Fall 2019

CS 220

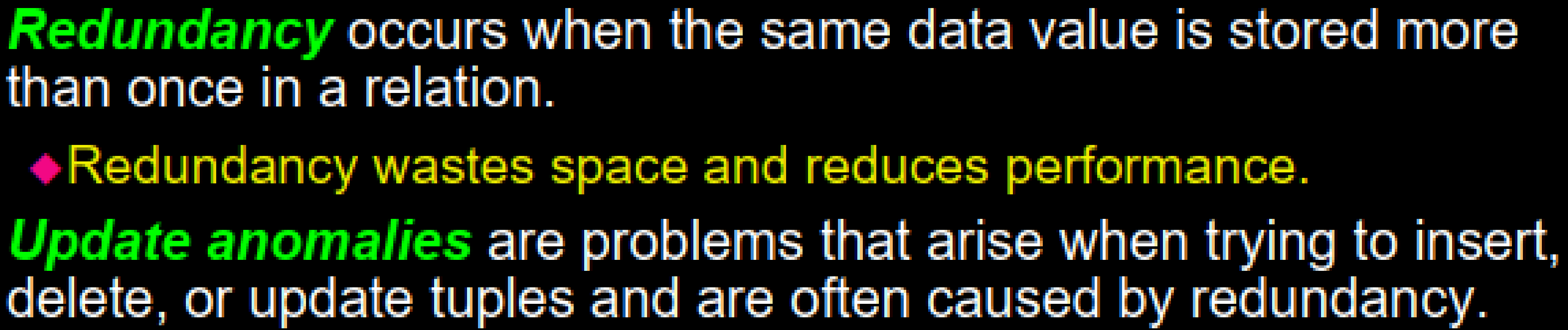
Database Systems

Lecture 11

Functional Dependence & Normalization

Ideal Relational Schema

Minimize **Redundancy** & **Update Anomalies**



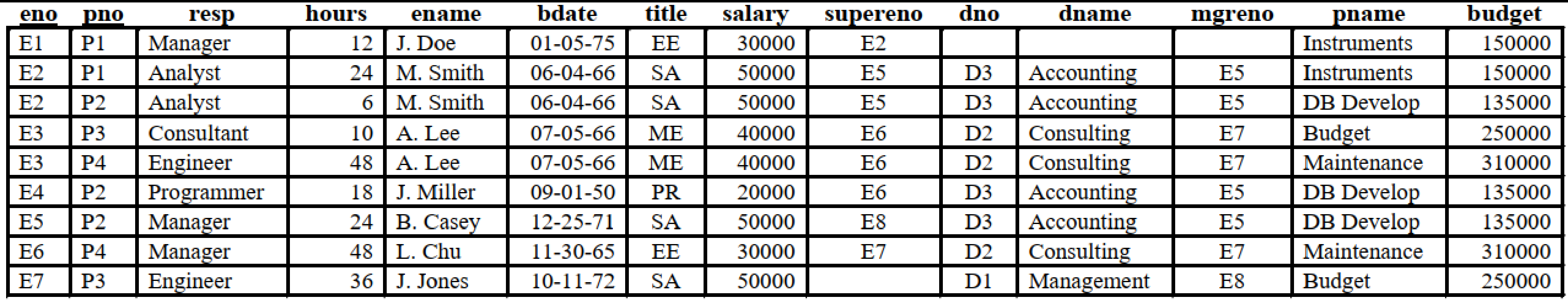
Redundancy occurs when the same data value is stored more than once in a relation.

- Redundancy wastes space and reduces performance.

Update anomalies are problems that arise when trying to insert, delete, or update tuples and are often caused by redundancy.

