmployeeName → Phone
EmployeeName → → TaskID
EmployeeName, ProjectID → TotalHours

L. Does this relation contain a partial dependency? If so, what is it?

YES, it contains a partial dependency. The key is EmployeeName + ProjectID + TaskID but EmployeeName → Phone. So there is a partial dependency.

M. Does this relation contain a multi-value dependency? If so, what are the unrelated attributes?

YES: The related attributes are ProjectID and TaskID. EmployeeName → → ProjectID and EmployeeName → → TaskID

N. What is the deletion anomaly that this relation contains?

If Employee Salman no longer has TaskID B-1 two rows must be deleted, Row 1 and Row 3. The deletion of one fact requires deletion or two rows.

O. How many themes does this relation have?

It would appear that there are at least three themes. 1) Employees and their Tasks 2) Employees and their Phone Numbers and 3) Employees and hours worked on a project.

P. Redesign this relation to eliminate the modification anomalies. How many relations did you use? How many themes does each of your new relations contain?

EMPLOYEE (EmployeeName, Phone)
EMPLOYEE_TASKS (EmployeeName, TaskID)
PROJECT-HOURS (EmployeeName, ProjectID, TotalHours)

Three relations are required, one for each theme. Each relation now carries one theme

Multiple Choice

1. **The columns of a table correspond to:**
   a. Table b. Record c. Field d. Cell
2. **The foreign key is found in:**
   a. Parent table b. Dependent table c. Pivot table d. Index table
3. **A table must have:**
   a. Primary key b. secondary key c. Composite key d. Sort key
4. **A two-dimensional table of data is called a:**
   A. Group b. Set c. Declaration d. Relation
5. **A table is perceived as two-dimensional structure composed of:**
   a. x and y co-ordinates b. Matrix elements
   c. Rows and columns d. Intersection of data
6. **A relation is analogous to a:**
   a. File b. Field c. Record d. Row

7. **A relation is also known as:**
   a. Table b. Tuple c. Relationship d. Attribute
8. **An attribute is also known as a:**
   a. Table b. Relation c. Row d. Field
9. **A row of a relation is called a (n):**
   a. Attribute b. Entity c. Tuple d. Both a and c
10. **A person, place, thing, event, or condition about which data is kept in the database is called:**
    a. Attribute b. Field c. Record d. Entity
11. **A category of data or information that describes an entity is called a(n):**
    a. Attribute b. Data item c. Record. d. Tuple
12. **Which of the following is NOT a general characteristic of relations?**
    a. Each row is unique. b. The order of columns is significant
    c. The order of rows is insignificant d. Columns are all elemental or atomic.
13. **An index can be used to:**
    a. Improve the performance of the database
    b. Document the structure of the database itself
    c. Reduce data dependency for application programs
    d. All
14. **Which of the following are properties of relations?**
    a. Each attribute has a unique name
    b. No two rows in a relation are identical
    c. There are no multivalued attributes in a relation
    d. All of the above
15. **A key is:**
    a. a field that identifies only one record b. the most important field in a record
    c. the first field of table d. None
16. **Which of the following describes the primary key?**
    a. It must be unique b. It helps in indexing of a large database
    c. It makes sorting quicker d. All of the above
17. **Which of the following is NOT a good primary key?**
    a. Social security number b. Order number c. Zip code d. Student ID number
18. **How many primary keys can a table have?**
    a. One b. At least one, but not more than two c. Between 1 and 5 d. No limit
19. **Which of the following should be a primary key?**
    a. A person's last name b. An employee's salary
    c. A customer's ID number d. A salesperson's region
20. **Which field listed below is the most appropriate primary key?**
    a. A person's name b. A person's street address
    c. A person's birth date d. A person's Social Security Number
21. **One field or combination of fields for which more than one record may have the same combination of values is called:**
    a. Secondary key b. Index c. Composite key d. Linked key
22. **A candidate key is:**
    a. Primary key
    b. The primary key selected to be the key of a relation
    c. An attribute or group of attributes that can be the primary key
    d. All

23. An attribute in a relation of a database that serves as the primary key of another relation in the same database is called a:
a. Global key b. Link key c. Foreign key d. None

24. A primary key that consists of more than one attribute is called a:
a. Foreign key b. Composite key c. Multivalued key d. Global key

