Normal Forms

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

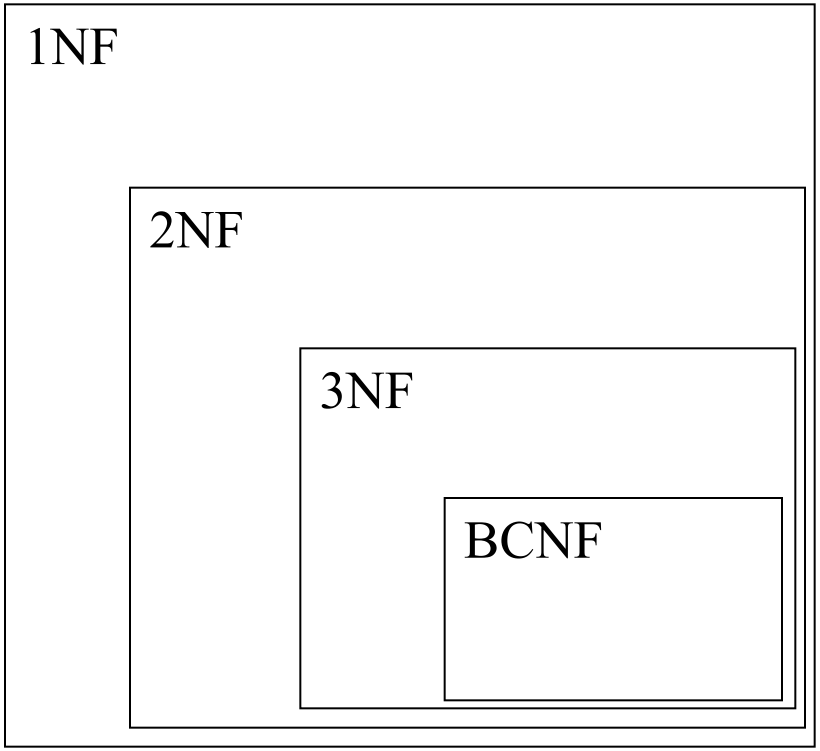
School of Computer Engineering

Dr. Hrudaya Kumar Tripathy
Associate Professor [II]

Chapter Contents

- Normal Forms
- 1NF (First Normal Form)
- Partial FD
- 2NF (Second Normal Form)
- Transitive FD
- 3NF (Third Normal Form)
- BCNF (Boyce-Codd Normal Form)

Normalization



a relation in BCNF, is also in 3NF

a relation in 3NF is also in 2NF

a relation in 2NF is also in 1NF

Normal Forms

- **Normal forms**
  - ✓ It provides a stepwise progression towards the construction of normalized relation schemas, which are free from data redundancies.
  - ✓ A relation schema is said to be in a particular normal form if it is satisfying certain defined conditions.

The *objective* of normalization:
"*to create relations where every dependency is on the key, the whole key, and nothing but the key*".

First Normal Forms (1NF)

- A relation will be 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.
- First normal form disallows the multi-valued attribute, composite attribute, and their combinations.
- Example: Relation STUDENT is not in 1NF because of multi-valued attributes are **Subject** and **Grade**.

Id No	Name	DOB	Subject	Grade
960100	Ram	14/11/77	Data Base, Soft Engg,ISDE	C A B
960105	Rishi	10/05/75	Soft Engg ISDE	B B
960120	Abhi	1/03/70	BDMS Soft Engg Workshop	A B C
96015	Ram	09/01/72	DBMS	B
960150	VIKASH	21/01/73	DBMS SOFT ENGGISDE WORKSHO P	D B C D

In the above table subject and grade are known as repeating groups. They have more than one value for each student. The problem arises because each student will study a number of subjects and we need to store their marks in each subject. We have solved the problem by putting those repeating values in to a single cell.

First Normal Form (1NF)

- **Solving the repeated values**

Id No	Name	DOB	Subject	Grade
960100	Ram	14/11/77	Data Base,	C
			Soft Engg,	A
			ISDE	B
960105	Rishi	10/05/75	Soft Engg	B
			ISDE	B
960120	Abhi	1/03/70	BDMS	A
			Soft Engg	B
			Workshop	C
96015	Ram	09/01/72	DBMS	B
960150	VIKASH	21/01/73	DBMS	D
			SOFT ENGG	B
			ISDE	C
			WORKSHOP	D

First Normal Form (1NF)

- **In order to store it as a relational database we must the data into INF i.e. we remove the repeating groups . By two ways we can solve the problem.**
  - 1. Flattern the table and Change the Key
  - 2. Decompostion the relation.

- **1. Flattern the table and Change the Key**

As we can only guess with student has got an **A grade** in soft Engg in line -2 . We cannot access the data where the primary Key. This cannot be left in the present state because the **id-no which** is the primary key has blank values it it. This is not allowed so, **the next stage is to fill the blanks.** we can simple copy down the data and fill in the id-no, name, and DOB fields.

Although it is an improvment , we are facing another problem, i.e the id-no no longer a promary key. so we have to choose a new primary key .**In the below table id-No and Subject is the new primary Key**

First Normal Form (1NF)

Id No	Name	DOB	Subject	Grade
960100	Ram	14/11/77	Data Base,	C
960100	Ram	14/11/77	Soft Engg,	A
960100	Ram	14/11/77	ISDE	B
960105	Rishi	10/05/75	Soft Engg	B
	Rishi	10/05/75	ISDE	B
960120	Abhi	1/03/70	BDMS	A
960120	Abhi	1/03/70	Soft Engg	B
960120	Abhi	1/03/70	Workshop	C
96015	**Ram**	**09/01/72**	**DBMS**	**B**
960150	VIKASH	21/01/73	DBMS	D
960150	VIKASH	21/01/73	SOFT ENGG	B
960150	VIKASH	21/01/73	ISDE	C
960150	VIKASH	21/01/73	WORKSHOP	D

First Normal Form (1NF)

- **2. Decompostion the relation.**

- The alternative approach is to split the table in to two parts. **One part for repeating groups and another is for non repeating groups.**
- The primary key of the original relation sholud be included in both the relations.