Universal Relation With All Attributes

Universal(eno, pno, resp, hours, ename, bdate, title, salary, supereno, dno, dname, mgreno, pname, budget)



eno	pno	resp	hours	ename	bdate	title	salary	supereno	dno	dname	mgreno	pname	budget
E1	P1	Manager	12	J. Doe	01-05-75	EE	30000	E2				Instruments	150000
E2	P1	Analyst	24	M. Smith	06-04-66	SA	50000	E5	D3	Accounting	E5	Instruments	150000
E2	P2	Analyst	6	M. Smith	06-04-66	SA	50000	E5	D3	Accounting	E5	DB Develop	135000
E3	P3	Consultant	10	A. Lee	07-05-66	ME	40000	E6	D2	Consulting	E7	Budget	250000
E3	P4	Engineer	48	A. Lee	07-05-66	ME	40000	E6	D2	Consulting	E7	Maintenance	310000
E4	P2	Programmer	18	J. Miller	09-01-50	PR	20000	E6	D3	Accounting	E5	DB Develop	135000
E5	P2	Manager	24	B. Casey	12-25-71	SA	50000	E8	D3	Accounting	E5	DB Develop	135000
E6	P4	Manager	48	L. Chu	11-30-65	EE	30000	E7	D2	Consulting	E7	Maintenance	310000
E7	P3	Engineer	36	J. Jones	10-11-72	SA	50000		D1	Management	E8	Budget	250000

Duplicate values?

Challenges with Update?

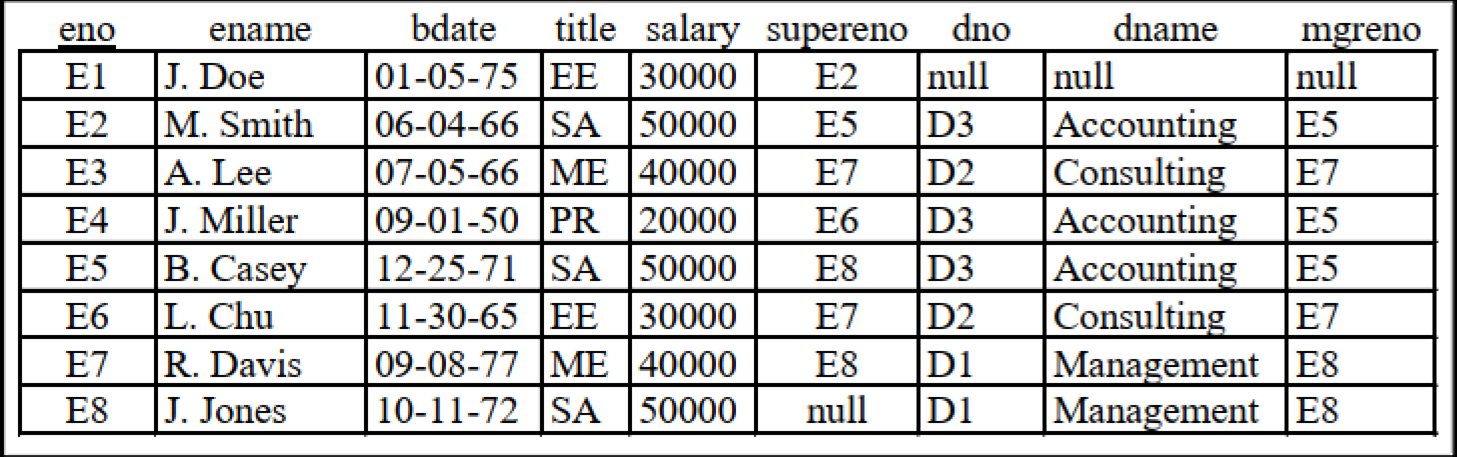
Update Anomalies

There are three major types of update anomalies:

- ***Insertion Anomalies*** - Insertion of a tuple into the relation either requires insertion of redundant information or cannot be performed without setting key values to `NULL`.

- ***Deletion Anomalies*** - Deletion of a tuple may lose information that is still required to be stored.