25. In 3NF, which form of dependency is removed?
a. Functional b. Non-functional c. Associative d. Transitive

26. In relational database, table is also called:
a. Tuple b. Relation c. File d. Schema

27. In 3NF, a non-key attribute must not depend on a:
a. Non-key attribute b. key attribute c. Composite key d. Sort key

28. Different attributes in two different tables having same name are referred to as:
a. Synonym b. Homonym c. Acronym d. Mutually exclusive

29. Every relation must have:
a. Primary key b. Candidate key c. Secondary key d. Mutually exclusiveness

30. The entity integrity rule states that:
a. No primary key attribute can be null b. Each entity must have a primary key
c. Primary key must have only one attribute. d. None

31. A rule that states that each foreign key value must match a primary key value in the other relation is called:
a. Referential integrity constraint b. Key match rule
c. Entity key group rule d. Foreign/primary match rule

32. Two or more attributes having different names but same meaning are called:
a. Homonyms. b. Aliases c. Synonyms d. Alternate attributes

33. A constraint between two attributes is called a(n):
a. Functional relation b. Attribute dependency.
c. Functional dependency. d. Functional relation constraint.

34. The attribute on the left-hand side of the arrow in a functional dependency is:
a. Candidate key b. Determinant c. Foreign key d. Primary key

35. The goal of normalization is to:
a. Get stable data structure b. Increase number of relation c. Increase redundancy d. None

36. A relation is in 2NF if it is in 1NF and all its non-key attributes are:
a. Dependent on part of the primary key b. Dependent on the entire primary key
c. Independent of the primary key d. Independent of any other relation

37. In 2NF, which form of dependency is removed?
a. Functional b. Partial c. Associative d. Transitive

38. Which of the following are anomalies that can be caused by redundancy in tables?
a. Insertion b. Deletion c. Modification d. All

39. A functional dependency between two or more non-key attributes is called:
a. Partial functional dependency b. Partial non-key dependency
c. Transitive dependency d. None

40. Which of the following anomalies result from a transitive dependency?
a. Insertion b. Modification c. Deletion d. All

41. A relation is in third normal form if it is in second normal form and:
a. Dependent on part of the key b. Dependent on all of the key
c. Independent of the key d. Has no transitive dependencies

42. A relation that contains minimal redundancy and allows easy use is called:
a. Clean. b. Simple. c. Complex. d. Well-structured.

43. The 1NF describes the tabular format in which:
a. All the key attributes are defined
b. No repeating groups in the table
c. All attributes are dependent on the primary key
d. All

44. In a functional dependency, the determinant:
a. Will be paired with one value of the dependent attribute
b. May be paired with one or more values of the dependent attribute
c. May consist of more than one attribute
d. a and c

45. When the determinant contains two attributes:
a. The first attribute determines the dependent attribute
b. The second attribute determines the dependent attribute
c. Both attributes determine the dependent attribute
d. Either the first or second attribute determines the dependent attribute

46. An anomaly in a relation is:
a. An unusual data value
b. A duplicate data value caused by changing the data
c. An undesirable consequence of changing the data
d. An error in the design

47. Restrictions on operations on a relation are called:
a. Deletion anomalies
b. Insertion anomalies
c. Modification anomalies
d. Referential integrity constraints

48. The normalization process generally:
a. Reduces the number of relations
b. Increases the number of relations
c. Reduces the number of functional dependencies
d. Increases the number of functional dependencies

49. A relation is automatically in:
a. First Normal Form
b. Second Normal Form
c. Third Normal Form
d. Boyce-Codd Normal Form

50. A relation is in domain/key normal form if:
a. Every key of relation is a logical consequence of definition of constraints and determinants
b. Every key of relation is a logical consequence of definition of constraints and domains
c. Every constraint on relation is a logical consequence of definition of keys and determinants
d. Every constraint on relation is a logical consequence of definition of keys and domains

51. Which of the following is TRUE from functional dependency shown as (A, B) → (C, D)?
a. A is the determinant of C
b. A and B together are determined by C and D together
c. A and B together determine D
d. C and D together determine A

52. What is the highest normal form a relation is in if every determinant is a candidate key?
a. First b. Second c. Third d. BCNF

53. A relation is considered a:
a. Column
b. One-dimensional table.
c. Two-dimensional table
d. Three-dimensional table

54. A functional dependency is a relationship between or among
a. Tables b. Rows c. Columns d. Attributes

55. Which of the following is a group of one or more attributes that uniquely identifies a row?
a. Key b. Determinant c. Tuple d. Relation

56. For some relations, changing the data can have undesirable consequences, called:
a. Referential integrity constraints. b. Modification anomalies
c. Normal forms d. Normal forms

57. When the values in one attribute must exist in another attribute, it is called:
a Transitive dependency b. Insertion anomaly
c. Referential integrity constraints d. Normal forms

58. The different classes of relations and the techniques for preventing anomalies are called:
a. Normal forms b. Referential integrity constraints
c. Modification anomalies d. None of the above

59. Two or more attributes or attribute collections that can be a key are called
a. Candidate keys b. Determinants c. Primary keys d. BCNF

60. A relation is in this form if it is in BCNF and has no multi-value dependencies
a. Second b. Third c. Fourth d. Domain/key normal form.