Id No	Subject	Grade
960100	Data Base,	C
960100	Soft Engg,	A
960100	ISDE	B
960105	Soft Engg	B
960105	ISDE	B
960120	BDMS	A
960120	Soft Engg	B
960120	Workshop	C
96015	**DBMS**	**B**
960150	DBMS	D
960150	SOFT ENGG	B
960150	ISDE	C
960150	WORKSHOP	D

Id No	Name	DOB
960100	Ram	14/11/77
960105	Rishi	10/05/75
960120	Abhi	1/03/70
96015	**Ram**	**09/01/72**
960150	VIKASH	21/01/73

Partial FD

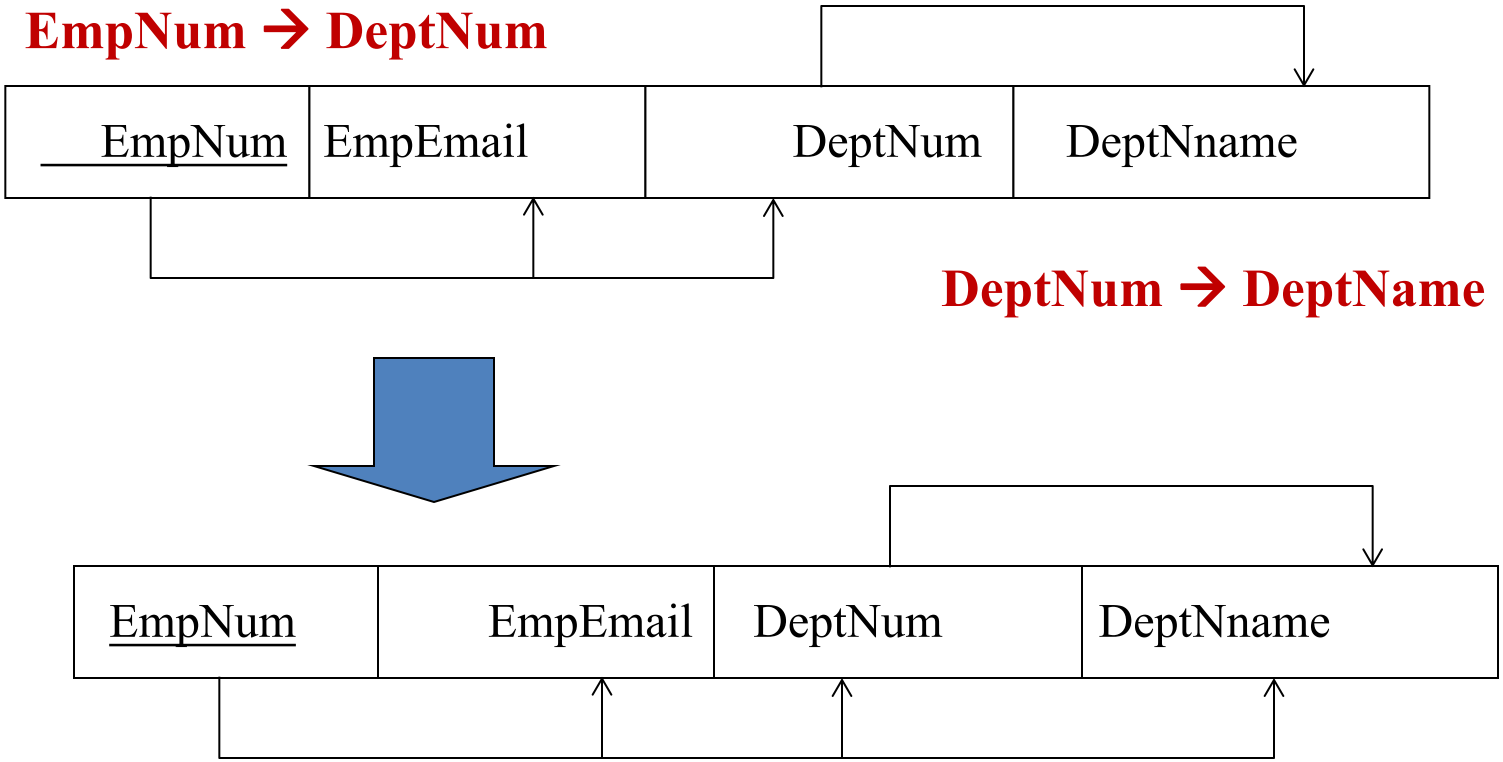
- A FD A → B is a partial FD, if some attribute of A can be removed and the FD still holds. That means there is some proper subset of A, C ⊂ A, such that C → B
  - ✓ **Key attributes:** are the attributes which are part of some candidate key
  - ✓ **Non-key attributes:** are the attributes which are not part of any candidate key

What is Partial Dependency in DBMS?

Ans: The FD (functional dependency) A->B happens to be a partial dependency if B is functionally dependent on A, and also B can be determined by any other proper subset of A. For instance, we have a relationship like;

MO->N, M->P, and P->N.