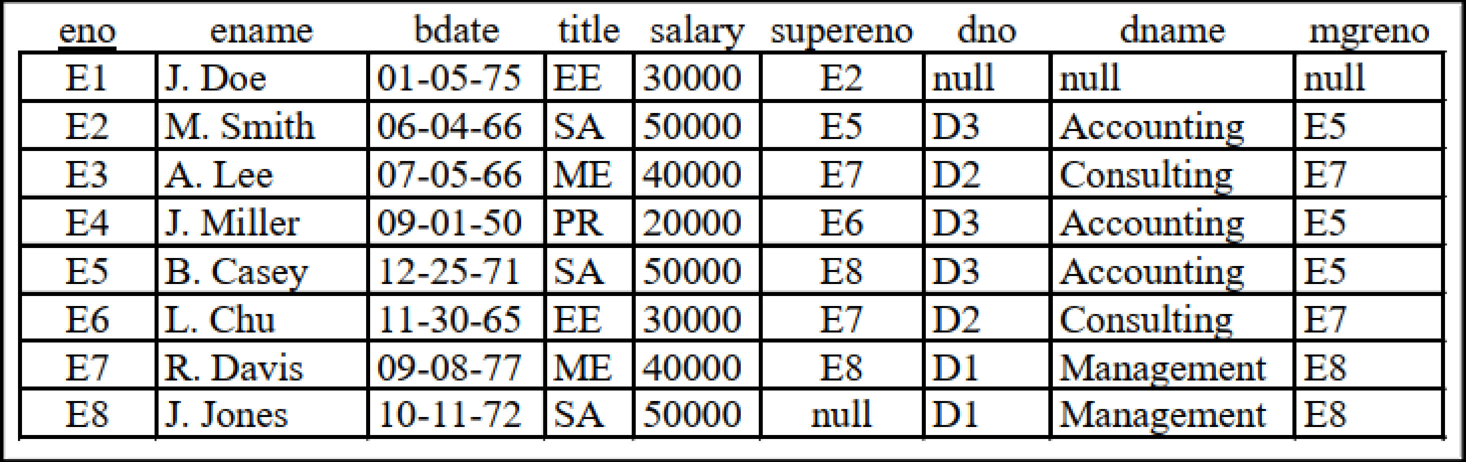
- ***Modification Anomalies*** - Changing an attribute of a tuple may require changing multiple attribute values in other tuples.



<u>eno</u>	ename	bdate	title	salary	supereno	dno	dname	mgreno
E1	J. Doe	01-05-75	EE	30000	E2	null	null	null
E2	M. Smith	06-04-66	SA	50000	E5	D3	Accounting	E5
E3	A. Lee	07-05-66	ME	40000	E7	D2	Consulting	E7
E4	J. Miller	09-01-50	PR	20000	E6	D3	Accounting	E5
E5	B. Casey	12-25-71	SA	50000	E8	D3	Accounting	E5
E6	L. Chu	11-30-65	EE	30000	E7	D2	Consulting	E7
E7	R. Davis	09-08-77	ME	40000	E8	D1	Management	E8
E8	J. Jones	10-11-72	SA	50000	null	D1	Management	E8

Consider these two types of insertion anomalies:

