Relational Database Design

Informal Design Guidelines for Relation Schema –

There are *four informal measures* for relation schema design

- Semantics of the attribute
- Reducing the redundant values in tuples (records)
- Reducing the null values in tuples
- Disallowing spurious tuples

Semantics of the Attributes –

semantics specifies how to interpret the attribute values stored in a tuple of a relation. The attributes defined must be self-explanatory. It should not be like xyz or 123.

Reducing Redundant Information in Tuples –

Minimize the storage space that the base relations (files) occupy. Grouping attributes into relation schemas has a significant effect on storage space.

For example, the space consumed by the two relations EMPLOYEE and DEPARTMENT is less than the space consumed by EMP_DEPT relation.

EMP_DEPT Relation

EMP_ID	Emp_Name	DOB	Address	Dept_No	DName	MGR_ID
101	Anurag	16/10/1975	Palam	1	CSC	M101
102	Kapil	11/12/1978	Dwarka	2	Elect	M102
103	Sunil	12/01/1988	Tilak Nagar	1	CSC	M101
104	Neeraj	16/04/1972	Uttam Nagar	2	Elect	M102
105	Pankaj	01/01/1970	Janakpuri	6	Tcom	M103
106	Ram	09/08/1977	Vikaspuri	2	Elect	M102
107	Pragati	10/11/1978	Gurgaon	NULL	NULL	NULL

EMPLOYEE Relation

EMP_ID	Emp_Name	DOB	Address	Dept_No
101	Anurag	16/10/1975	Palam	1
102	Kapil	11/12/1978	Dwarka	2
103	Sunil	12/01/1988	Tilak Nagar	1
104	Neeraj	16/04/1972	Uttam Nagar	2
105	Pankaj	01/01/1970	Janakpuri	6
106	Ram	09/08/1977	Vikaspuri	2
107	Pragati	10/11/1978	Gurgaon	NULL

DEPARTMENT Relation

Dept_No	DName	MGR_ID
1	CSC	M101
2	Elect	M102
6	Tcom	M103

The EMP_DEPT relation has following Anomalies –

Deletion Anomaly –

If we delete a tuple from this relation for Emp_ID = 105 then we also loose the information that we have a department Tcom (Telecom).

Insert Anomaly –

Suppose in the above relation a new employee joins in, say Emp_ID = 107 (Pragati) as a clerical staff and since long no department is allocated to her, in this case we have put three NULL values in the database.

Update Anomaly –

If we want to rename the Dept_Name Elect to Electronics then in the database, we have to make corrections at three different places (for Emp_ID = 102, 104 and 106).

Reducing Null values in Tuples –

If many of the attributes do not have values for some tuple in the relation, we end up with a number of NULL values in those tuples.

- The attribute does not apply to this tuple.
- The attribute value for this tuple is unknown.
- The value is known but absent.

Disallowing Spurious Tuples –Sometimes when we combine the tuple from two relations, we get spurious or wrong tuples/information that is not valid

STUDENT

Roll_No	Name	Class
201	Aayushi	B.E-II
202	Nisha	B.E. I
203	Rashmi	MCA-III

LIBRARY

Roll_No	Lib_Card_No	ACC_NO
201	10022	289
202	10209	128
203	10578	907

If we combine the above two tables then the result will form the Cartesian product and the resultant table will contain 9 records.

Result Without Equality Operator

Roll_No	Name	Class	Roll_No	Lib_Card_No	ACC_NO
201	Aayushi	B.E-II	201	10022	289
201	Aayushi	B.E-II	202	10209	128
201	Aayushi	B.E-II	203	10578	907
202	Nisha	B.E-I	201	10022	289
202	Nisha	B.E-I	202	10209	128
202	Nisha	B.E-I	203	10578	907
203	Rashmi	MCA-III	201	10022	289
203	Rashmi	MCA-III	202	10209	128
203	Rashmi	MCA-III	203	10578	907

Result With Equality Operator

Roll_No	Name	Class	Lib_Card_No	ACC_NO
201	Aayushi	B.E.-II	10022	289
202	Nisha	B.E-I	10209	128
203	[illegible]	MCA-III	10578	907