Transitive dependency

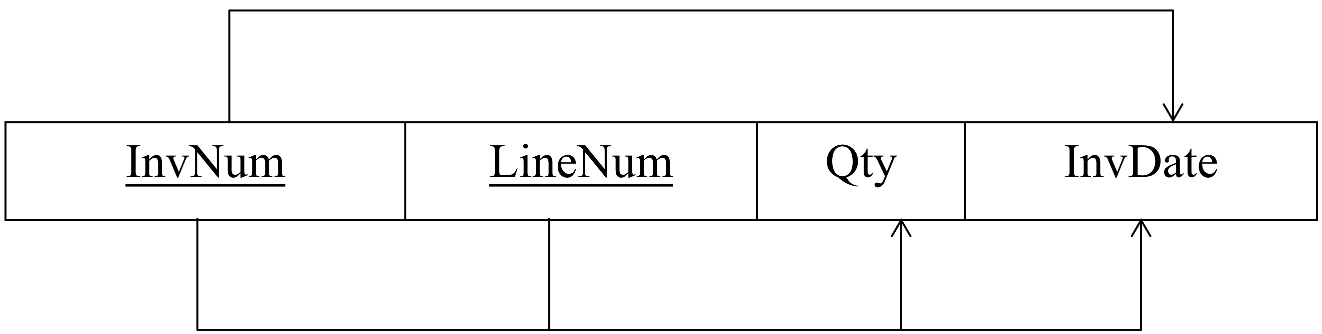


DeptName is ***transitively dependent*** on EmpNum via DeptNum

EmpNum → DeptName

Partial dependency

A **partial dependency** exists when an attribute B is functionally dependent on an attribute A, and A is a component of a multipart candidate key.



Candidate keys: {InvNum, LineNum} InvDate is ***partially dependent*** on {InvNum, LineNum} as **InvNum is a determinant of InvDate and InvNum is part of a candidate key**

2NF (Second Normal Form)

- A relation is in **2NF** iff the following two conditions are met simultaneously:
  - ✓ It is in 1NF
  - ✓ No non-key attribute is partially dependent on any key
- To be in second normal form, a relation must be in **first normal form** and relation **must not contain any partial dependency**.
- A relation is in 2NF iff it has **No Partial Dependency**, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Partial Dependency – If proper subset of candidate key determines non-prime attribute, it is called partial dependency.

2NF (Second Normal Form)

Example 2 – Consider following functional dependencies in relation R (A, B , C, D)

AB -> C [A and B together determine C]

BC -> D [B and C together determine D]

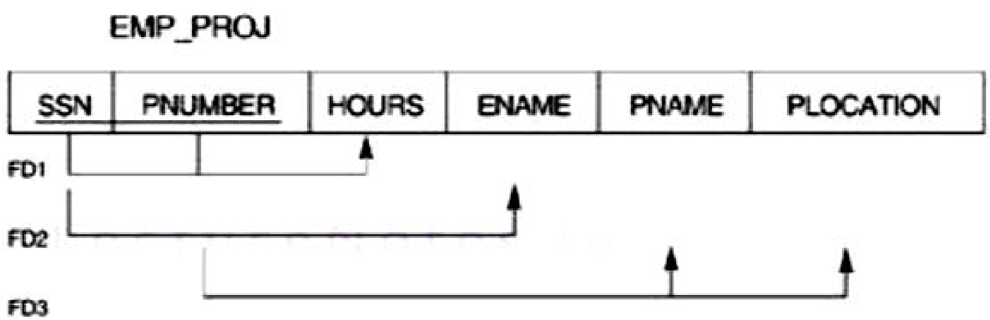
In the above relation, AB is the only candidate key and there is no partial dependency, i.e., any proper subset of AB doesn't determine any non-prime attribute.

2NF (Second Normal Form)

Second Normal Form (2NF)

Fully Functional Dependancy: A FD X→Y is said to be **Full functional dependency** when for any A∊X, (X-{A}) does not functionally determines Y. If (X-{A})→Y then we say X→Y is **partially dependency.**

Example: {SSN, PNUMBER}→HOURS is a full functional dependency where as {SSN, PNUMBER}→ENAME is a partial functional dependency.