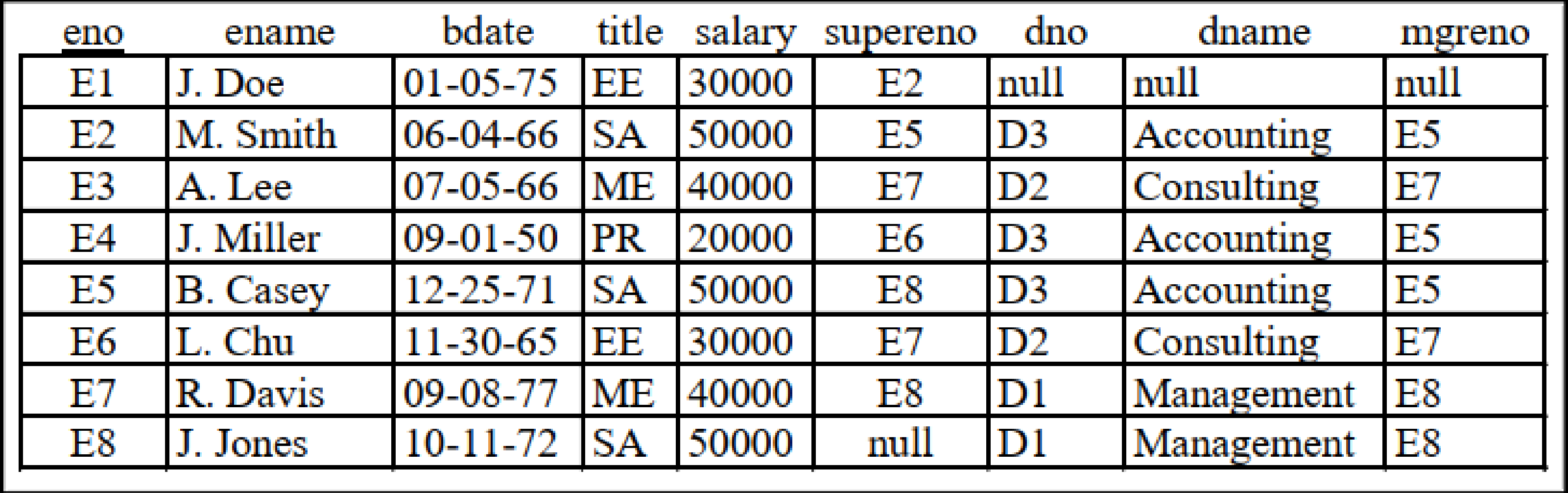
- 1) Insert a new employee E9 working in department D2.
  - ⇨ You have to redundantly insert the department name and manager when adding this record.
- 2) Insert a department D4 that has no current employees.
  - ⇨ This insertion is not possible without creating a dummy employee id and record because `eno` is the primary key of the relation.



eno	ename	bdate	title	salary	supereno	dno	dname	mgreno
E1	J. Doe	01-05-75	EE	30000	E2	null	null	null
E2	M. Smith	06-04-66	SA	50000	E5	D3	Accounting	E5
E3	A. Lee	07-05-66	ME	40000	E7	D2	Consulting	E7
E4	J. Miller	09-01-50	PR	20000	E6	D3	Accounting	E5
E5	B. Casey	12-25-71	SA	50000	E8	D3	Accounting	E5
E6	L. Chu	11-30-65	EE	30000	E7	D2	Consulting	E7
E7	R. Davis	09-08-77	ME	40000	E8	D1	Management	E8
E8	J. Jones	10-11-72	SA	50000	null	D1	Management	E8

Consider this deletion anomaly:

- Delete employees E3 and E6 from the database.
- Deleting those two employees removes them from the database, and we now have lost information about department D2!

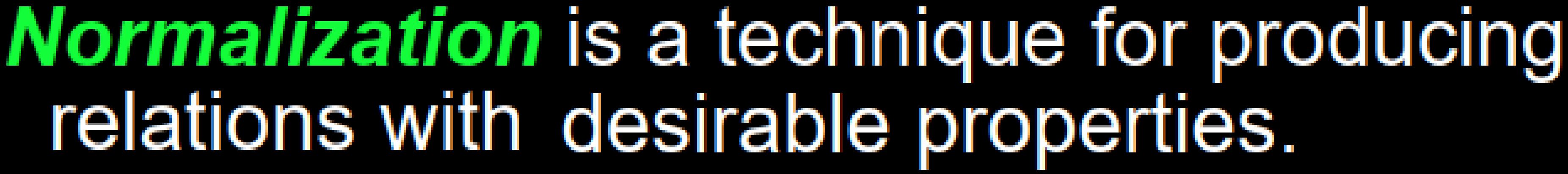


eno	ename	bdate	title	salary	supereno	dno	dname	mgreno
E1	J. Doe	01-05-75	EE	30000	E2	null	null	null
E2	M. Smith	06-04-66	SA	50000	E5	D3	Accounting	E5
E3	A. Lee	07-05-66	ME	40000	E7	D2	Consulting	E7
E4	J. Miller	09-01-50	PR	20000	E6	D3	Accounting	E5
E5	B. Casey	12-25-71	SA	50000	E8	D3	Accounting	E5
E6	L. Chu	11-30-65	EE	30000	E7	D2	Consulting	E7
E7	R. Davis	09-08-77	ME	40000	E8	D1	Management	E8
E8	J. Jones	10-11-72	SA	50000	null	D1	Management	E8

Consider these modification anomalies:

- 1) Change the name of department D3 to Embezzling.
  - You must update the department name in 3 different records.
- 2) Change the manager of D1 to E4.
  - You must update the `mgreno` field in 2 different records.

