Normalization

UNIT-IV Syllabus

Schema Refinement (Normalization): Purpose of Normalization, concept of functional dependency, Closure of functional dependency and attribute closure, Normal forms based on functional dependency (1NF, 2NF and 3 NF), concept of surrogate key, Boyce-Codd normal form (BCNF), Lossless join and dependency preserving decomposition, Fourth normal form (4NF), Fifth Normal Form (5NF).

Normalization

A large database defined as a single relation may result in data duplication. This repetition of data may result in:

- Making relations very large.
- It isn't easy to maintain and update data as it would involve searching many records in relation.
- Wastage and poor utilization of disk space and resources.
- The likelihood of errors and inconsistencies increases.

What is Normalization?

- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.
- Normalization divides the larger table into smaller and links them using relationships.
- The normal form is used to reduce redundancy from the database table.

Why do we need Normalization?

- The main reason for normalizing the relations is removing anomalies.
- Failure to eliminate anomalies leads to data redundancy and can cause data integrity and other problems as the database grows.
- Normalization consists of a series of guidelines that helps to guide you in creating a good database structure.

Data modification anomalies can be categorized into three types:

- **Insertion Anomaly:** Insertion Anomaly refers to when one cannot insert a new tuple into a relationship due to lack of data.

- **Deletion Anomaly:** The delete anomaly refers to the situation where the deletion of data results in the unintended loss of some other important data.

- **Updation Anomaly:** The update anomaly is when an update of a single data value requires multiple rows of data to be updated.

Functionally Dependency

- In relational database management, functional dependency is a concept that specifies the relationship between two sets of attributes where one attribute determines the value of another attribute.

- It is denoted as **X** → **Y**, where the attribute set on the left side of the arrow, **X** is called **Determinant**, and **Y** is called the **Dependent.**

Functionally Dependency

What is Functional Dependency?

A functional dependency occurs when one attribute uniquely determines another attribute within a relation. It is a constraint that describes how attributes in a table relate to each other. If attribute A functionally determines attribute B we write this as the **A→B**.

Functional dependencies are used to mathematically express relations among database entities and are very important to understanding concepts in Relational Database Systems.

- Inference Rule (IR):
- The Armstrong's axioms are the basic inference rule.
- Armstrong's axioms are used to conclude functional dependencies on a relational database.
- The inference rule is a type of assertion. It can apply to a set of FD(functional dependency) to derive other FD.
- Using the inference rule, we can derive additional functional dependency from the initial set.
- The Functional dependency has different types of inference rule:

1. Reflexive Rule (IR1)

In the reflexive rule, if Y is a subset of X, then X determines Y.

If $X \supseteq Y$ then $X \rightarrow Y$

Example:

1. X = {a, b, c, d, e}
2. Y = {a, b, c}

2. Augmentation Rule (IR2)

The augmentation is also called as a partial dependency.

In augmentation, if X determines Y, then XZ determines YZ for any Z.

If $X \rightarrow Y$ then $XZ \rightarrow YZ$

Example:

For R(ABCD), **if** $A \rightarrow B$ then $AC \rightarrow BC$

3. Transitive Rule (IR3)

In the transitive rule, if X determines Y and Y determine Z, then X must also determine Z.

If $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z$

4. Union Rule

Union rule says, if X determines Y and X determines Z, then X must also determine Y and Z.

If $X \rightarrow Y$ and $X \rightarrow Z$ then $X \rightarrow YZ$

Functionally Dependency

- A functional dependency is a constraint that specifies the relationship between two sets of attributes where one set can accurately determine the value of other sets.
- It is denoted as **X** $\longrightarrow$ **Y**, where X is a set of attributes that is capable of determining the value of Y.
- The attribute set on the left side of the arrow, **X** is called **Determinant**, while on the right side, **Y** is called the **Dependent**.
- Functional dependencies are used to mathematically express relations among database entities

Functionally Dependency

Example:

roll_no	name	dept_name	dept_building
42	abc	CO	A4
43	pqr	IT	A3
44	xyz	CO	A4
45	xyz	IT	A3
46	mno	EC	B2
47	jkl	ME	B2

Functionally Dependency

Armstrong's axioms/properties of functional dependencies:

1. **Reflexivity:** If Y is a subset of X, then X→Y holds by reflexivity rule
   Example, {roll_no, name} → name is valid.

2. **Augmentation:** If X → Y is a valid dependency, then XZ → YZ is also valid by the augmentation rule.

   Example, {roll_no, name} → dept_building is valid, hence {roll_no, name, dept_name} → {dept_building, dept_name} is also valid.

3. **Transitivity**: If X → Y and Y → Z are both valid dependencies, then X→Z is also valid by the Transitivity rule.

   Example, roll_no → dept_name & dept_name → dept_building, then roll_no → dept_building is also valid.

Functionally Dependency

- **From the above table, some valid functional dependencies:**
- roll_no → { name, dept_name, dept_building },→ Here, roll_no can determine values of fields name, dept_name and dept_building, hence a valid Functional dependency

- roll_no → dept_name , Since, roll_no can determine whole set of {name, dept_name, dept_building}, it can determine its subset dept_name also.

- dept_name → dept_building , Dept_name can identify the dept_building accurately, since departments with different dept_name will also have a different dept_building

- More valid functional dependencies: roll_no → name, {roll_no, name} → {dept_name, dept_building}, etc.

Functionally Dependency

Here are some invalid functional dependencies:

- name → dept_name

Students with the same name can have different dept_name, hence this is not a valid functional dependency.

- dept_building → dept_name

There can be multiple departments in the same building. Example, in the above table departments ME and EC are in the same building B2, hence dept_building → dept_name is an invalid functional dependency.

- More invalid functional dependencies: name → roll_no, {name, dept_name} → roll_no, dept_building → roll_no, etc.

Types of Functional dependencies in DBMS

- Trivial functional dependency
- Non-Trivial functional dependency
- Multivalued functional dependency
- Transitive functional dependency

Trivial Functional Dependency

In **Trivial Functional Dependency**, a dependent is always a subset of the determinant.

i.e. If **X** → **Y** and **Y is the subset of X**, then it is called trivial functional dependency

For example,

roll_no	name	age
42	abc	17
43	pqr	18
44	xyz	18

Here, **{roll_no, name}** → **name** is a trivial functional dependency, since the dependent **name** is a subset of determinant set **{roll_no, name}**

Non Trivial Functional Dependency

In **Non-trivial functional dependency**, the dependent is strictly not a subset of the determinant.

i.e. If **X** → **Y** and **Y is not a subset of X**, then it is called Non-trivial functional dependency.

roll_no	name	age
42	abc	17
43	pqr	18
44	xyz	18

Here, **roll_no** → **name** is a non-trivial functional dependency, since the dependent **name** is **not a subset of** determinant **roll_no**

Similarly, **{roll_no, name}** → **age** is also a non-trivial functional dependency, since **age** is **not a subset of {roll_no, name}**

Normal Forms

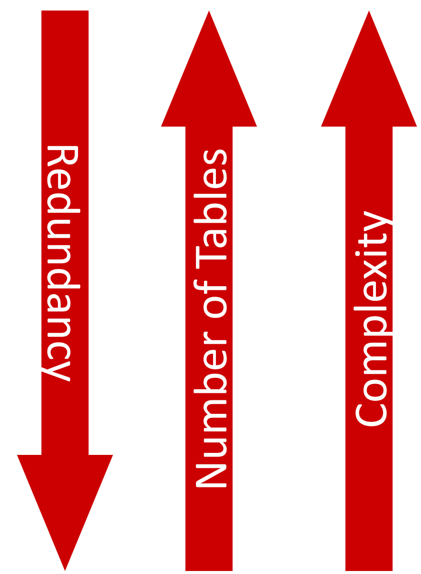
- A set of conditions on table structure that improves maintenance.