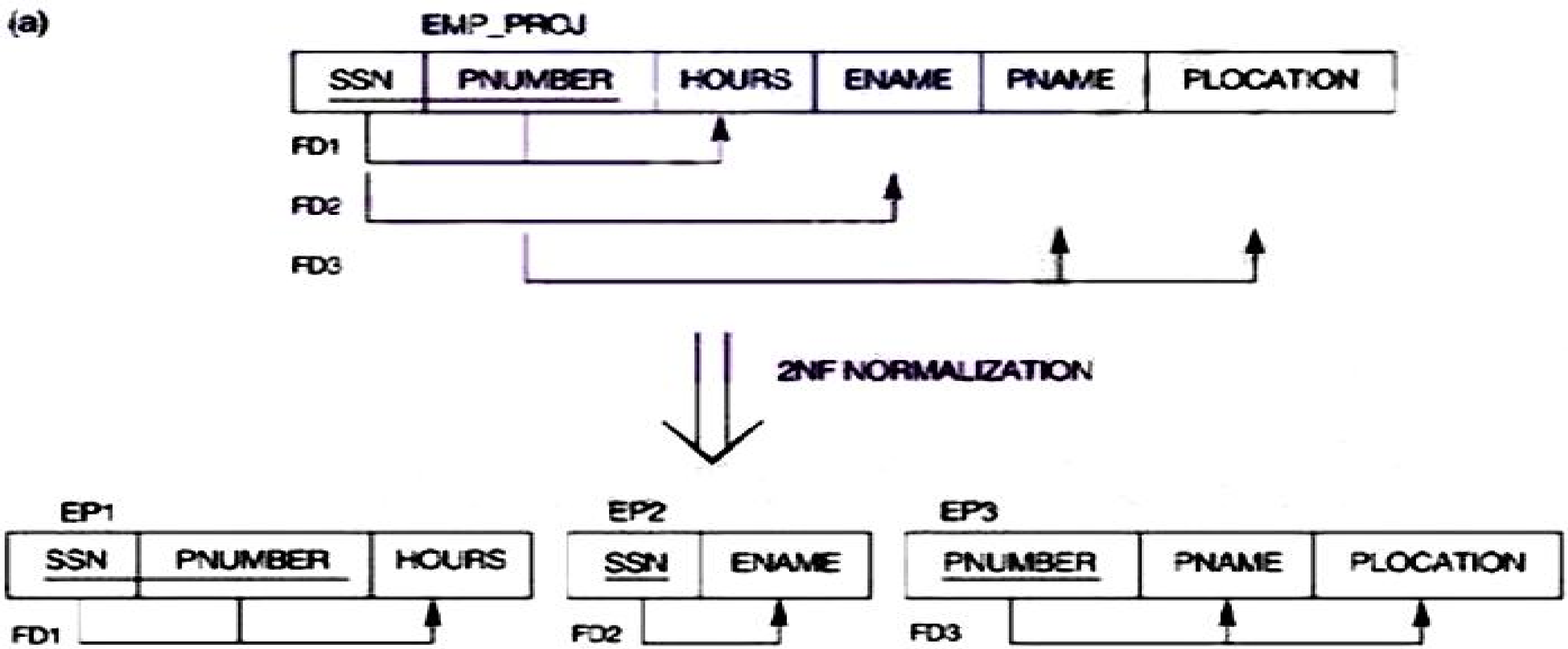


A relation is said to be in 2NF if it is in 1NF and if every nonprime attribute A in R is fully functionally dependendent on any key (candidate) of R.
If the key contains a single attribute the test need not be applied at all.

Example: EMP_PROJ is in 1NF but it is not in 2NF because for the FD2 and FD3. FD2 ENAME is dependent on SSN but it should be dependent on (SSN,PNUMBER). Similarly PNAME and PLOCATION are dependent on only PNUMBER but they should be dependent on (SSN, PNUMBER).

2NF (Second Normal Form)

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



2NF : Algorithm/Process to convert

Procedure: To verify that given relational schema R is in 2NF or NOT, If NOT then Convert it to 2NF:

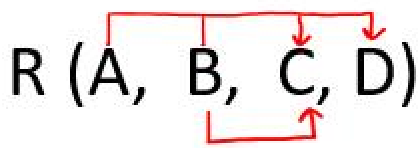
- **STEP 1:** Calculate the Candidate Key of given R by using an arrow diagram on R.
- **STEP 2:** Verify each FD with Definition of 2NF (No non-prime attribute should be partially dependent on Candidate Key)
- **STEP 3:** Make a set of FD which do not satisfy 2NF, i.e. all those FD which are partial.
- **STEP 4:** Convert the table R in 2NF by decomposing R such that each decomposition based on FD should satisfy the definition of 2NF:
- **STEP 5:** Once the decomposition based on FD is completed, create a separate table of attributes in the Candidate key.
- **STEP 6:** All the decomposed R obtained from STEP 4 and STEP 5 forms the required decomposition where each decomposition is in 2NF.

(Ref: https://www.javatpoint.com/dbms-questions-on-normalization)

2NF : Example-1

Q: Given a relation R(A, B, C, D) and Functional Dependency set FD = { AB → CD, B → C }, determine whether the given R is in 2NF? If not convert it into 2 NF.

Solution: Let us construct an arrow diagram on R using FD to calculate the candidate key.



- From above arrow diagram on R, we can see that an attributes **AB is not determined by any of the given FD**, hence **AB will be the integral part of the Candidate key**, i.e. no matter what will be the candidate key, and how many will be the candidate key, but all will have W compulsory attribute.
- **Let us calculate the closure of AB**
- AB + = ABCD (from the method we studied earlier)
- Since the closure of AB contains all the attributes of R, hence AB is Candidate Key
- From the definition of Candidate Key(Candidate Key is a Super Key whose no proper subset is a Super key)

2NF : Example-1

- Since all key will have **AB as an integral part,** and we have proved that AB is Candidate Key, Therefore, any superset of AB will be Super Key but not Candidate key.
- Hence there will be **only one candidate key AB**

Definition of 2NF: No non-prime attribute should be partially dependent on Candidate Key

- Since R has 4 attributes: - A, B, C, D, and Candidate Key is AB, Therefore, prime attributes (part of candidate key) are A and B while a non-prime attribute are C and D
  a) FD: **AB → CD** satisfies the definition of 2NF, that non-prime attribute(C and D) are fully dependent on candidate key AB
  b) FD: **B → C** does not satisfy the definition of 2NF, as a non-prime attribute(C) is partially dependent on candidate key AB(i.e. key should not be broken at any cost)

As FD B → C, the above table R(A, B, C, D) is not in 2NF

Convert the table R(A, B, C, D) in 2NF:
Since **FD: B → C,** our table was not in 2NF, let's decompose the table

2NF : Example-1

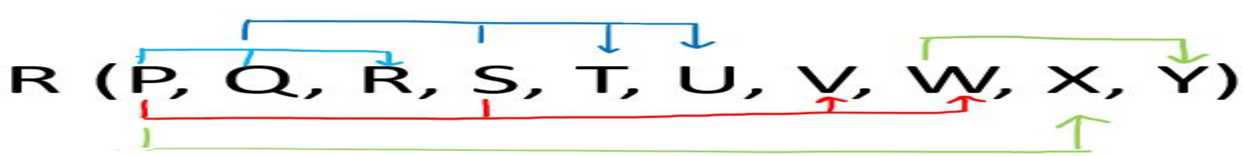
- **R1(B, C)**
- Since the key is AB, and from FD AB → CD, we can create R2(A, B, C, D) but this will again have a problem of partial dependency B → C, hence R2(A, B, D).
- Finally, the decomposed table which is in 2NF
  a) R1(B, C)
  b) R2(A, B, D)

2NF : Questions

- **2. Given a relation R(P, Q, R, S, T) and Functional Dependency set FD = { PQ → R, S → T }, determine whether the given R is in 2NF? If not convert it into 2 NF**

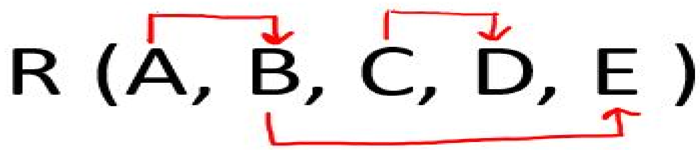
- From above arrow diagram on R, we can see that an attributes PQS is not determined by any of the given FD, hence PQS will be the integral part of the Candidate key, i.e., no matter what will be the candidate key, and how many will be the candidate key, but all will have PQS compulsory attribute..