61. If attribute A determines attribute B and B determines A, the values of the attributes have this relationship:
a. One-to-one b. Many-to-one c. Normalized d. Many-to-many

62. If attribute A determines B but B does not determine A, the relationship among their data values is:
a. One-to-one relationship b. Many-to-one relationship
c. Normalized relationship d. Many-to-many relationship

63. One solution to the multi-value dependency constraint problem is to:
a. Split relation in two relations, each with a single theme b. Change the theme
c. Create a new relation d. Add a composite key

64. In ___ normal form, any multi-valued attributes have been removed:
a. First b Second c. Fourth d. Fifth

65. A relation is in fourth normal form if it is in BCNF and it has no:
a. Deletion dependencies. b. Transitive dependencies.
c. Multi-value dependencies d. Partial dependencies.

66. A partial functional dependency (FD) means that:
a. Some attributes of an entity are not known
b. Not all attributes on right-hand side of FD are necessary
c. No dependency exists in the entity
d. Not all of the attributes on the left-hand side of the FD are necessary

67. The anomalies addressed by moving from BCNF to 4NF generally deal with:
a. Excessive updates and redundancy of data for each entity
b. Inability to uniquely identify an entity
c. Inability to reconstruct relations once they have been decomposed.
d. Creation of identical rows in a relation

68. In general, a row in a relation should have all of the data about
a. One instance of the relation's theme b. Each instance of the relation's theme
c. Every instance of the relation's theme d. No instance of the relation's theme

69. Which of the following is a requirement of 3NF?
a. Must contain a partial dependency. b. Must contain a composite.
c. Must contain no transitive dependencies Must contain no partial dependencies

70. Which is TRUE from functional dependency shown as A → (X, Y)?
a. X is functionally dependent on A b. A determines Y
c. A is a determinant d. X and Y are functionally dependent on A
e. All of the above

71. Which of the following should not be placed in a relational table?
a. Entity b. Attribute c. Relationship d. Repeating group

72. In BCNF, every ____ in a table is a candidate key.
a. determinant b. entity c. primary key d. atomic attribute

73. An index:
a. Makes retrievals faster
b. Is always generated for a primary key
c. Increases the space needed for the database
d. Both a and b

74. A 3NF violation means:
a. An attribute in the table can have several values in one record
b. An attribute in the table depends on only one part of a concatenated key
c. An attribute in the table belongs in another entity
d. The table has no primary key

75. Normalization:
a. Identifies incorrect referential integrity
b. Identifies incorrect placement of attributes
c. Identifies incorrect cardinality
d. Identifies incorrect foreign keys

76. Normalized databases
a. Are designed for efficient update
b. Remove key/foreign key redundancy
c. Remove data redundancy
d. Both a and c

77. A relation is not allowed to have:
a. Two columns with the same values
b. Two columns with the same domain
c. Two columns with the same names
d. None of these

Answers

1. c	2. b	3. a	4. d	5. c	6. a
7. a	8. d	9. c	10. d	11. a	12. b
13. a	14. d	15. a	16. d	17. c	18. a
19. c	20. d	21. a	22. c	23. c	24. b
25. d	26. b	27. a	28. b	29. a	30. a
31. a	32. c	33. c	34. b	35. a	36. b
37. b	38. d	39. c	40. d	41. d	42. d
43. d	44. d	45. c	46. c	47. d	48. b
49. a	50. d	51. c	52. d	53. c	54. d
55. a	56. b	57. c	58. a	59. a	60. c
61. a	62. b	63. a	64. a	65. c	66. d
67. a	68. a	69. c	70. e	71. d	72. a
73. d	74. c	75. b	76. d	77. c	