Normalization



Normalization is a technique for producing relations with desirable properties.

Normalization decomposes relations into smaller relations that contain less redundancy.

Decomposition requires that

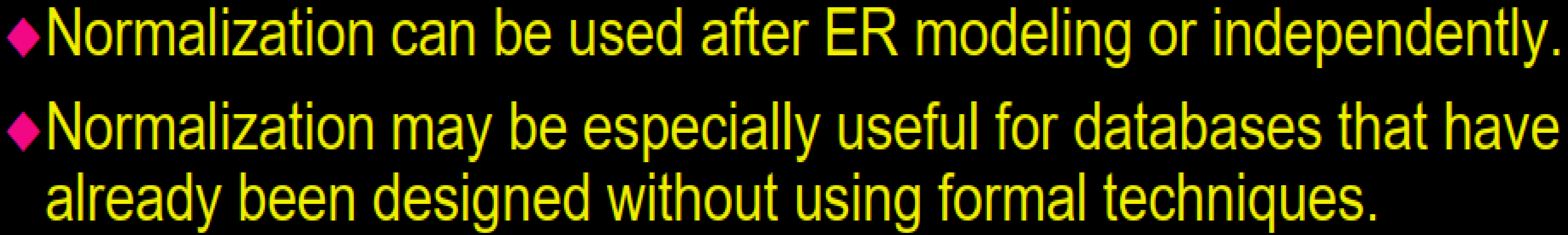
no information is lost

reconstruction of the original relations must be possible.

Relational schemas that are well-designed have several important properties:

- 1) The most basic property is that relations consists of attributes that are logically related.
  - The attributes in a relation should belong to only one entity or relationship.
- 2) ***Lossless-join property*** ensures that the information decomposed across many relations can be reconstructed using natural joins.
- 3) ***Dependency preservation property*** ensures that constraints on the original relation can be maintained by enforcing constraints on the normalized relations.

Normalization



- Normalization can be used after ER modeling or independently.
- Normalization may be especially useful for databases that have already been designed without using formal techniques.

The purpose of normalization is to develop good relational schemas that minimize **redundancies** and **update anomalies**

Functional Dependency

A functional dependency is a relationship between two attributes, typically between the **PK** and other **non-key attributes** within a table.

For a relation R, attribute Y is functionally dependent on attribute X, if for every valid instance of X can uniquely determines the value of Y.

Functional Dependency

A ***functional dependency*** (abbreviated ***FD***) is a statement about the relationship between attributes in a relation. We say a set of attributes *X* functionally determines an attribute *Y* if given the values of *X* we always know the only possible value of *Y*.

- Notation: $X \rightarrow Y$
- *X* functionally determines *Y*
- *Y* is functionally dependent on *X*

Example:

- eno → ename
- eno, pno → hours

Notation For Functional Dependency

A functional dependency has a left-side called the ***determinant*** which is a set of attributes, and one attribute on the right-side.

$$\underset{\substack{\uparrow \\ \text{determinant}}}{\text{eno, pno}} \rightarrow \underset{\substack{\uparrow \\ \text{determined} \\ \text{attribute}}}{\text{hours}}$$

$$\text{eno, pno} \rightarrow \text{hours}$$
$$\text{eno, pno} \rightarrow \text{resp}$$

Strictly speaking, there is always only one attribute on the RHS, but we can combine several functional dependencies into one:

$$\text{eno, pno} \rightarrow \text{hours, resp}$$

Remember that this is really short-hand for two functional dependencies.