- **3. Given a relation R(P, Q, R, S, T, U, V, W, X, Y) and Functional Dependency set FD = { PQ → R, PS → VW, QS → TU, P → X, W → Y }, determine whether the given R is in 2NF? If not convert it into 2 NF.**



2NF : Questions

4. Given a relation R(A, B, C, D, E) and Functional Dependency set FD = { A → B, B → E, C → D}, determine whether the given R is in 2NF? If not convert it into 2 NF.



- From above arrow diagram on R, we can see that an attributes AC is not determined by any of the given FD, hence AC will be the integral part of the Candidate key, i.e. no matter what will be the candidate key, and how many will be the candidate key, but all will have W compulsory attribute.

3NF (Third Normal Form)

Definition 1: A relational schema R is said to be in 3NF, First, it should be in 2NF and, no non-prime attribute should be transitively dependent on the Key of the table.

If X → Y and Y → Z exist then X → Z also exists which is a transitive dependency, and it should not hold.

or

Definition 2: First it should be in 2NF and if there exists a non-trivial dependency between two sets of attributes X and Y such that X → Y (i.e., Y is not a subset of X) (**Trivial:X→Y is a trivial FD i.e Y⊂= X**)

then

1. If X →Y, then X is a super key.
2. If X→Y , then (Y-X) is a prime attribute. (attribute prese)

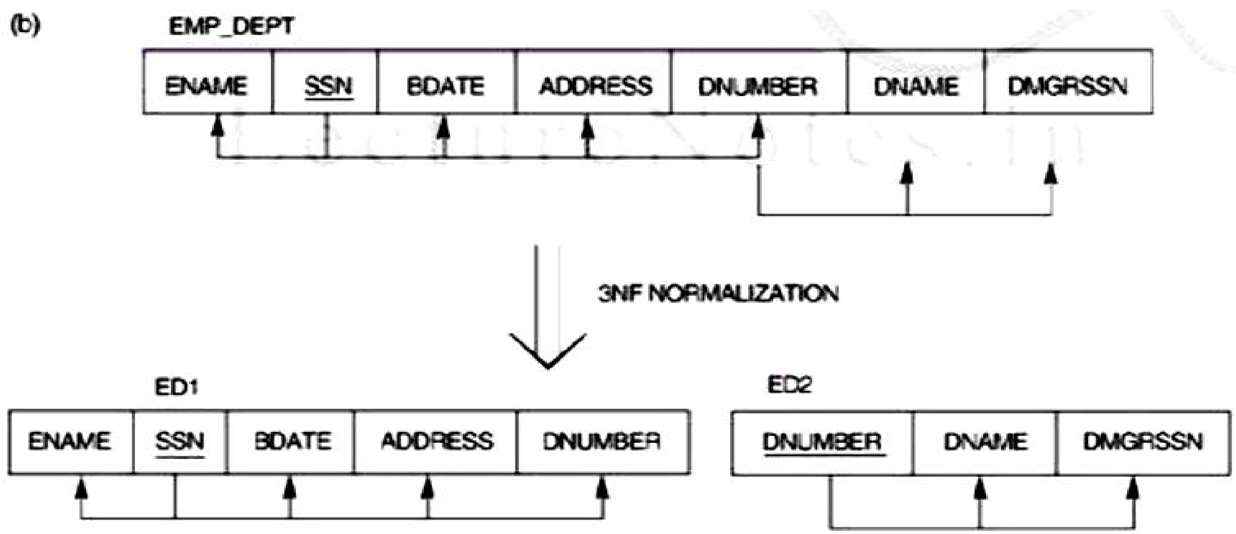
3NF (Third Normal Form)

Third Normal Form (3NF)

Transitive Functional Dependency: A FD X→Y in a relation schema is a transitive dependency if there is a set of attributes Z that is neither a candidate key nor a subset of any key of R and both X→Z and Z→Y holds.

Example: SSN→DMGRSSN is a transitive dependency through **DNUMBER** because we have **SSN→DNUMBER** and **DNUMBER→DMGRSSN** and DNUMBER is neither a key nor a part of key.

A relation is in 3NF if it is in 2NF and no nonprime attribute of R is transitively dependent on the primary key.



Solution: Decompose and set up a relation that includes the non key attributes that functionally determine other non key attribute.

Algorithm/Process to Convert

Followed to check whether the given relational schema R is in 3 NF or not? If not, how to decompose it into 3 NF.

STEP 1: Calculate the Candidate Key of given R by using an arrow diagram and then using the closure of an attribute on R, such that from the calculated candidate key, we can separate the prime attributes and non-prime attributes.

STEP 2: Verify each FD with **Definition of 3NF** (First it should be in 2NF and if there exist a non-trivial dependency between two sets of attributes X and Y such that $X \rightarrow Y$ (i.e., Y is not a subset of X) then Either X is Super Key or Y is a prime attribute).

STEP 3: Make a set of FD which does not satisfy 3NF, i.e. all those FD which do not have an attribute on the left side of FD as a super key or attribute on the right side of FD as a prime attribute.

STEP 4: Convert the table R in 3NF by decomposing R such that each decomposition based on FD should satisfy the definition of 3NF.

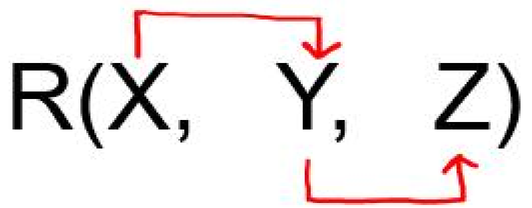
STEP 5: Once the decomposition based on FD is completed, create a separate table of attributes in the Candidate key.

STEP 6: All the decomposed R obtained from STEP 4 and STEP 5 forms the required decomposition where each decomposition is in 3NF.