- Normalization removes processing anomalies:

  – Update
  – Inconsistent Data
  – Addition
  – Deletion

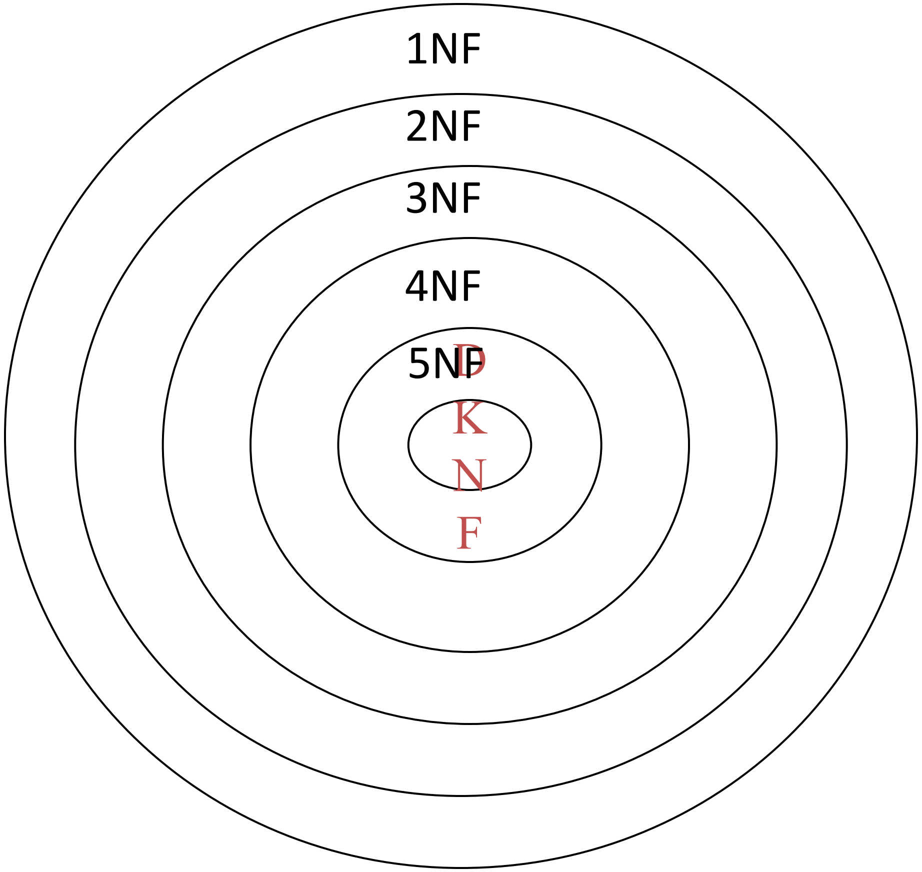
Levels of Normalization

- Levels of normalization based on the amount of redundancy in the database.
- Various levels of normalization are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)
  - Boyce-Codd Normal Form (BCNF)
  - Fourth Normal Form (4NF)
  - Fifth Normal Form (5NF)
  - Domain Key Normal Form (DKNF)



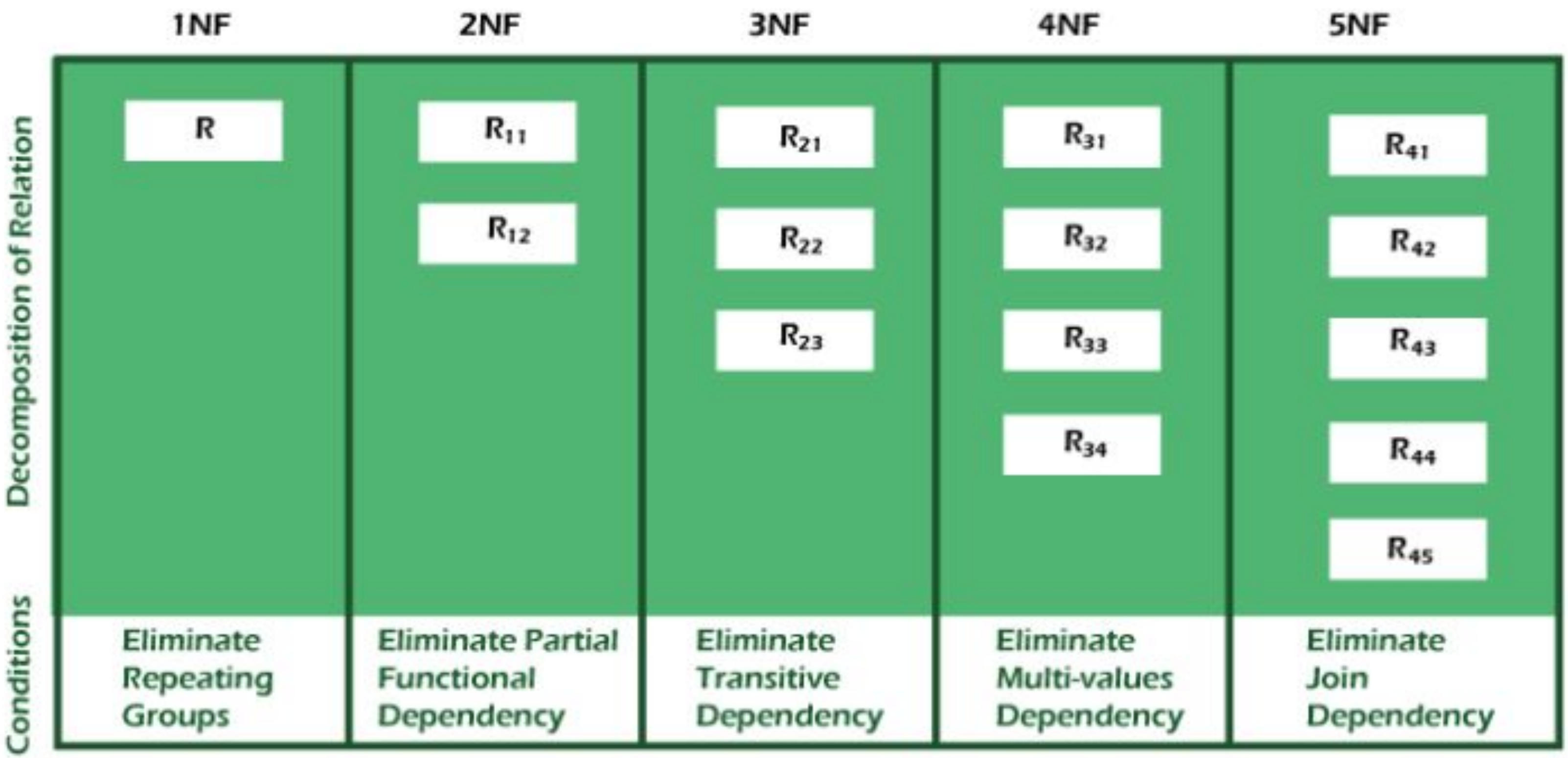
Most databases should be 3NF or BCNF in order to avoid the database anomalies.

Levels of Normalization



Each higher level is a subset of the lower level

Decomposition



Normal Forms

All attributes depend on the key, the whole key and nothing but the key.

1NF Keys and no repeating groups

2NF No partial dependencies

3NF No Transitive dependencies

BCNF All determinants are candidate keys

4NF No multi valued dependencies

5NF No Join dependencies

1st Normal Form

- Table has a primary key
- Table has no repeating groups

A multivalued attribute is an attribute that may have several values for one record

A repeating group is a set of one or more multivalued attributes that are related

1st Normal Form

- If a relation contain composite or multi-valued attribute, it violates first normal form or a relation is in first normal form if it does not contain any composite or multi-valued attribute.
- A relation is in first normal form if every attribute in that relation is **singled valued attribute**.
- A relation R is in 1NF if and only if every tuple contains exactly one for each attribute value
- Or
- First Normal Form : A relation is in 1NF if it contains an atomic value.

Relation STUDENT in table 1 is not in 1NF because of multi-valued attribute STUD_PHONE. Its decomposition into 1NF has been shown in table 2

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721, 9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 1

↓ Conversion to first normal form

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721	HARYANA	INDIA
1	RAM	9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 2

In the below table Course is a multi-valued attribute so it is not in 1NF.

In second Table is in 1NF as there is no multi-valued attribute

ID	Name	Courses
1	A	c1, c2
2	E	c3
3	M	C2, c3

ID	Name	Course
1	A	c1
1	A	c2
2	E	c3
3	M	c2
3	M	c3

Un Normalized Data

S#	STATUS	CITY	P#	QTY
S1	20	London	P1 P2 P3 P4 P5 P6	300 200 400 200 100 100
S2	10	Paris	P1 P2	300 400
S3	10	PariS	P2	200
S4	20	London	P2 P4 P5	200 300 400

Un Normalized Data

S#	STATUS	CITY	P#	QTY
S1	20	London	P1 P2 P3 P4 P5 P6	300 200 400 200 100 100
S2	10	Paris	P1 P2	300 400
S3	10	PariS	P2	200
S4	20	London	P2 P4 P5	200 300 400

1NF

S#	STATUS	CITY	P#	QTY
S1	20	London	P1	300
S1	20	London	P2	200
S1	20	London	P3	400
S1	20	London	P4	200
S1	20	London	P5	100
S1	20	London	P6	100
S2	10	Paris	P1	300
S2	10	Paris	P2	400
S3	10	Paris	P2	200
S4	20	London	P2	200
S4	20	London	P4	300
S4	20	London	P5	400

2nd Normal Forms

Second Normal Form : 2NF

A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.

To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency.

2nd Normal Forms

STUD_NO	COURSE_NO	COURSE_FEE
1	C1	1000
2	C2	1500
1	C4	2000
4	C3	1000
4	C1	1000
2	C5	2000

2nd Normal Forms

COURSE_FEE would be a non-prime attribute, as it does not belong to the one only candidate key {STUD_NO, COURSE_NO} ;

But, COURSE_NO -> COURSE_FEE, i.e., COURSE_FEE is dependent on COURSE_NO

Non-prime attribute COURSE_FEE is dependent on a proper subset of the candidate key, which is a partial dependency and so this relation is not in 2NF.