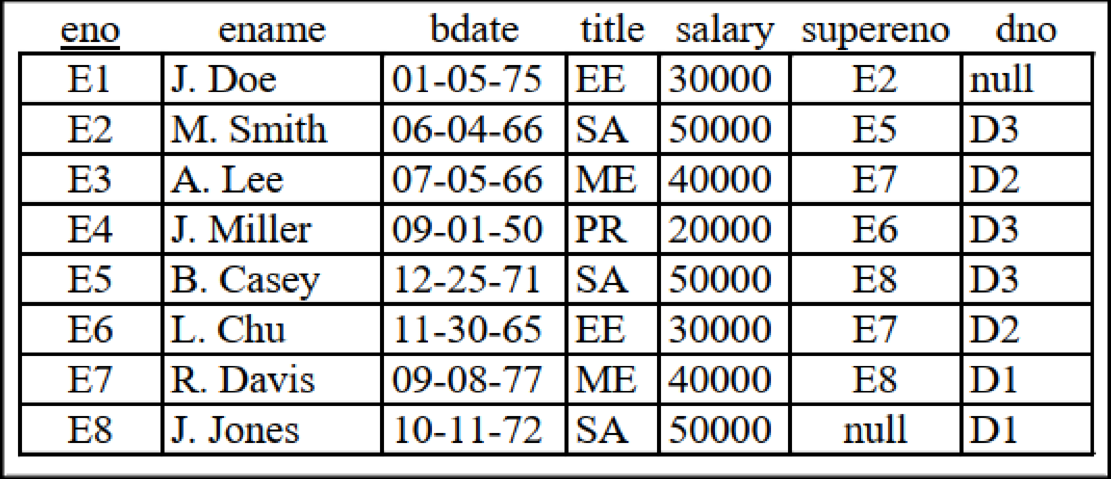
The Semantics of Functional Dependencies

Functional dependencies are a property of the ***domain*** being modeled **NOT** of the data instances currently in the database.

- This means that similar to keys you cannot tell if one attribute is functionally dependent on another by looking at the data.

Example:

Emp Relation



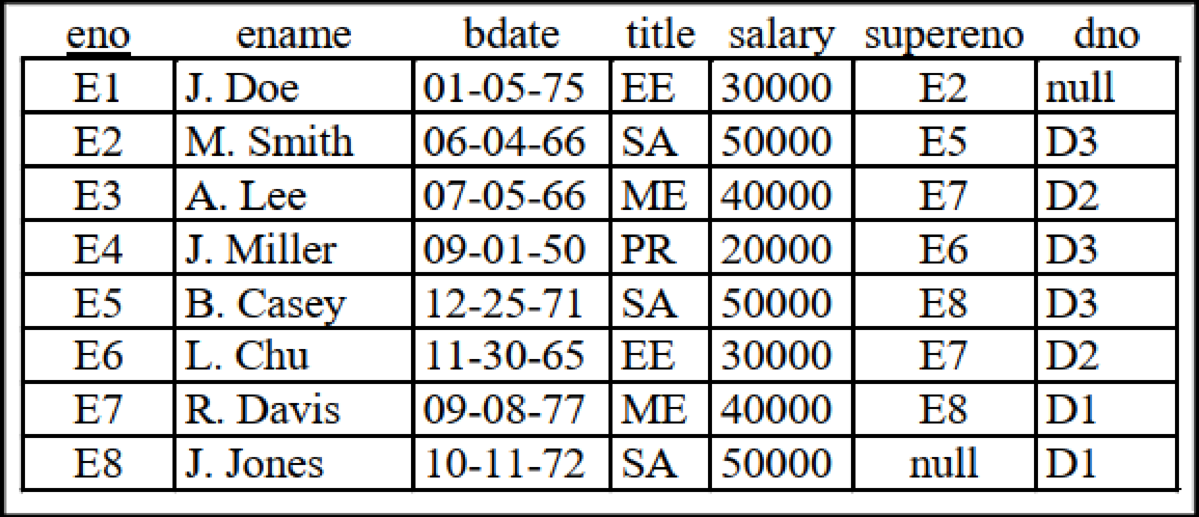
eno	ename	bdate	title	salary	supereno	dno
E1	J. Doe	01-05-75	EE	30000	E2	null
E2	M. Smith	06-04-66	SA	50000	E5	D3
E3	A. Lee	07-05-66	ME	40000	E7	D2
E4	J. Miller	09-01-50	PR	20000	E6	D3
E5	B. Casey	12-25-71	SA	50000	E8	D3
E6	L. Chu	11-30-65	EE	30000	E7	D2
E7	R. Davis	09-08-77	ME	40000	E8	D1
E8	J. Jones	10-11-72	SA	50000	null	D1