Functional Dependency (FD)

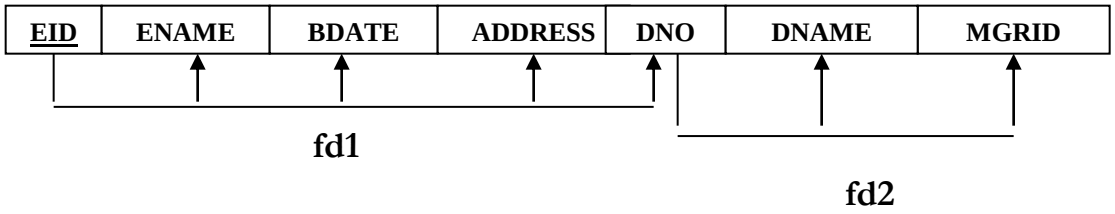
- Functional dependency (X→Y) between two sets of attributes X and Y, which are subsets of relation schema R, specifies a constraint on the possible tuples.

· This means that the values of the Y component of a tuple depends on/determined by the values of the X component (or Y is **functionally dependent** on X).

· Alternatively, the values of the X component of a tuple **uniquely/functionally determine** the values of the Y component.

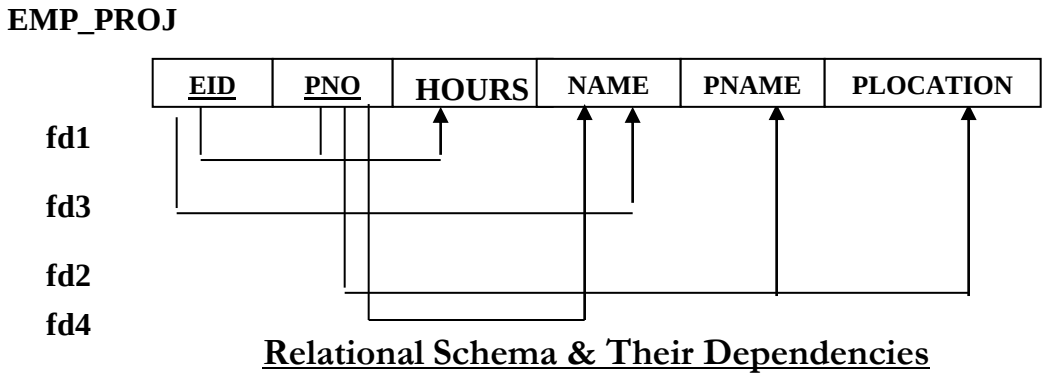
·

EMP_DEPT



Relational Schema & Their Dependencies

fd1: EID →{ENAME, BDATE, ADDRESS, DNO}
fd2: DNO →{DNAME, MGRID}



Relational Schema & Their Dependencies

In EMP_PROJ, there are three FDs –

Fd1: {EID, PNO}→ HOURS

Fd2: PNO → {PNAME, PLOCATION}

Fd3: EID → NAME

Fd4: PNO → NAME

fd3 & fd4 can be combined as {EID, PNO} → Name

Full Functional Dependency: A FD X → Y is a full functional dependency if removal of any attribute A from X (Primary Key) means that the **dependency does not exist/hold any more.**

Example – **{EID, PNO} → HOURS** is a full functional dependency.

Partial Dependency: A FD X → Y is a partial dependency if removal of any attribute A from X means that the **dependency still holds**.

Example – In the above figure, if we consider **{EID, PNO} → NAME.**
It is partial dependency because EID → NAME or PNO → NAME holds.

Transitive Dependency: A FD X → Y is a transitive dependency if there is a set of attributes Z that is not a subset of any key, and both X → Z and Z → Y holds.

Example – In the above figure the dependency **EID → MGRID** is transitive through **DNO** in EMP_DEPT relation because both the dependency **EID → DNO and DNO → MGRID hold and also DNO is not subset of the key EID.**

Normalization –

Normalization is a process in which unsatisfactory relation schemas are decomposed by breaking up their attributes into smaller relation schemas those having desirable properties.