2NF

To convert the above relation to 2NF,

split the table into two tables such as :

Table 1: STUD_NO, COURSE_NO

Table 2: COURSE_NO, COURSE_FEE

2NF

To convert the above relation to 2NF,
split the table into two tables such as :
Table 1: STUD_NO, COURSE_NO
Table 2: COURSE_NO, COURSE_FEE

Table 1

STUD_NO	COURSE_NO
1	C1
2	C2
1	C4
4	C3
4	C1
2	C5

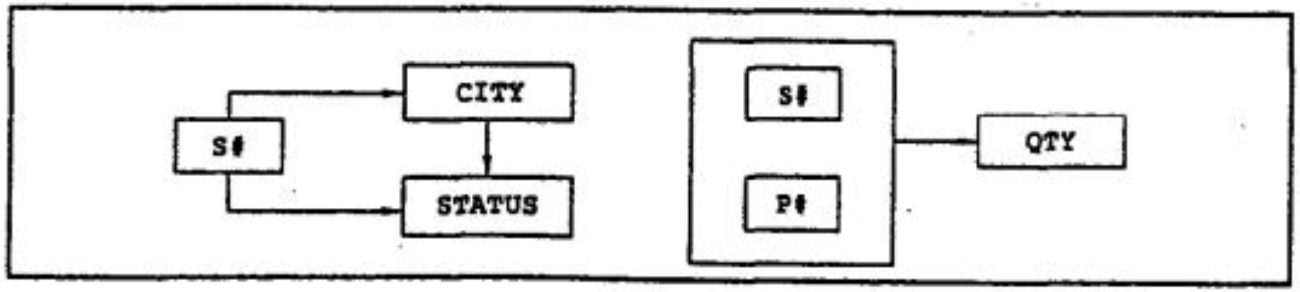
Table 2

COURSE_NO	COURSE_FEE
C1	1000
C2	1500
C3	1000
C4	2000
C5	2000

1NF

S#	STATUS	CITY	P#	QTY
S1	20	London	P1	300
S1	20	London	P2	200
S1	20	London	P3	400
S1	20	London	P4	200
S1	20	London	P5	100
S1	20	London	P6	100
S2	10	Paris	P1	300
S2	10	Paris	P2	400
S3	10	Paris	P2	200
S4	20	London	P2	200
S4	20	London	P4	300
S4	20	London	P5	400

2NF



SECOND

S#	STATUS	CITY
S1	20	London
S2	10	Paris
S3	10	Paris
S4	20	London
S5	30	Athens

No counterpart in Fig. 12.6

SP

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

3rd Normal Forms

Third Normal Form :

A relation will be in 3NF if it is in 2NF and no transitive dependency exists.

Transitive Functional Dependency

- In transitive functional dependency, dependent is indirectly dependent on determinant.

  i.e. If **a** $\rightarrow$ **b** & **b** $\rightarrow$ **c**, then according to axiom of transitivity, **a** $\rightarrow$ **c**. This is a **transitive functional dependency**

Transitive Dependency

enrol_no	name	dept	building_no
42	abc	CO	4
43	pqr	EC	2
44	xyz	IT	1
45	abc	EC	2

Here, **enrol_no** → **dept** and **dept** → **building_no**,

Hence, according to the axiom of transitivity, **enrol_no** → **building_no** is a valid functional dependency.

This is an indirect functional dependency, hence called Transitive functional dependency.

Transitive Dependency

enrol_no	dept	name
42	CO	abc
43	EC	pqr
44	IT	xyz
45	EC	abc

dept	Building_no
CO	4
EC	2
IT	1

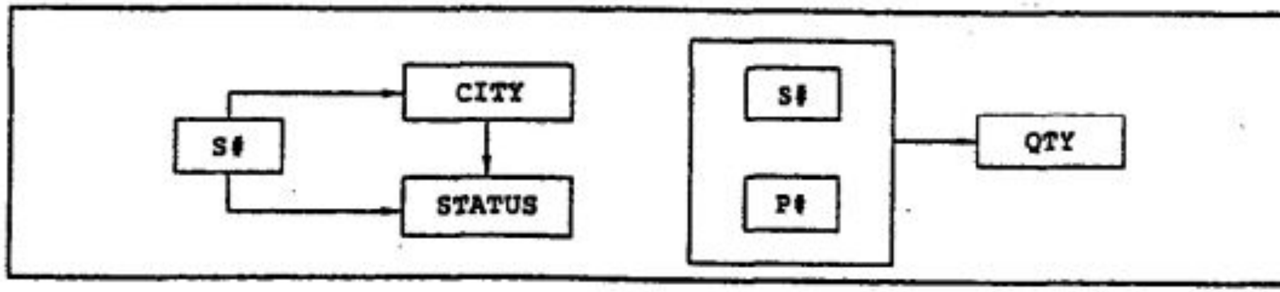
enrol_no	Building_no
42	4
43	2
44	1
45	2

Here, **enrol_no** → **dept** and **dept** → **building_no**,

Hence, according to the axiom of transitivity, **enrol_no** → **building_no** is a valid functional dependency.

This is an indirect functional dependency, hence called Transitive functional dependency.

2NF



SECOND

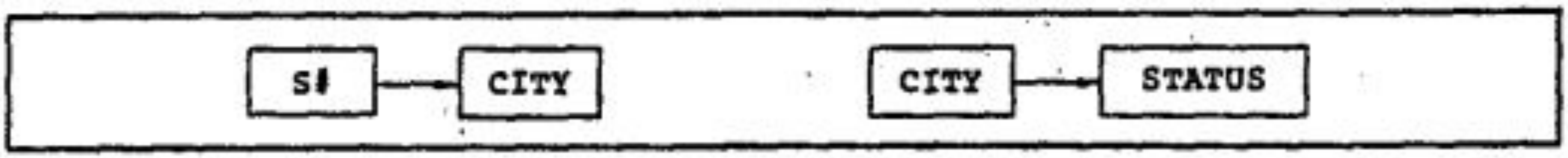
S#	STATUS	CITY
S1	20	London
S2	10	Paris
S3	10	Paris
S4	20	London
S5	30	Athens

No counterpart in Fig. 12.6

SP

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

3NF



SC

S#	CITY
S1	London
S2	Paris
S3	Paris
S4	London
S5	Athens

CS

CITY	STATUS
Athens	30
London	20
Paris	10
Rome	50

S#	STATUS

Types of Normal Forms

First Normal Form : A relation is in 1NF if it contains an atomic value.

Second Normal Form : A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.

Third Normal Form : A relation will be in 3NF if it is in 2NF and no transitive dependency exists.

Types of Normal Forms

Boyce Codd’s Normal Form (BCNF): A stronger definition of 3NF is known as Boyce Codd's normal form. A relation is in BCNF if it is in 3NF and every determinant is a candidate key

Fourth Normal Form : A relation will be in 4NF if it is in Boyce Codd's normal form and has no multi-valued dependency.

Fifth Normal Form : A relation is in 5NF. If it is in 4NF and does not contain any join dependency, joining should be lossless.

1NF

Full Names	Physical Address	Movies rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3^{rd} Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5^{th} Avenue	Clash of the Titans	Mr.

1NF (First Normal Form) Rules

- Each table cell should contain a single value.
- Each record needs to be unique.

1NF

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

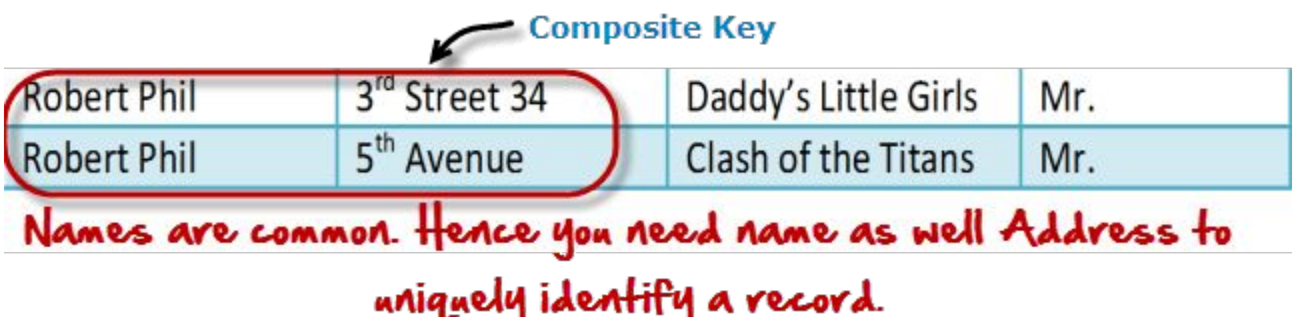
Keys Concept

What is a Primary Key?