Functional dependencies are directional.

Why?

Example:

Emp Relation



eno	ename	bdate	title	salary	supereno	dno
E1	J. Doe	01-05-75	EE	30000	E2	null
E2	M. Smith	06-04-66	SA	50000	E5	D3
E3	A. Lee	07-05-66	ME	40000	E7	D2
E4	J. Miller	09-01-50	PR	20000	E6	D3
E5	B. Casey	12-25-71	SA	50000	E8	D3
E6	L. Chu	11-30-65	EE	30000	E7	D2
E7	R. Davis	09-08-77	ME	40000	E8	D1
E8	J. Jones	10-11-72	SA	50000	null	D1

Trivial Functional Dependencies

A functional dependency is ***trivial*** if the attributes on its left-hand side are a superset of the attributes on its right-hand side.

Examples: eno → eno

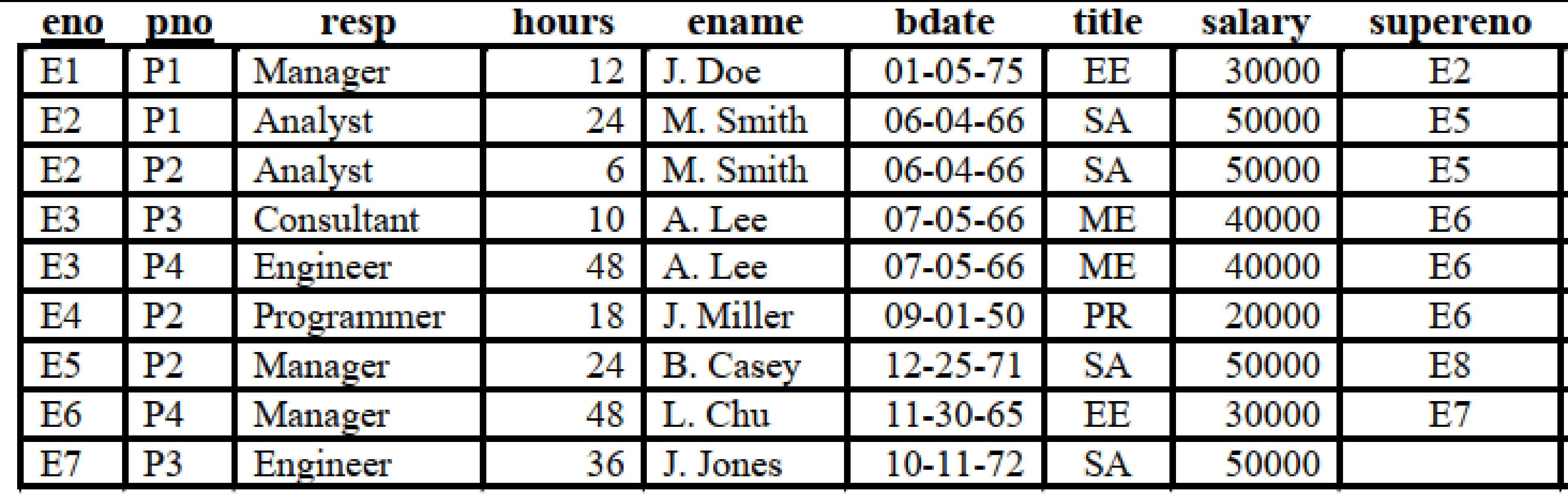
eno, ename → eno

eno, pno, hours → eno, hours

Trivial functional dependencies are not interesting because they do not tell us anything.