Codd's Definition – The normalization process takes a relation schema through a series of tests to 'certify' whether or not it belongs to a certain **normal form**.

Prime Attribute – An attribute A in a relation R is called a prime attribute if it is a member/part of any key (candidate key) of the relation.

Nonprime Attribute – If A is not a member/part of any key (candidate key) of relation R.

Example – In **WORKS_ON** relation both EID and PNO are prime attributes, whereas other attributes (HOURS etc) are nonprime.

WORKS_ON

EID	**PNO**	**HOURS**	**…….**

First Normal Form (1NF) –

It states that the domain of attributes must include only **atomic (simple, individual) values** and the values of attribute in a tuple must be a **single value.** It means, It disallows **multi-valued attributes**, **composite attributes** and their combinations.

DNO	DNAME	MGRID	DLOCATION
1	Head office	102	{A-Block}
4	Administration	104	{B-Block}
5	Research	108	{C-Block, D-Block, A-Block}

DNO	DLOCATION	DNAME	MGRID
1	A-Block	Head Office	102
4	B-Block	Administration	104
5	A-Block	Research	108
5	C-Block	Research	108
5	D-Block	Research	108

DNO	DNAME	MGRID
1	Head office	102
4	Administration	104
5	Research	108

DNO	DLOCATION
1	A-Block
4	B-Block
5	A-Block
5	C-Block
5	D-Block

Student Course Relation (Not in 1NF)

Roll No*	Student Name	Subject	Course
11	Rashmi	Economics	M.A.
12	Kavita	Math & Physics	B.Sc.
13	Shalini	History & Civics	B.A.
14	Ram	Math	M.Sc.

Normalized Student Course Relation (In 1NF)

Roll No*	Student Name	Subject	Course
11	Rashmi	Economics	M.A.
12	Kavita	Math	B.Sc.
12	Kavita	Physics	B.Sc.
13	Shalini	History	B.A.
13	Shalini	Civics	B.A.
14	Ram	Math	M.Sc.

Second Normal form (2NF) –

A relation schema R is in 2NF if each and every **nonprime attribute** A in R is **fully functionally dependent** on the primary key of R.

Example:

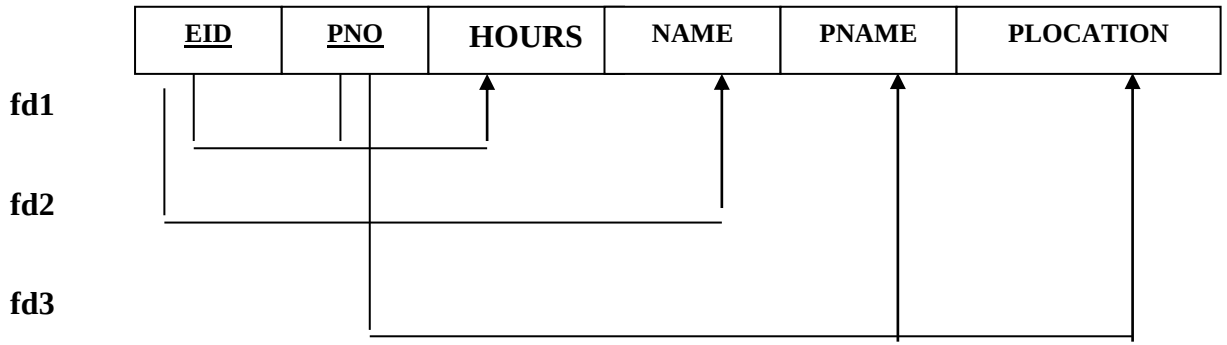
- **fd1:** {EID, PNO}→HOURS
  fd2: EID →NAME
  fd3: PNO →{PNAME, PLOCATION}