- A primary is a single column value used to identify a database record uniquely.
- It has following attributes
- A primary key cannot be NULL
- A primary key value must be unique
- The primary key values should rarely be changed
- The primary key must be given a value when a new record is inserted.

What is Composite Key?

- A composite key is a primary key composed of multiple columns used to identify a record uniquely
- In our database, we have two people with the same name Robert Phil, but they live in different places.



2NF (Second Normal Form) Rules

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key that does not functionally dependant on any subset of candidate key relation

2NF (Second Normal Form)

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key that does not functionally dependant on any subset of candidate key relation

We have divided our 1NF table into two tables . Table 1 and Table2.
Table 1 contains member information. Table 2 contains information on movies rented.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
1	Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
1	Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
2	Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
2	Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
3	Robert Phil	5th Avenue	Clash of the Titans	Mr.

2NF (Second Normal Form)

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key that does not functionally dependant on any subset of candidate key relation

Membership ID	Full Names	Physical Address	Salutation
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

2NF

- **Database – Foreign Key**

In Table 2, Membership_ID is the Foreign Key

Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Foreign Key references the primary key of another Table. It helps connect Tables

A foreign key can have a different name from its primary key

It ensures rows in one table have corresponding rows in another

Unlike the Primary key, they do not have to be unique.

Foreign Key

Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Foreign Key references Primary Key
Foreign Key can only have values present in primary key
It could have a name other than that of Primary Key

Primary Key

Membership ID	Full names	Physical Address	Salutation
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Insert a record in Table 2 where Member ID =101

MEMBERSHIP ID	MOVIES RENTED
101	Mission Impossible

But Membership ID 101 is not present in Table 1

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Database will throw an ERROR. This helps in referential integrity

You will only be able to insert values into your foreign key that exist in the unique key in the parent table. This helps in referential integrity.

The above problem can be overcome by declaring membership id from Table2 as foreign key of membership id from Table1

Now, if somebody tries to insert a value in the membership id field that does not exist in the parent table, an error will be shown!

3NF (Third Normal Form) Rules

- Rule 1- Be in 2NF
- Rule 2- Has no transitive functional dependencies

Membership ID	Full names	Physical Address	Salutation ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

Salutation
Ms.
Mr.
Mr.

Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

We have again divided our tables and created a new table which stores Salutations.

There are no transitive functional dependencies, and hence our table is in 3NF

In Table 3 Salutation ID is primary key, and in Table 1 Salutation ID is foreign to primary key in Table 3

3NF

Membership ID	Full names	Physical Address	Salutation ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3^{rd} Street 34	1
3	Robert Phil	5^{th} Avenue	1

Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Salutation ID	Salutation
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

Normalization

The Normalization Example in the Text Book
Figure 4-24 INVOICE (Pine Valley Furniture Company)

PVFC Customer Invoice

Customer ID	2	**Order ID**	1006
Customer Name	Value Furniture	**Order Date**	10/24/2010
Address	15145 S.W. 17th St. Plano TX 75022		

Product ID	Product Description	Finish	Quantity	Unit Price	Extended Pric
7	Dining Table	Natural Ash	2	$800.00	$1,600.00
5	Writer's Desk	Cherry	2	$325.00	$650.00
4	Entertainment Center	Natural Maple	1	$650.00	$650.00
				Total	$2,900.00

Normalization

Figure 4-25 INVOICE Data
Table with multivalued attributes, not in 1st normal form

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	Product Description	Product Finish	Product StandardPrice	Ordered Quantity
1006	10/24/2010	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
					5	Writer's Desk	Cherry	325.00	2
					4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	11	4-Dr Dresser	Oak	500.00	4
					4	Entertainment Center	Natural Maple	650.00	3

Note: this is NOT a relation. WHY?

Normalization

Figure 4-26 INVOICE relation (1NF)
Table with no multivalued attributes and unique rows

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	Product Description	Product Finish	Product StandardPrice	Ordered Quantity
1006	10/24/2010	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2010	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2010	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

Note: this is relation, but not a well-structured one. WHY?

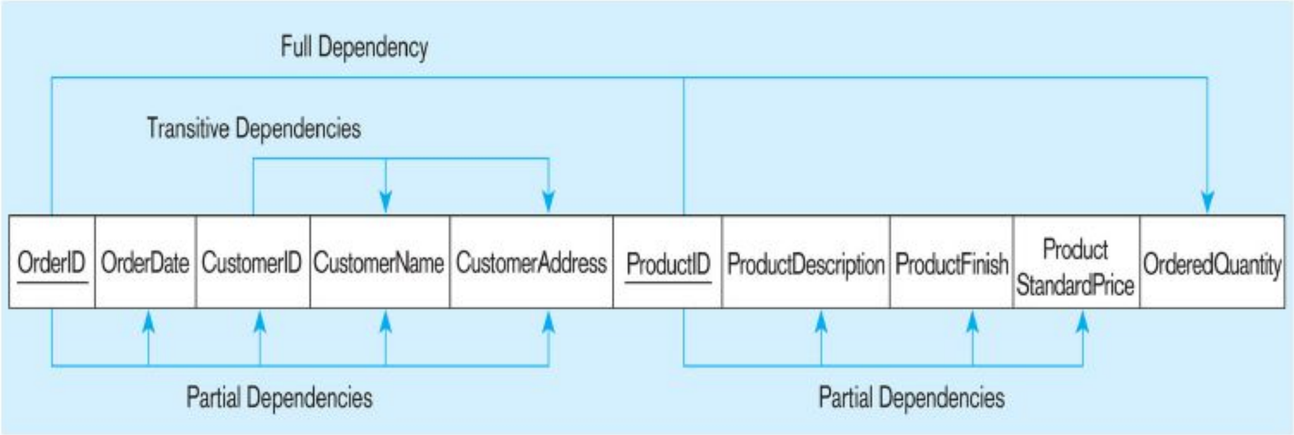
Normalization

Anomalies in this Table

- **Insertion**–if new product is ordered for order 1007 of existing customer, customer data must be re-entered, causing duplication
- **Deletion**–if we delete the Dining Table from Order 1006, we lose information concerning this item's finish and price
- **Update**–changing the price of product ID 4 requires update in several records

Normalization

Figure 4-27 Functional dependency diagram for INVOICE



Order_ID ➔ Order_Date, Customer_ID, Customer_Name, Customer_Address

Customer_ID ➔ Customer_Name, Customer_Address

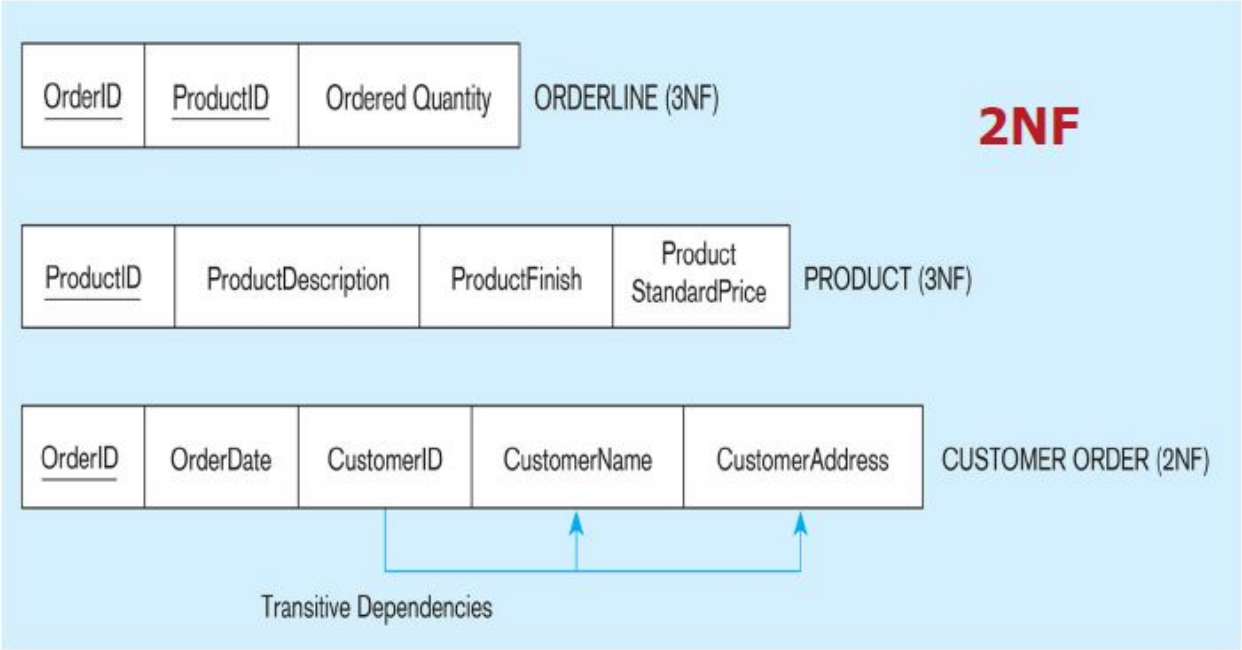
Product_ID ➔ Product_Description, Product_Finish, Unit_Price