3NF: Example-1

Question 1: Given a relation R(X, Y, Z) and Functional Dependency set FD = { X → Y and Y → Z }, determine whether the given R is in 3NF? If not convert it into 3 NF.

Solution: Let us construct an arrow diagram on R using FD to calculate the candidate key.



From above arrow diagram on R, we can see that an attribute **X is not determined by any of the given FD**, hence **X will be the integral part of the Candidate key**, i.e. no matter what will be the candidate key, and how many will be the candidate key, but all will have X compulsory attribute

Let us calculate the closure of X

X + = XYZ (from the closure method we studied earlier)

Since the closure of X contains all the attributes of R, hence **X is Candidate Key**

3NF: Example-1

- From the definition of Candidate Key (Candidate Key is a Super Key whose no proper subset is a Super key)
- Since **all key will have X as an integral part**, and we have proved that X is Candidate Key, Therefore, any superset of X will be Super Key but not the Candidate key.
- Hence there will be **only one candidate key X**

Since R has 3 attributes: - X, Y, Z, and Candidate Key is X, Therefore, prime attribute (part of candidate key) is X while a non-prime attribute are Y and Z
Given FD are X → Y and Y → Z

So, we can **write X → Z (which is a transitive dependency)**
In above **FD X → Z**, a non-prime attribute(Z) is transitively depending on the key of the table(X) hence as per the definition of 3NF it is not in 3 NF, **because no non-prime attribute should be transitively dependent on the key of the table.**

3NF: Example-1

- **Now check the above table is in 2 NF.**

FD: $X \rightarrow Y$ is in 2NF (as Key is not breaking and its Fully functional dependent)
FD: $Y \rightarrow Z$ is also in 2NF(as it does not violate the definition of 2NF)
Hence above table R(X, Y, Z) is in 2NF but not in 3NF.

We can also prove the same from Definition 2: First, it should be in 2NF and if there exists a non-trivial dependency between two sets of attributes X and Y such that $X \rightarrow Y$ (i.e., Y is not a subset of X) then

- Either X is Super Key
- Or Y is a prime attribute.

Since we have just proved that above table R is in 2 NF. Let's check it for 3NF using definition 2.
FD: $X \rightarrow Y$ is in 3NF (as X is a super Key)
FD: $Y \rightarrow Z$ is not in 3NF (as neither Y is Key nor Z is a prime attribute)
Hence because of $Y \rightarrow Z$ using definition 2 of 3NF, we can say that above table R is not in 3NF.

3NF: Example-1

Convert the table R(X, Y, Z) into 3NF:

Since due to FD: Y → Z, our table was not in 3NF, let's decompose the table
FD: Y → Z was creating issue, hence one table R1(Y, Z)
Create one Table for key X, R2(X, Y), since X → Y

Hence decomposed tables which are in 3NF are:
R1(X, Y)
R2(Y, Z)

3NF: Examples

Question 2: Given a relation R(X, Y, Z, W, P) and Functional Dependency set FD = { X → Y, Y → P, and Z → W}, determine whether the given R is in 3NF? If not convert it into 3NF.

Solution: Let us construct an arrow diagram on R using FD to calculate the candidate key.

R(X, Y, Z, W, P)

From the arrow diagram on R, we can see that an attributes XZ is not determined by any of the given FD, hence XZ will be the integral part of the Candidate key, i.e. no matter what will be the candidate key, and how many will be the candidate key, but all will have XZ compulsory attribute.

XZ + = XZYPW , Since the closure of XZ contains all the attributes of R, hence XZ is Candidate Key
Since R has 5 attributes: - X, Y, Z, W, P and Candidate Key is XZ, Therefore, prime attribute (part of candidate key) are X and Z while a non-prime attribute are Y, W, and P

Given FD are X → Y, Y → P, and Z → W and Super Key / Candidate Key is XZ

FD: X → Y does not satisfy the definition of 3NF, that neither X is Super Key nor Y is a prime attribute.
FD: Y → P does not satisfy the definition of 3NF, that neither Y is Super Key nor P is a prime attribute.
FD: Z → W satisfies the definition of 3NF, that neither Z is Super Key nor W is a prime attribute.
Convert the table R(X, Y, Z, W, P) into 3NF: Since all the FD = { X → Y, Y → P, and Z → W} were not in 3NF, let us convert R in 3NF

R1(X, Y) {Using FD X → Y}, R2(Y, P) {Using FD Y → P}, R3(Z, W) {Using FD Z → W}, And create one table for Candidate Key XZ, R4(X, Z) { Using Candidate Key XZ }
All the decomposed tables R1, R2, R3, and R4 are in 2NF(as there is no partial dependency) as well as in 3NF.

Hence decomposed tables are: R1(X, Y), R2(Y, P), R3(Z, W), and R4(X, Z)

3NF: Examples

Question 3: Given a relation R(P, Q, R, S, T, U, V, W, X, Y) and Functional Dependency set FD = { PQ → R, P → ST, Q → U, U → VW, and S → XY}, determine whether the given R is in 3NF? If not convert it into 3NF.

R(P, Q, R, S, T, U, V, W, X, Y)

PQ + = P Q R S T U X Y V W, Since the closure of XZ hence PQ is Candidate Key

Given FD are {PQ → R, P → ST, Q → U, U → VW and S → XY} and Super Key / Candidate Key is PQ

FD: PQ → R satisfy the definition of 3NF, as PQ Super Key
FD: P → ST does not satisfy the definition of 3NF, that neither P is Super Key nor ST is the prime attribute
FD: Q → U does not satisfy the definition of 3NF, that neither Q is Super Key nor U is a prime attribute
FD: U → VW does not satisfy the definition of 3NF, that neither U is Super Key nor VW is a prime attribute
FD: S → XY does not satisfy the definition of 3NF, that neither S is Super Key nor XY is a prime attribute

Since all the FD = { P → ST, Q → U, U → VW, and S → XY } were not in 3NF, let us convert R in 3NF

R1(P, S, T) {Using FD P → ST }, R2(Q, U) {Using FD Q → U }, R3(U, V, W) { Using FD U → VW },
R4(S, X, Y) { Using FD S → XY }, R5(P, Q, R) { Using FD PQ → R, and candidate key PQ }

All the decomposed tables R1, R2, R3, R4, and R5 are in 2NF(as there is no partial dependency) as well as in 3NF.