Example –

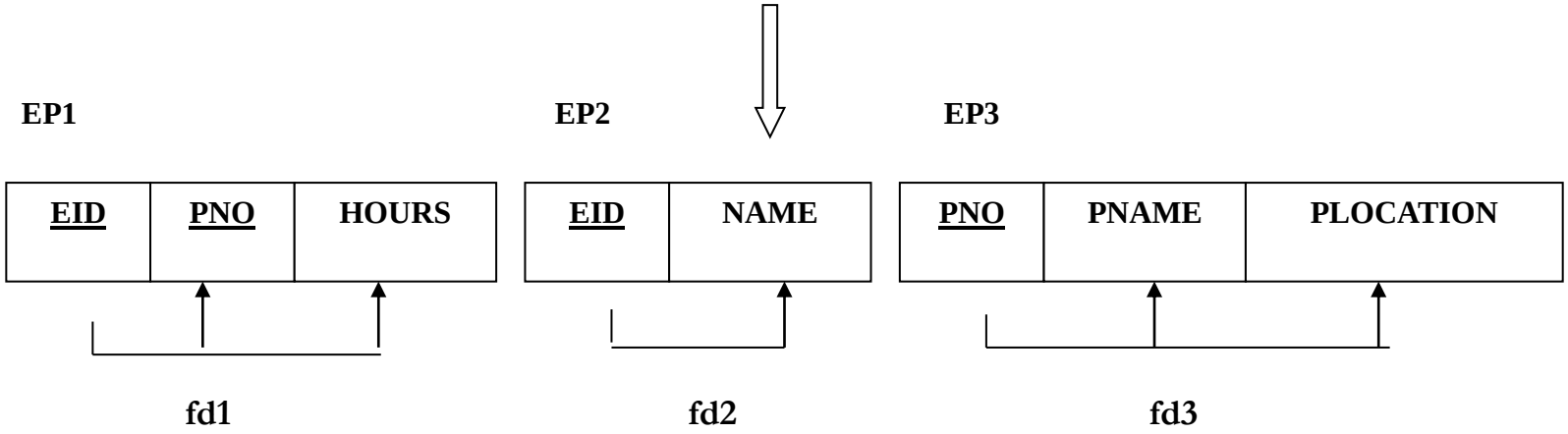
The EMP_PROJ relation is in 1NF but not in 2NF. The nonprime attribute NAME violates 2NF because of fd2. Similarly, nonprime attributes PNAME and PLOCATION because of fd3.

The functional dependencies fd2 and fd3 make NAME, PNAME, and PLOCATION partially dependent on the primary key {EID, PNO}.

EMPPROJ



Relational Schema That Is Not In 2NF



Relational Schema that is in 2NF

An Example Based on Second Normal Form

- **Consultant Client Relation (Not in Second Normal Form)**

Consultant ID*	Consultant Name	Client ID*	Client	Time Spent
S-1	Sunil	978	L&T	14 hrs
S-1	Sunil	665	IITM	26 hrs
S-1	Sunil	782	APJ	9 hrs
S-2	Shyam	221	P&G	67 hrs
S-2	Shyam	982	HLL	2 hrs
S-2	Shyam	665	BHEL	4 hrs

Normalized Consultant Client Relation (In Second Normal Form)

Consultant ID*	Client ID*	Time With Client
S-1	978	14 hrs
S-1	665	26 hrs
S-1	782	9 hrs
S-2	221	67 hrs
S-2	982	2 hrs
S-2	665	4 hrs

Client ID*	Client
978	L&T
665	IITM
782	APJ
221	P&G
982	HLL

Consultant ID*	Consultant Name
S-1	Sunil
S-2	Shyam

Third Normal Form (3NF) –

3NF is based on the concept of transitive dependency.