Order_ID, Product_ID ➔ Order_Quantity

Therefore, NOT in 2nd Normal Form

Normalization

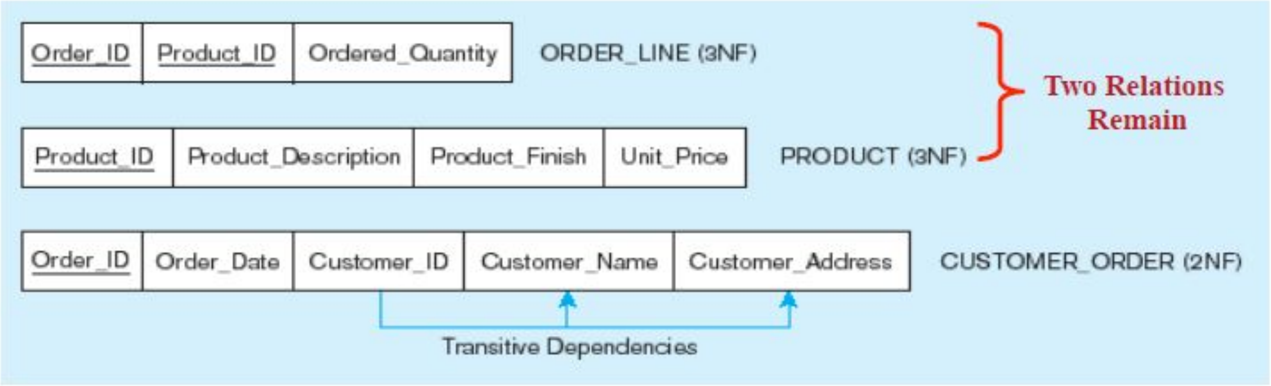
Figure 4-28 Partial Dependencies were Removed (2NF)

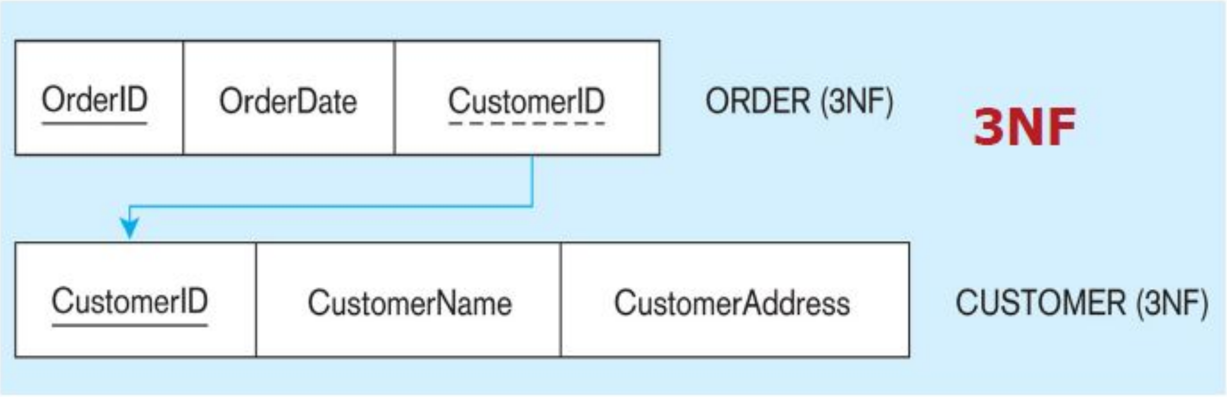


Partial dependencies are removed, but there are still transitive dependencies

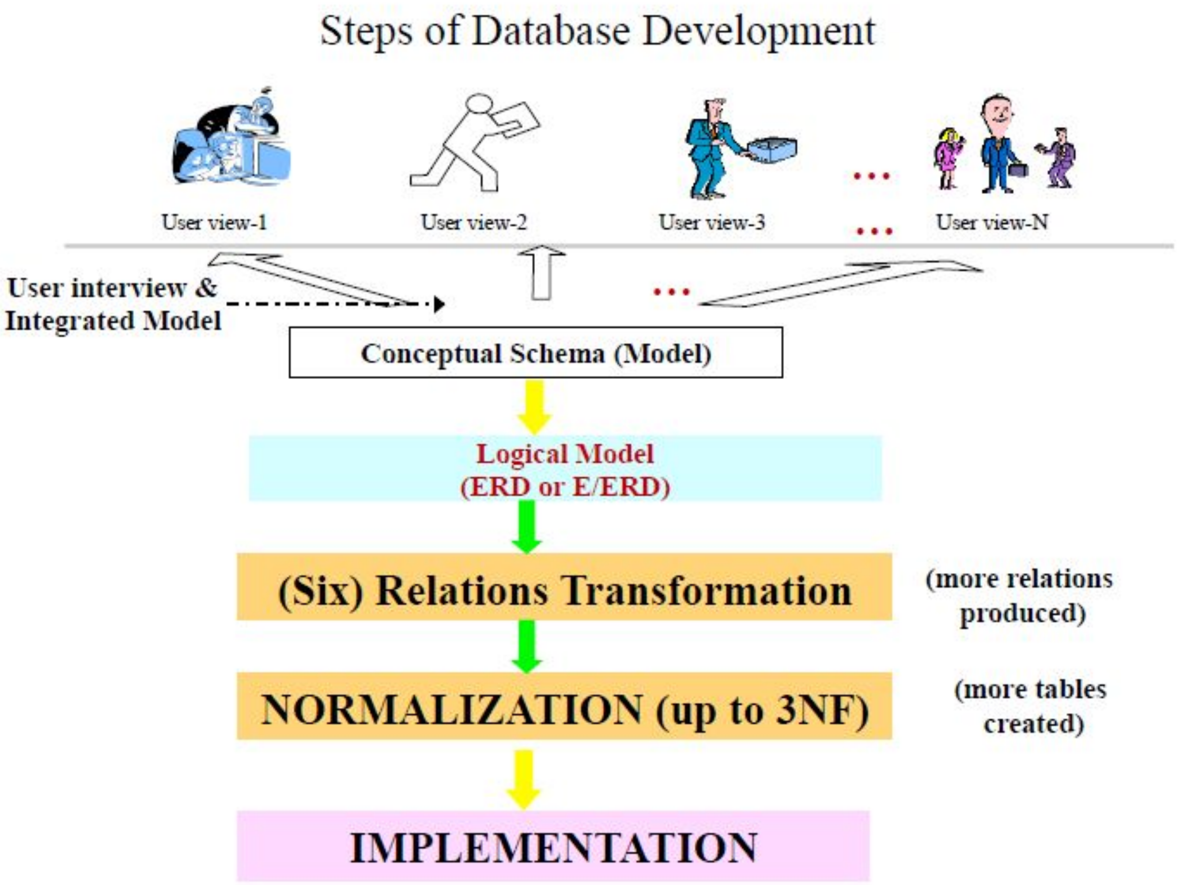
Normalization

Figure 4-29 Transitive Dependencies were Removed (3NF)





Normalization



BCNF

- A Relation R is in BCNF iff every determinant is a candidate key
- A table is in Boyce-Codd Normal Form (BCNF) when it is in 3NF and every determinant in the table is a candidate key.
- For example, if the table is in 3NF and it contains a nonprime attribute that determines a prime attribute, the BCNF requirements are not met. This description clearly yields the following conclusions:
- If a table is in 3NF and it contains only one candidate key, 3NF and BCNF are equivalent.
- BCNF can be violated only if the table contains more than one candidate key.

Boyce Codd normal form (BCNF)

- BCNF is the advance version of 3NF.
- A table is in BCNF if every functional dependency $X \longrightarrow Y$, X is the candidate key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is candidate key.

BCNF

Example: Let's assume there is a company where employees work in more than one department.

EMPLOYEE table:

EMP_ID	EMP_COUNTRY	EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

BCNF

EMP_DEPT_MAPPING table:

EMP_ID	EMP_DEPT
D394	283
D394	300
D283	232
D283	549

Functional dependencies:

EMP_ID → EMP_COUNTRY
EMP_DEPT → {DEPT_TYPE, EMP_DEPT_NO}

Candidate keys:
For the first table: EMP_ID
For the second table: EMP_DEPT
For the third table: {EMP_ID, EMP_DEPT}

Now, this is in BCNF because left side part of both the functional dependencies is a candidate key.

BCNF

- **In the above table Functional dependencies are as follows:**

EMP_ID $\rightarrow$ EMP_COUNTRY

EMP_DEPT $\rightarrow$ {DEPT_TYPE, EMP_DEPT_NO}

- **Candidate key: {EMP-ID, EMP-DEPT}**

BCNF

- The table is not in BCNF because neither EMP_DEPT nor EMP_ID alone are keys.
- To convert the given table into BCNF, we decompose it into three tables:

EMP_COUNTRY table:

EMP_ID	EMP_COUNTRY
264	India
264	India

EMP_DEPT table:

EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

4NF

- A **multivalued dependency** always requires at least three attributes because it consists of at least two attributes that are dependent on a third.