Hence decomposed tables are:

R1(P, S, T), R2(Q, U), R3(U, V, W), R4(S, X, Y), and R5(P, Q, R)

Boyce-codd Normal Form(BCNF)

- It is extension of 3NF on strict terms.
- A relation is in BCNF if atleast one of the condtion holds .
  I. X->Y is a trivial FD
  II. X->Y , then X is a super key.

- Every BCNF is a 3NF but vice-versa is not true
- Any relational schema with only two attributes is automatically in BCNF.

Example:1 R is in 3NF OR BCNF ?

R(ABC)

FD: {AB->C, C->A}

Ans:

(AB)+ = ABC, (BC)+ = BCA

so Candidate Key: AB and BC

AB->C (BCNF)

C-> A (is 3NF but not in BCNF)

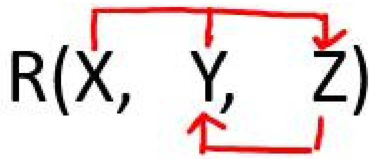
Algorithm: BCNF Conversion

Algorithm:

1. Calculate the Candidate Key of given R by using an arrow diagram and then using the closure of an attribute on R, such that from the calculated candidate key, we can separate the prime attributes and non-prime attributes.
2. Verify each FD with Definition of BCNF (First it should be in 3NF and if there exist a non-trivial dependency between two sets of attributes X and Y such that X → Y (i.e., Y is not a subset of X) then X is Super Key
3. Make a set of FD which does not satisfy BCNF, i.e. all those FD which do not have an attribute on the left side of FD as a super key
4. Convert the table R in BCNF by decomposing R such that each decomposition based on FD should satisfy the definition of BCNF.
5. Once the decomposition based on FD is completed, create a separate table of attributes in the Candidate key.
6. All the decomposed R obtained from STEP 4 and STEP 5 forms the required decomposition where each decomposition is in BCNF.

BCNF:Example-1

- **Question:** Given a relation **R(X, Y, Z)** and Functional Dependency set **FD = { XY → Z and Z → Y },** determine whether the given R is in BCNF? If not convert it into BCNF.
- **Solution**: Let us construct an arrow diagram on R using FD to calculate the candidate key.



- From the above arrow diagram on R, we can see that **an attribute X is** not determined by any of the given FD, **hence X will be the integral part of the Candidate key,** i.e. no matter what will be the candidate key, and how many will be the candidate key, but all will have X compulsory attribute.
- **Let us calculate the closure of X**
- Since the closure of X contains only X, hence it is not a candidate key.

Example-1…

Let us check the combination of Y, i.e. XY, XZ.

- **XY + = XYZ** (from the closure method we studied earlier)
  Since the closure of XY contains all the attributes of R, hence **XY is Candidate Key**
- **XZ + = XZY** (from the closure method we studied earlier)
  Since the closure of XZ contains all the attributes of R, hence **XZ is Candidate Key**
- Hence there are two **candidate key XY and XZ**

- **Since R has 3 attributes: - X, Y, Z**, and Candidate Key is **XY and XZ**, Therefore, prime attribute(part of candidate key) are X, Y, and Z while a non-prime attribute is none.

- **Using the Definition of 3NF to check whether R is in 3NF?:** First, it should be in 2NF and if there exists a non-trivial dependency between two sets of attributes X and Y such that X → Y (i.e. Y is not a subset of X) then

Example-1…

a) Either X is Super Key
b) **Or** Y is a prime attribute.

Given FD are XY → Z, and Z → Y and Super Key / Candidate Key are XZ and XY

- **FD: X Y → Z** satisfies the definition of 3NF, as XY is Super Key also Z is a prime attribute.
- **FD: Z → Y** satisfies the definition of 3NF, even though Z is not Super Key but Y is a prime attribute.

Since both FD of R, XY → Z and Z → Y satisfy the definition of 3NF hence R is in 3 NF

Using the Definition of BCNF to check whether R is in BCNF?:

First, it should be in 3NF and if there exists a non-trivial dependency between two sets of attributes X and Y such that X → Y (i.e. Y is not a subset of X) then

a) X is Super Key

Given FD are XY → Z, and Z → Y and Super Key / Candidate Key is XZ and XY

Example-1…

b) FD: **X Y → Z** satisfies the definition of BCNF, as XY is Super Key.
c) FD: **Z → Y** does not satisfy the definition of BCNF, as Z is not Super Key

Since both FD of R, XY → Z and Z → Y satisfy the definition of 3NF hence R is in 3 NF

Convert the table R(X, Y, Z) into BCNF:
Since due to FD: Z → Y, our table was not in BCNF, let's decompose the table

FD: Z→ Y was creating an issue, hence one table R1(Z, Y)

Create Table for key XY R2(X, Y) as XY was candidate key
Create Table for key XZ R2(X, Z) as XZ was candidate key

Example-1…

Considering R1(Z, Y) and R2(X, Y) both tables have one common attribute Y, but Y is not key in any of the table R1 and R2, hence we discard R2(X, Y) i.e. discarding candidate key XY.

Considering R1(Z, Y) and R3(X, Z) both tables have one common attribute Z, and Z is key of the table R1, hence we include R3(X, Z) i.e. including candidate key XZ.