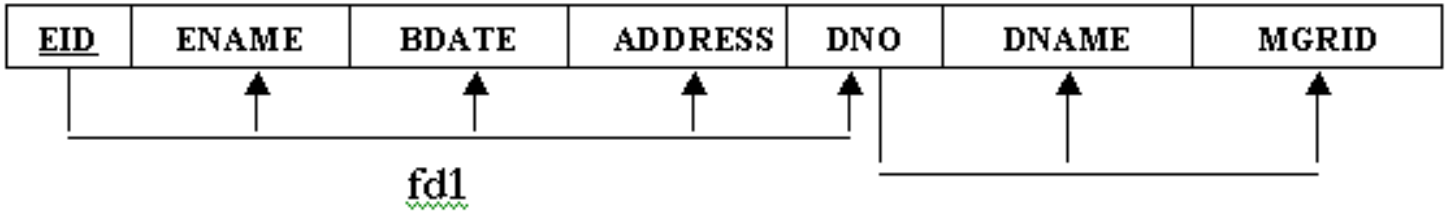
Definition – 1 – A relational schema R is in 3NF if it in 2NF and no nonprime attribute of R is transitively dependent on primary key.

Definition – 2 – **(General Definition of 3NF)** – A relational schema R is in 3NF if whenever a functional dependency $X \rightarrow A$ holds in R, **either**

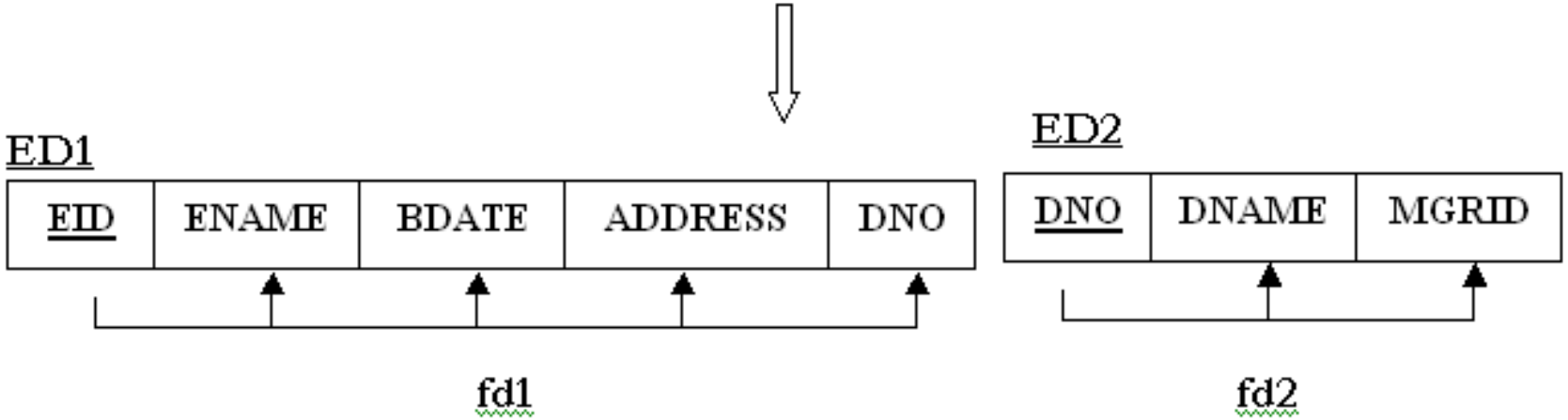
(a) X is super key of R **or**,

(b) A is a prime attribute of R.

EMP_DEPT



Relational Schema With Transitive Dependencies



Relation After Normalizing EMP_DEPT into 3 NF

Boyce-Codd Normal Form (BCNF) –

A relational schema R is in BCNF if whenever a functional dependency X → A holds in R, then X is a super key of R. [Part (b) of Definition - 2 is missing].

Note –

1) The only difference between BCNF and 3NF is that condition (b) of Definition – 2 of 3NF is absent from BCNF.