- For a dependency A -> B, if for a single value of A, multiple value of B exists, then the table may have multi-valued dependency.

- The table should have at least 3 attributes and B and C should be independent for A ->> B multivalued dependency.

4NF

- **Properties –** A relation R is in 4NF if and only if the following conditions are satisfied:

- It should be in the Boyce-Codd Normal Form (BCNF).

- the table should not have any Multi-valued Dependency.

4NF

SID	SNAME	CID	CNAME
S1	A	C1	C
S1	A	C2	D
S2	B	C1	C
S2	B	C2	D

Multivalued dependencies (MVD) are:

SID->->CID; SID->->CNAME; SNAME->->CNAME

4NF

COURSE	TEACHERS	TEXTS
Physics	**TEACHER** Prof. Green Prof. Brown	**TEXT** *Basic Mechanics* *Principles of Optics*
Math	**TEACHER** Prof. Green	**TEXT** *Basic Mechanics* *Vector Analysis* *Trigonometry*

CTX

COURSE	TEACHER	TEXT
Physics	Prof. Green	*Basic Mechanics*
Physics	Prof. Green	*Principles of Optics*
Physics	Prof. Brown	*Basic Mechanics*
Physics	Prof. Brown	*Principles of Optics*
Math	Prof. Green	*Basic Mechanics*
Math	Prof. Green	*Vector Analysis*
Math	Prof. Green	*Trigonometry*

4NF

CT

COURSE	TEACHER
Physics	Prof. Green
Physics	Prof. Brown
Math	Prof. Green

CX

COURSE	TEXT
Physics	*Basic Mechanics*
Physics	*Principles of Optics*
Math	*Basic Mechanics*
Math	*Vector Analysis*
Math	*Trigonometry*

4NF

STUDENT

STU_ID	COURSE	HOBBY
21	Computer	Dancing
21	Math	Singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey

The given STUDENT table is in 3NF, but the COURSE and HOBBY are two independent entity. Hence, there is no relationship between COURSE and HOBBY.

In the STUDENT relation, a student with STU_ID, **21** contains two courses, **Computer** and **Math** and two hobbies, **Dancing** and **Singing**

So there is a Multi-valued dependency on STU_ID, which leads to unnecessary repetition of data.

So to make the above table into 4NF, we can decompose it into two tables:

4NF

STUDENT_COURSE

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics

STUDENT_HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

4NF

Assume that

- Each subject is taught by many Instructors
- The same books are used in many subjects
- Each Instructor uses a different book

Course, Instructor → Text

Course, Text → Instructor

Course	Instructor	Text
CS 121	Dr. A. James	Int to Com Science
CS 121	Dr. P. Hold	Comp Scien Int

Course	Instructor	Text
CS 141	Dr. T. Watson	Int to Com Science
CS 141	Dr. P. Hold	Comp Scien Int
CS 101	Dr. M. Jones	COMP SCIEN

4NF: Decompose into (Course, Instructor) and (Course, Text)

5NF

- A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.

- 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy.

- 5NF is also known as Project-join normal form (PJ/NF).

5NF

SUBJECT	LECTURER	SEMESTER
Computer	Anshika	Semester 1
Computer	John	Semester 1
Math	John	Semester 1
Math	Akash	Semester 2
Chemistry	Praveen	Semester 1

In the above table, John takes both Computer and Math class for Semester 1 but he doesn't take Math class for Semester 2. In this case, combination of all these fields required to identify a valid data.

Suppose we add a new Semester as Semester 3 but do not know about the subject and who will be taking that subject so we leave Lecturer and Subject as NULL.

But all three columns together acts as a primary key, so we can't leave other two columns blank.

5NF

- So to make the above table into 5NF, we can decompose it into three relations P1, P2 & P3:

P1

SEMESTER	SUBJECT
Semester 1	Computer
Semester 1	Math
Semester 1	Chemistry
Semester 2	Math

P2

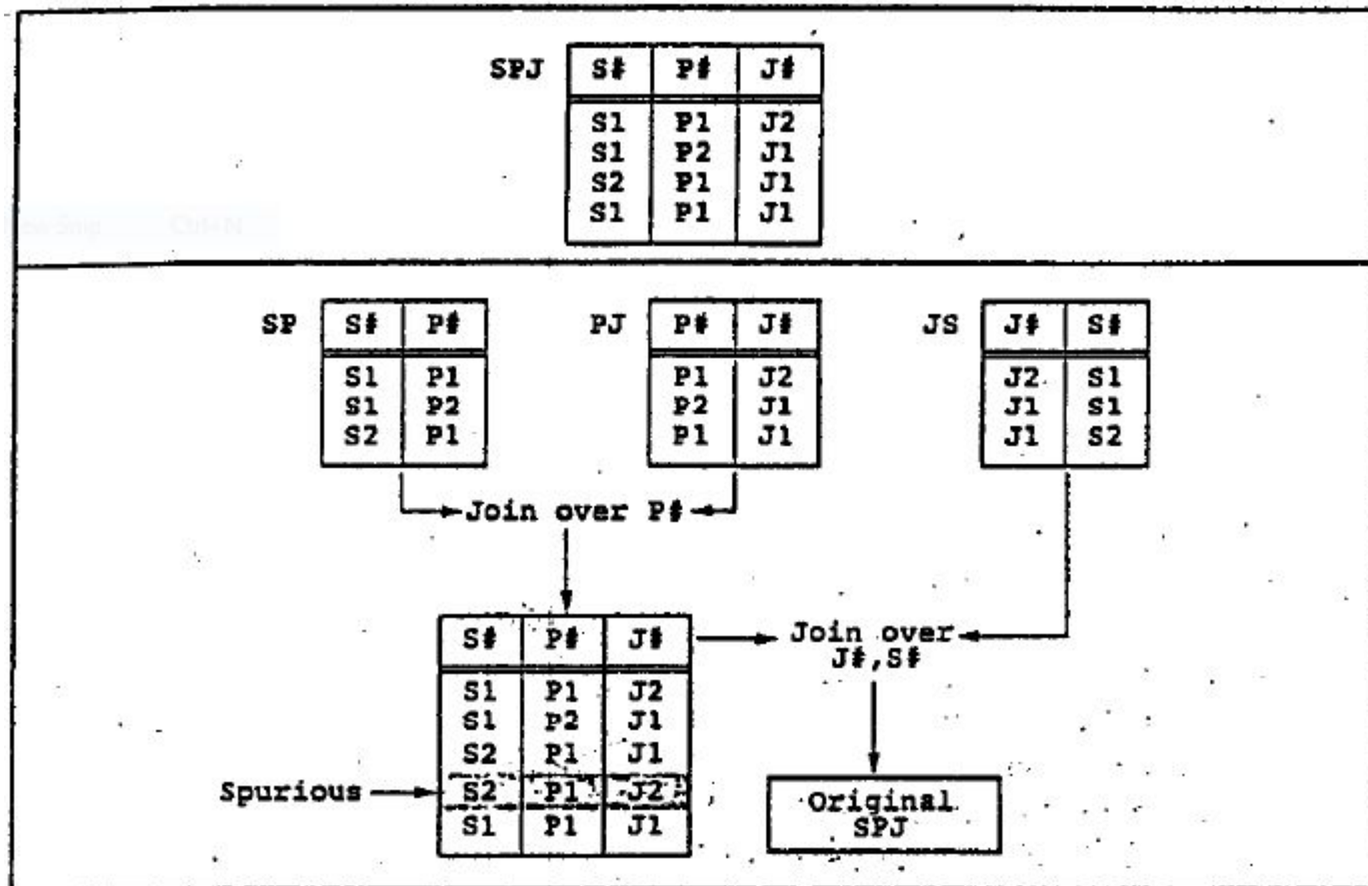
SUBJECT	LECTURER
Computer	Anshika
Computer	John
Math	John
Math	Akash
Chemistry	Praveen

5NF

P3

SEMSTER	LECTURER
Semester 1	Anshika
Semester 1	John
Semester 1	John
Semester 2	Akash
Semester 1	Praveen

5NF


SPJ
S# P# J#
S1 P1 J2
S1 P2 J1
S2 P1 J1
S1 P1 J1
SP
S# P#
S1 P1
S1 P2
S2 P1
PJ
P# J#
P1 J2
P2 J1
P1 J1
JS
J# S#
J2 S1
J1 S1
J1 S2
Join over P#
S# P# J#
S1 P1 J2
S1 P2 J1
S2 P1 J1
S2 P1 J2
S1 P1 J1
Spurious
Join over J#,S#
Original SPJ

Types of Normal Forms

First Normal Form : A relation is in 1NF if it contains an atomic value.

Second Normal Form : A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.