True / False

1. A relation is a three-dimensional table
2. A characteristic of a relation is that the cells of the relation hold a single value.
3. A characteristic of a relation is that the rows of a relation may hold identical values
4. The columns of a relation are sometimes called "tuples."
5. Keys are always unique
6. A tuple is a group of one or more columns that uniquely identifies a row
7. A row can be uniquely identified by a key
8. A key can be composed of a group of attributes taken together
9. It is possible to have a relation that does not have a key
10. Attribute Y is functionally dependent on attribute X if value of X determines value of Y
11. The functional dependency A → B means that value of A can be determined from value of B.
12. In the functional dependency shown as A → B, B is the determinant
13. Functional dependencies can involve groups of attributes
14. A determinant of a functional dependency may or may not be unique in a relation
15. Normalization is the process of splitting a relation into two or more relations.
16. A functional dependency is a relationship between or among attributes.
17. The known or given attribute is called the determinant in a functional dependency.
18. The relationship in a functional dependency is one-to-one (1:1).
19. The selection of the attributes to use for the key is determined by the database programmers.
20. A deletion anomaly occurs when deleting one entity results in deleting facts about another entity.
21. An insertion anomaly occurs when we cannot insert some data into the database without inserting another entity first.
22. Modification anomalies do not occur in tables that meet the definition of a relation.
23. A table of data that meets the minimum definition of a relation is automatically in first normal form.
24. A relation is in first normal form if all of its non-key attributes are dependent on part of key.
25. A relation is in second normal form if all of its non-key attributes are dependent on entire key.
26. A relation can be in third normal form without being in second normal form.
27. Fifth normal form is the highest normal form.
28. Normalization decomposes relations to produce small, well-structured relations
29. A determinant is a constraint between two attributes or sets of attributes
30. A determinant is an attribute or set of attributes that uniquely identifies a row in a relation.
31. A determinant of a functional dependency will always be unique in a relation.
32. A functional dependency is a relationship between or among attributes.
33. Every relation has at least one key
34. Classes of relations and the techniques for preventing anomalies are called normal forms..
35. A relation is in 4th normal form if it is in 2nd normal form and has no transitive dependencies.
36. A relation is in BCNF if every determinant is a candidate key.
37. A key is a unique identifier of an attribute
38. If two attributes functionally determine each other, the relationship of their data values is one to many.
39. If two attributes have a one-to-one relationship, they functionally determine each other
40. If A determines B and B does not determine A, the relationship among their values is many-to-many.
41. If attribute A determines B but B does not determine A, the relationship among their data values is many to one
42. Normalized relations avoid modification anomalies.
43. Anomalies can be removed by splitting the relation into two or more relations
44. When a key of one relation is stored in a second relation, it is called a foreign key
45. A null value is an attribute value that has never been supplied.

46. If A → B and B → A, then A and B have a many-to-many attribute relationship.
47. Fifth normal form deals with obscure problems with transitive dependencies
48. A relation is in 4NF if it is in 3NF and contains no multi-value dependencies.
49. A relation can have only one candidate key
50. A relation is in 3NF if it is in 2NF and contains no transitive dependencies
51. A transitive dependency exists when a non-key attribute is determined by only part of the key.
52. Relations that have a composite key and are in 1NF are automatically in 2NF.
53. Any table that meets the definition of a relation is in 2NF
54. Relations are classified into "normal forms" based on the types of modification anomalies that they are vulnerable to
55. Breaking a relation into two or more relations may create the need for a referential integrity constraint to be defined
56. If a table meets minimum definition of a relation, it has an effective or appropriate structure.
57. Functional dependencies can involve groups of attributes
58. It is possible to have a relation that does not have a key.
59. Indexes makes the searching of a record faster
60. One property of a relation is that each attribute within a relation has a unique name.
61. In 2NF, every non-key attribute must depend on the key attribute.
62. Partial dependencies are removed in 3NF.
63. The database is normalized to avoid certain database anomalies.
64. A database anomaly leads the database on to an inconsistent state.
65. Relational models view the data as part of a table or collection of tables in which all key values must be identified.
66. Normalization produces a lower normal form.

Answers

1. T	2. T	3. F	4. F
5. F	6. F	7. T	8. T
9. F	10. T	11. F	12. F
13. T	14. T	15. T	16. T
17. T	18. F	19. F	20. T
21. T	22. F	23. T	24. F
25. T	26. F	27. F	28. T
29. F	30. F	31. F	32. T
33. T	34. T	35. F	36. T
37. F	38. F	39. T	40. F
41. T	42. T	43. T	44. T
45. T	46. F	47. F	48. T
49. F	50. T	51. F	52. F
53. F	54. T	55. T	56. F
57. T	58. F	59. T	60. T
61. T	62. F	63. T	64. T
65. T	66. F		