Hence decomposed tables which are in BCNF:
R1(Z, Y)
R2(X, Z)

Example-2

Example-2

R(ABCDEFGHIJ)

FD: {AB->C, A->DE, B->F, F->GH, D->IJ}

Decompose it into BCNF

Ans:

Candidate Key check

(AB)+ = ABCDEFGHIJ

so Candidate Key= AB

AB-> C (BCNF)

R1(ABC) R2(DEFGHIJ)

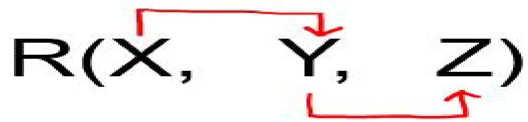
AB->C F->GH, D->IJ

(continute this process by calculatig key that every id left hand side is super key)

Example-3

Question 3: Given a relation R(X, Y, Z) and Functional Dependency set FD = { X → Y and Y → Z }, determine whether the given R is in BCNF? If not convert it into BCNF.

Solution: Let us construct an arrow diagram on R using FD to calculate the candidate key.



From the above arrow diagram on R, we can see that an attribute **X is not determined** by any of the given FD, **hence X will be the integral part of the Candidate key**, i.e. no matter what will be the candidate key, and how many will be the candidate key, but all will have X compulsory attribute.

Let us calculate the closure of X

X + = XYZ (from the closure method we studied earlier)

Since the closure of X contains all the attributes of R, **hence X is Candidate Key**

From the definition of Candidate Key (Candidate Key is a Super Key whose no proper subset is a Super key)

Example-3…

Using the Definition of BCNF to check whether R is in BCNF?: First, it should be in 3NF and if there exists a non-trivial dependency between two sets of attributes X and Y such that X → Y (i.e. Y is not a subset of X) then

a) X is Super Key

First, we check that table is in 3NF?

Using the Definition of 3NF to check whether R is in 3NF?: If there exists a non-trivial dependency between two sets of attributes X and Y such that X → Y (i.e. Y is not a subset of X) then

a) Either X is Super Key

b) Or Y is a prime attribute.

a) FD: X → Y is in 3NF (as X is a super Key)

b) FD: Y → Z is not in 3NF (as neither Y is Key nor Z is a prime attribute)

Hence because of Y → Z using definition 2 of 3NF, we can say that above table R is not in 3NF.

Example-3…

Convert the table R(X, Y, Z) into 3NF:
Since due to FD: Y → Z our table was not in 3NF, let's decompose the table
FD: Y → Z was creating issue, hence one table R1(Y, Z)
Create one Table for key X, R2(X, Y), since X → Y

Hence decomposed tables which are in 3NF:
R1(X, Y)
R2(Y, Z)
Both R1(X, Y) and R2(Y, Z) are in BCNF

BCNF (Boyce-Codd Normal Form)

Q. BCNF is a stronger form of normalization than 3NF. Justify. (2 marks)

Ans: 3NF (citing from head so forgive slight wording inaccuracies) requires that every *non-key* attribute is fully and non-transitively dependent on each candidate key. There is no such requirement for key attributes.

- BCNF requires that *every* attribute is fully and non-transitively dependent on each candidate key. Which can also be phrased as: satisfies 3NF and in addition also requires that every *key* attribute is fully and non-transitively dependent on each candidate key.
- The above means that every BCNF relation is also 3NF, but every 3NF relation is not necessarily BCNF. Hence BCNF is stronger than 3NF.

BCNF (Boyce-Codd Normal Form)

- **Find the highest normal form of given relation schema R(A,B,C,D,E) with set of functional dependencies, F = {AB→CE, BC→D, C→E}.**

Find the highest normal form of given relation schema R(A,B,C,D,E) with set of functional dependencies, F = {AB→CE, BC→D, C→E}.

Solution:

R is NOT in BCNF as (AB)+ = ABCDE = R ⇒ **AB is the key**

(BC)+ ≠ R and C+ ≠ R

R is NOT in 3NF as C→E is a transitive FD.

But 'R' is in 2 Normal Form as neither BC→D or C→E is a partial FD.

BCNF (Boyce-Codd Normal Form)

Q. BCNF is a stronger form of normalization than 3NF. Justify. (2 marks)

Ans: 3NF (citing from head so forgive slight wording inaccuracies) requires that every *non-key* attribute is fully and non-transitively dependent on each candidate key. There is no such requirement for key attributes.

- BCNF requires that *every* attribute is fully and non-transitively dependent on each candidate key. Which can also be phrased as: satisfies 3NF and in addition also requires that every *key* attribute is fully and non-transitively dependent on each candidate key.
- The above means that every BCNF relation is also 3NF, but every 3NF relation is not necessarily BCNF. Hence BCNF is stronger than 3NF.

Let R = (A, B, C, D, E) be a given relation schema with functional dependencies, F = {CE → D,D → B,C → A}.

(1+ 2+ 3 = 6 marks)

- Find all candidate keys.
- Check whether R is in BCNF or not?
- If R is not in BCNF, then do the BCNF decomposition and check whether the decomposition is lossless and dependency preserving decomposition or not??
-

Example

Solution:

Given: **R = (A,B,C,D,E) and F = {CE → D, D → B, C → A}**

a) The only candidate key is (CE),
As $(CE)^+$ = CEDBA = R

b) **R is NOT in BCNF** as D+ = DB ≠ R and C+ ≠ R (both D → B, C → A violates)
c)

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

R1=(D,B) with D → B　　**R2=(C,A)** with C → A　　**R3= (C,D,E)** with CE → D

Now R1, R2 and R3 are in BCNF

Check for Lossless join decomposition:

	A	B	C	D	E
R1	a11	a12	a13	a14	a15
R2	a21	a22	a23	a24	a25
R3	a31	a32	a33	a34	a35

	A	B	C	D	E
R1	a11	B2	a13	B4	a15
R2	B1	a22	B3	a24	a25
R3	a31	a32	B3	B4	B5

	A	B	C	D	E
R1	a11	B2	a13	B4	a15
R2	B1	a22	B3	a24	a25
R3	~~a31~~ B1	~~a32~~ B2	B3	B4	B5

All the values are B_j for j= 1-5. **Therefore it is a Lossless Join decomposition.**

THANK YOU!