Third Normal Form : A relation will be in 3NF if it is in 2NF and no transitive dependency exists.

Boyce Codd normal form (BCNF)

- BCNF is the advance version of 3NF.
- A table is in BCNF if every functional dependency $X \longrightarrow Y$, X is the candidate key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is candidate key.

BCNF

Boyce Codd normal form (BCNF)

It is an advance version of 3NF that's why it is also referred as 3.5 NF. BCNF is stricter than 3NF. A table complies with BCNF if it is in 3NF and for every **functional dependency** X->Y, X should be the super key of the table.

Stu_ID	Stu_Branch	Stu_Course	Branch_Number	Stu_Course_No
101	Computer Science & Engineering	DBMS	B_001	201
101	Computer Science & Engineering	Computer Networks	B_001	202
102	Electronics & Communication Engineering	VLSI Technology	B_003	401
102	Electronics & Communication Engineering	Mobile Communication	B_003	402

BCNF

Why this Table is Not in BCNF?

- The table present above is not in BCNF, because as we can see that neither Stu_ID nor Stu_Course is a Super Key.

- As the rules mentioned above clearly tell that for a table to be in BCNF, it must follow the property that for functional dependency X–>Y, X must be in Super Key and here this property fails, that's why this table is not in BCNF.

Functional Dependency of the above is as mentioned:

```
Stu_ID -> Stu_Branch Stu_Course -> {Branch_Number, Stu_Course_No}
```

Candidate Keys of the above table are: **{ S tu_ID , S tu_Course }**

Stu_ID	Stu_Branch	Stu_Course	Branch_Number	Stu_Course_No
101	Computer Science & Engineering	DBMS	B_001	201
101	Computer Science & Engineering	Computer Networks	B_001	202
102	Electronics & Communication Engineering	VLSI Technology	B_003	401
102	Electronics & Communication Engineering	Mobile Communication	B_003	402

Now, this is in BCNF because left side part of both the functional dependencies is a candidate key.

BCNF

Stu_ID	Stu_Branch	Stu_Course	Branch_Number	Stu_Course_No
101	Computer Science & Engineering	DBMS	B_001	201
101	Computer Science & Engineering	Computer Networks	B_001	202
102	Electronics & Communication Engineering	VLSI Technology	B_003	401
102	Electronics & Communication Engineering	Mobile Communication	B_003	402

- **How to Satisfy BCNF?**
- For satisfying this table in BCNF, we have to decompose it into further tables. Here is the full procedure through which we transform this table into BCNF. Let us first divide this main table into two tables **Stu_Branch** and **Stu_Course** Table.

Stu_Branch Table

Stu_ID	Stu_Branch
101	Computer Science & Engineering
102	Electronics & Communication Engineering

Candidate Key for this table: **Stu_ID**.

Stu_Course Table

Stu_Course	Branch_Number	Stu_Course_No
DBMS	B_001	201
Computer Networks	B_001	202
VLSI Technology	B_003	401
Mobile Communication	B_003	402

Candidate Key for this table: **Stu_Course**.

BCNF

Stu_ID to Stu_Course_No Table

Stu_ID	Stu_Course_No
101	201
101	202
102	401
102	402

Candidate Key for this table: **{Stu_ID, Stu_Course_No}.**

BCNF Example

EMPLOYEE table:

EMP_ID	EMP_COUNTRY	EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

In the above table Functional dependencies are as follows:

1. EMP_ID → EMP_COUNTRY
2. EMP_DEPT → {DEPT_TYPE, EMP_DEPT_NO}

Candidate key: {EMP-ID, EMP-DEPT}

BCNF Example

EMP_COUNTRY table:

EMP_ID	EMP_COUNTRY
264	India
264	India

EMP_DEPT table:

EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

EMP_DEPT_MAPPING table:

EMP_ID	EMP_DEPT
D394	283
D394	300
D283	232
D283	549

BCNF Example

Functional Dependencies :

1. EMP_ID $\rightarrow$ EMP_COUNTRY
2. EMP_DEPT $\rightarrow$ {DEPT_TYPE, EMP_DEPT_NO}

Candidate keys:

For the first table: EMP_ID
For the second table: EMP_DEPT
For the third table: {EMP_ID, EMP_DEPT}

Now, this is in BCNF because left side part of both the functional dependencies is a key.

4NF

- A **multivalued dependency** always requires at least three attributes because it consists of at least two attributes that are dependent on a third.
- For a dependency A -> B, if for a single value of A, multiple value of B exists, then the table may have multi-valued dependency.
- The table should have at least 3 attributes and B and C should be independent for A ->> B multivalued dependency.

4NF

- **Properties –** A relation R is in 4NF if and only if the following conditions are satisfied:

- It should be in the Boyce-Codd Normal Form (BCNF).

- the table should not have any Multi-valued Dependency.

4NF

SID	SNAME	CID	CNAME
S1	A	C1	C
S1	A	C2	D
S2	B	C1	C
S2	B	C2	D

Multivalued dependencies (MVD) are:

SID->->CID; SID->->CNAME; SNAME->->CNAME

4NF

COURSE	TEACHERS	TEXTS
Physics	**TEACHER** Prof. Green Prof. Brown	**TEXT** *Basic Mechanics* *Principles of Optics*
Math	**TEACHER** Prof. Green	**TEXT** *Basic Mechanics* *Vector Analysis* *Trigonometry*

CTX

COURSE	TEACHER	TEXT
Physics	Prof. Green	*Basic Mechanics*
Physics	Prof. Green	*Principles of Optics*
Physics	Prof. Brown	*Basic Mechanics*
Physics	Prof. Brown	*Principles of Optics*
Math	Prof. Green	*Basic Mechanics*
Math	Prof. Green	*Vector Analysis*
Math	Prof. Green	*Trigonometry*

4NF

CT

COURSE	TEACHER
Physics	Prof. Green
Physics	Prof. Brown
Math	Prof. Green

CX

COURSE	TEXT
Physics	*Basic Mechanics*
Physics	*Principles of Optics*
Math	*Basic Mechanics*
Math	*Vector Analysis*
Math	*Trigonometry*

4NF

STUDENT

STU_ID	COURSE	HOBBY
21	Computer	Dancing
21	Math	Singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey

The given STUDENT table is in 3NF, but the COURSE and HOBBY are two independent entity. Hence, there is no relationship between COURSE and HOBBY.

In the STUDENT relation, a student with STU_ID, **21** contains two courses, **Computer** and **Math** and two hobbies, **Dancing** and **Singing**

So there is a Multi-valued dependency on STU_ID, which leads to unnecessary repetition of data.

So to make the above table into 4NF, we can decompose it into two tables:

4NF

STUDENT_COURSE

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics

STUDENT_HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

4NF

Assume that

- Each subject is taught by many Instructors
- The same books are used in many subjects
- Each Instructor uses a different book

Course, Instructor → Text

Course, Text → Instructor

Course	Instructor	Text
CS 121	Dr. A. James	Int to Com Science
CS 121	Dr. P. Hold	Comp Scien Int

Course	Instructor	Text
CS 141	Dr. T. Watson	Int to Com Science
CS 141	Dr. P. Hold	Comp Scien Int
CS 101	Dr. M. Jones	COMP SCIEN

4NF: Decompose into (Course, Instructor) and (Course, Text)

5NF

- A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.

- 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy.

- 5NF is also known as Project-join normal form (PJ/NF).

5NF

SPJ

S#	P#	J#
S1	P1	J2
S1	P2	J1
S2	P1	J1
S1	P1	J1

SP

S#	P#
S1	P1
S1	P2
S2	P1

PJ

P#	J#
P1	J2
P2	J1
P1	J1

JS

J#	S#
J2	S1
J1	S1
J1	S2

Join over P#

S#	P#	J#
S1	P1	J2
S1	P2	J1
S2	P1	J1
S2	P1	J2
S1	P1	J1

Spurious → S2 P1 J2

Join over J#,S#

Original SPJ

5NF

SUBJECT	LECTURER	SEMESTER
Computer	Anshika	Semester 1
Computer	John	Semester 1
Math	John	Semester 1
Math	Akash	Semester 2
Chemistry	Praveen	Semester 1

In the above table, John takes both Computer and Math class for Semester 1 but he doesn't take Math class for Semester 2. In this case, combination of all these fields required to identify a valid data.

Suppose we add a new Semester as Semester 3 but do not know about the subject and who will be taking that subject so we leave Lecturer and Subject as NULL.

But all three columns together acts as a primary key, so we can't leave other two columns blank.