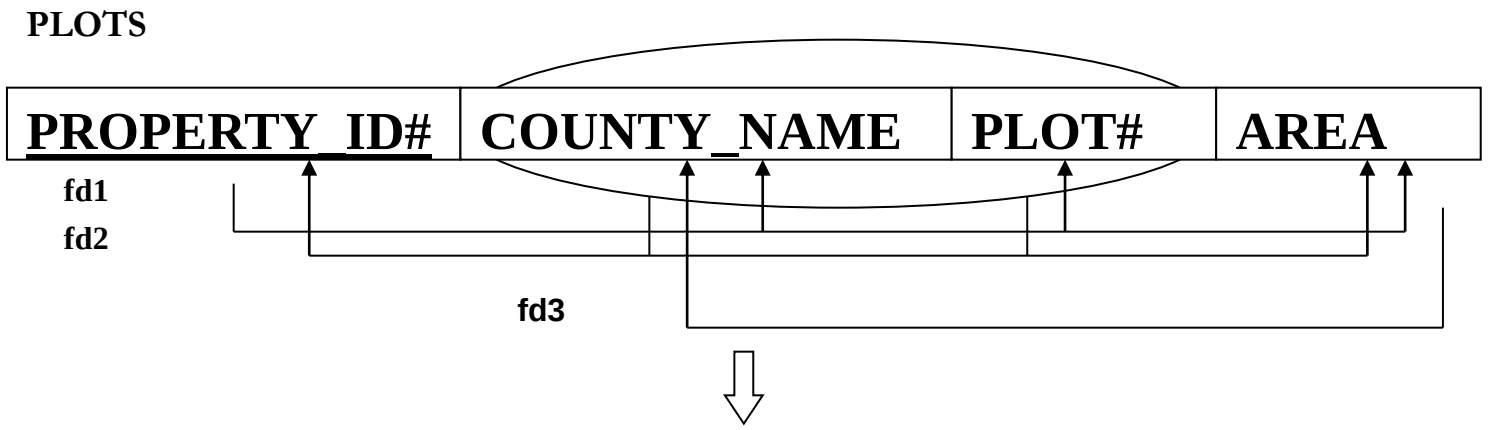
2) BCNF is stricter than 3NF, meaning that every relation in BCNF is also in 3NF. But a converse is not always true.

3) In practice, most of relational schemas that are in 3NF are also in BCNF.

PLOTS be a relational schema, which have two candidate keys PROPERTY_ID# and {COUNTY_NAME, PLOT#}.

- The relational schema PLOTS is in 3NF because fd1 & fd2 holds due to condition (a) of definition and fd3 holds due to condition (b) of definition.
- But relational schema PLOTS is not in BCNF because condition (b) does not exits in the definition of BCNF.
- We can decompose PLOTS into two BCNF relations PLOTS1 and PLOTS2 as shown below.

PLOTS Relation is in 3 NF but not in BCNF



PLOTS 1

PROPERTY_ID#	AREA	PLOT#

PLOTS 2

AREA	COUNTY_NAME

Fourth Normal Form

It is based on Multi-valued Dependency (MVD).

Informally, whenever two or more independent 1:N relationships are present in one relation, MVD may arise.

Whenever **X** $\rightarrow\!\!\!\!\rightarrow$ **Y** holds, we say that X **multi-determines** Y.

Example – 1 – Consider the relation EMP as shown below. A tuple in this relation represents the fact that an employee whose name is ENAME works on the project whose name is PROJ_NAME and has a dependent whose name is DEPEND_NAME. An employee may work on several projects and may have several dependents. Employee's projects and dependents are not directly related to one other.

EMP RELATION:

ENAME	PROJ_NAME	DEPEND_NAME
Rohit	X, Y	Vikas, Shikha
Sunil	X,Y	Kavita

In the above figure there are two MVDs ENAME → PROJ_NAME and ENAME → DEPEND_NAME hold in EMP relation. The employee with ENAME ‘Rohit’ works on projects with PROJ_NAME ‘X’ and ‘Y’ and has two dependents with DEPEND_NAME ‘Vikas’ and ‘Shikha’. Similarly employee with ENAME ‘Sunil’ works on projects ‘X’ and ‘Y’ with one dependent with DEPEND_NAME ‘Kavita’.

EMP_PROJECT

ENAME	PROJ_NAME
Rohit	X
Rohit	Y
Sunil	X
Sunil	Y

EMP_DEPENDENT

ENAME	DEPEND_NAME
Rohit	Vikas
Rohit	Shikha
Sunil	Kavita