5NF

- So to make the above table into 5NF, we can decompose it into three relations P1, P2 & P3:

P1

SEMESTER	SUBJECT
Semester 1	Computer
Semester 1	Math
Semester 1	Chemistry
Semester 2	Math

P2

SUBJECT	LECTURER
Computer	Anshika
Computer	John
Math	John
Math	Akash
Chemistry	Praveen

5NF

P3

SEMSTER	LECTURER
Semester 1	Anshika
Semester 1	John
Semester 1	John
Semester 2	Akash
Semester 1	Praveen

UNIT-IV Model Questions

1. What is Normalization ? Explain 1NF , 2NF AND 3NF with examples

2. What is functional dependency? Explain its use in database design

3. How do you use multivalued dependencies to determine whether a table is in 4NF or not?

4. Explain with an example join dependency and fifth normal form

5. Illustrate BCNF and 3 NF or Explain 3NF & BCNF . What is the difference between them ?

6. What is Transitive dependency? Explain its use in database design

7. What is Multivalued dependency? Explain its use in database design

8. 1. What is Normalization ? Explain BCNF , 4NF and 5NF with examples

1. A ___ is normalized after it has been organized.

1. Table
2. Database
3. Row
4. Column

1. A ___ is normalized after it has been organized.

1. Table
2. Database
3. Row
4. Column

Answer: 2) Database

Explanation:
A database is normalized after it has been organized.

2. By normalizing relations or sets of relations, one minimizes ___.

1. Data
2. Fields
3. Redundancy
4. Database

2. By normalizing relations or sets of relations, one minimizes ___.

1. Data
2. Fields
3. Redundancy
4. Database

Answer: 3) Redundancy

Explanation:

By normalizing relations or sets of relations, one minimizes redundancy.

3. In addition to removing undesirable characteristics, normalization also eliminates ___ anomalies.

1. Insert
2. Update
3. Delete
4. All of the above

3. In addition to removing undesirable characteristics, normalization also eliminates ___ anomalies.

1. Insert
2. Update
3. Delete
4. All of the above

Answer: 4) All of the above

Explanation:
In addition to removing undesirable characteristics, normalization also eliminates insert, update, and delete anomalies.

- **4. A common approach to normalization is to ___ the larger table into smaller tables and link them together by using relationships.**

1. Add
2. Subtract
3. Multiply
4. Divide

- **4. A common approach to normalization is to ___ the larger table into smaller tables and link them together by using relationships.**

1. Add
2. Subtract
3. Multiply
4. Divide

4. A common approach to normalization is to ___ the larger table into smaller tables and link them together by using relationships.

1. Add
2. Subtract
3. Multiply
4. Divide

Answer: 4) Divide

Explanation:

A common approach to normalization is to divide the larger table into smaller tables and link them together by using relationships.

8. Which of the following is a type of Normal Form?

1. ACNF
2. BCNF
3. CCNF
4. DCNF

8. Which of the following is a type of Normal Form?

1. ACNF
2. BCNF
3. CCNF
4. DCNF

Answer: 2) BCNF

Explanation:
BCNF is a type of Normal Form.

- **When a relation contains an atomic value, it is a ___ relation.**

1. 1NF
2. 2NF
3. 3NF
4. BCNF

- **When a relation contains an atomic value, it is a ___ relation.**

1. 1NF
2. 2NF
3. 3NF
4. BCNF

Answer: 1) 1NF

Explanation:
When a relation contains an atomic value, it is a 1NF relation.

2NF relations are those that are in 1NF with all the attribute types dependent on the ___ key.

1. Primary
2. Foreign
3. Composite
4. Alternate

2NF relations are those that are in 1NF with all the attribute types dependent on the ___ key.

1. Primary
2. Foreign
3. Composite
4. Alternate

Answer: 1) Primary

Explanation:

2NF relations are those that are in 1NF with all the attribute types dependent on the primary key.

When a relation is in 2NF and there is ___, it is in 3NF.

1. Transition Dependency
2. No Transition Dependency
3. Relational Dependency
4. No Relational Dependency

When a relation is in 2NF and there is ___, it is in 3NF.

1. Transition Dependency
2. No Transition Dependency
3. Relational Dependency
4. No Relational Dependency

Answer: 2) No Transition Dependency

Explanation:

When a relation is in 2NF and there is no transition dependency, it is in 3NF.

A relation is in ___ if it is in Boyce Codd normal form and does not have any multivalued dependencies.

1. 1NF
2. 2NF
3. 3NF
4. 4NF

A relation is in ___ if it is in Boyce Codd normal form and does not have any multivalued dependencies.

1. 1NF
2. 2NF
3. 3NF
4. 4NF

Answer: 4) 4NF

Explanation:
A relation is in 4NF if it is in Boyce Codd normal form and does not have any multivalued dependencies.

- **If a relation has a 4NF and no join dependency, and when it joins, it should be___, it is considered 5NF.**

1. Lossful
2. Lesser
3. Lossless
4. Full

- **If a relation has a 4NF and no join dependency, and when it joins, it should be___, it is considered 5NF.**

1. Lossful
2. Lesser
3. Lossless
4. Full

Answer: 3) Lossless

Explanation:
If a relation has a 4NF and no join dependency, and when it joins, it should be lossless, it is considered 5NF.

What is TRUE about the First Normal Form (1NF)?

1. If a relation contains an atomic value, it will be 1NF.
2. A table attribute cannot contain more than one value, according to this rule.
3. A single-valued attribute can only be stored in it.
4. All of the above

What is TRUE about the First Normal Form (1NF)?

1. If a relation contains an atomic value, it will be 1NF.
2. A table attribute cannot contain more than one value, according to this rule.
3. A single-valued attribute can only be stored in it.
4. All of the above

Answer: 4) All of the above

Explanation:

In case of First Normal Form (1NF) -

1. If a relation contains an atomic value, it will be 1NF.
2. A table attribute cannot contain more than one value, according to this rule.
3. A single-valued attribute can only be stored in it.

Neither multivalued nor composite attributes, nor their combinations, may be used in the ___ normal form.

1. First
2. Second
3. Third
4. Fourth

Neither multivalued nor composite attributes, nor their combinations, may be used in the ___ normal form.

1. First
2. Second
3. Third
4. Fourth

Answer: 1) First

Explanation:

- Neither multivalued nor composite attributes, nor their combinations, may be used in the first normal form.

What is TRUE about the Second Normal Form (2NF)?

1. Relational must belong to 1NF in the 2NF.
2. All attributes other than the primary key are fully functional in the second normal form
3. Both A and B
4. None of the above

What is TRUE about the Second Normal Form (2NF)?

1. Relational must belong to 1NF in the 2NF.
2. All attributes other than the primary key are fully functional in the second normal form
3. Both A and B
4. None of the above

Answer: 3) Both A and B

Explanation:

In case of Second Normal Form (2NF) -

1. Relational must belong to 1NF in the 2NF.
2. All attributes other than the primary key are fully functional in the second normal form

What is TRUE about the Third Normal Form (3NF)?

1. When a relation is in 2NF and does not contain transitive partial dependencies, it will be in 3NF.

2. Data duplication is reduced by using 3NF.

3. It helps maintain the integrity of the data.

4. All of the above

What is TRUE about the Third Normal Form (3NF)?

1. When a relation is in 2NF and does not contain transitive partial dependencies, it will be in 3NF.
2. Data duplication is reduced by using 3NF.
3. It helps maintain the integrity of the data.
4. All of the above

Answer: 4) All of the above

Explanation:

In case of Third Normal Form (3NF) –

1. When a relation is in 2NF and does not contain transitive partial dependencies, it will be in 3NF.
2. Data duplication is reduced by using 3NF.
3. It helps maintain the integrity of the data.

Non-prime attributes cannot be transitively dependent, so the relation must have the ___ normal form.

1. First
2. Second
3. Third
4. Fourth

Non-prime attributes cannot be transitively dependent, so the relation must have the ___ normal form.

1. First
2. Second
3. Third
4. Fourth

Answer: 3) Third

A relation is in ___ if it is in Boyce Codd normal form and does not have any multivalued dependencies.

1. 1NF
2. 2NF
3. 3NF
4. 4NF

A relation is in ___ if it is in Boyce Codd normal form and does not have any multivalued dependencies.

1. 1NF
2. 2NF
3. 3NF
4. 4NF

Answer: 4) 4NF

- **If more than one value of B is present for a single value of A in a dependency A -> B, then the relationship is ___.**

1. Single
2. Multi-valued
3. Both a and b
4. None of the above

- **If more than one value of B is present for a single value of A in a dependency A -> B, then the relationship is ___.**

1. Single
2. Multi-valued
3. Both a and b
4. None of the above

Answer: 2) Multi-valued

What is TRUE about 5NF?

1. A relation is in 5NF if it is in 4NF, does not contain any join dependencies, and has lossless joining.
2. In order to avoid redundancy, 5NF ensures that the tables are broken up in as many ways as possible.
3. Project-join normal form (5NF) is sometimes referred to as Project-join NF.
4. All of the above

What is TRUE about 5NF?

1. A relation is in 5NF if it is in 4NF, does not contain any join dependencies, and has lossless joining.

2. In order to avoid redundancy, 5NF ensures that the tables are broken up in as many ways as possible.

3. Project-join normal form (5NF) is sometimes referred to as Project-join NF.

4. All of the above

Answer: 4) All of the above