We are only interested in ***nontrivial*** FDs.

Identify all non-trivial Functional Dependencies



eno	pno	resp	hours	ename	bdate	title	salary	supereno
E1	P1	Manager	12	J. Doe	01-05-75	EE	30000	E2
E2	P1	Analyst	24	M. Smith	06-04-66	SA	50000	E5
E2	P2	Analyst	6	M. Smith	06-04-66	SA	50000	E5
E3	P3	Consultant	10	A. Lee	07-05-66	ME	40000	E6
E3	P4	Engineer	48	A. Lee	07-05-66	ME	40000	E6
E4	P2	Programmer	18	J. Miller	09-01-50	PR	20000	E6
E5	P2	Manager	24	B. Casey	12-25-71	SA	50000	E8
E6	P4	Manager	48	L. Chu	11-30-65	EE	30000	E7
E7	P3	Engineer	36	J. Jones	10-11-72	SA	50000	

Functional Dependencies & Keys

Functional dependencies can be used to determine the candidate and primary keys of a relation.

- For example, if an attribute functionally determines all other attributes in the relation, that attribute can be a key:

$$\text{eno} \rightarrow \text{eno, ename, bdate, title, supereno, dno}$$

Alternate definition of keys:

- A set of attributes *K* is a ***superkey*** for a relation *R* if the set of attributes *K* functionally determines all attributes in *R*.
- A set of attributes *K* is a ***candidate key*** for a relation *R* if *K* is a *minimal* superkey of *R*.

Functional Dependencies & Prime Attribute Types

A prime attribute type is an attribute type that is part of a candidate key

Example: R1(SSN, PNUMBER, PNAME, HOURS)

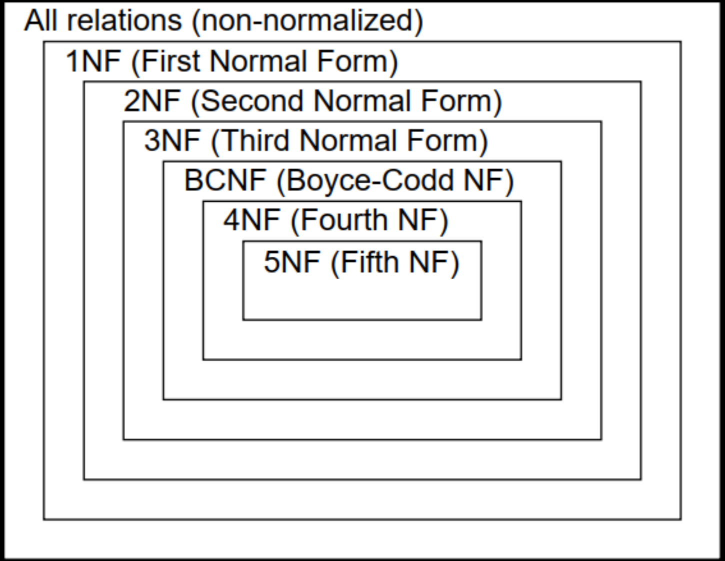
Prime attribute types: SSN and PNUMBER

Non-prime attribute types: PNAME and HOURS

Normalization Forms

A relation is in a particular ***normal form*** if it satisfies certain normalization properties.

- 1NF - First Normal Form
- 2NF - Second Normal Form
- 3NF - Third Normal Form
- BCNF - Boyce-Codd Normal Form
- 4NF - Fourth Normal Form
- 5NF - Fifth Normal Form



All relations (non-normalized)

1NF (First Normal Form)

2NF (Second Normal Form)

3NF (Third Normal Form)

BCNF (Boyce-Codd NF)

4NF (Fourth NF)

5NF (Fifth NF)

First Normal Form (1NF)

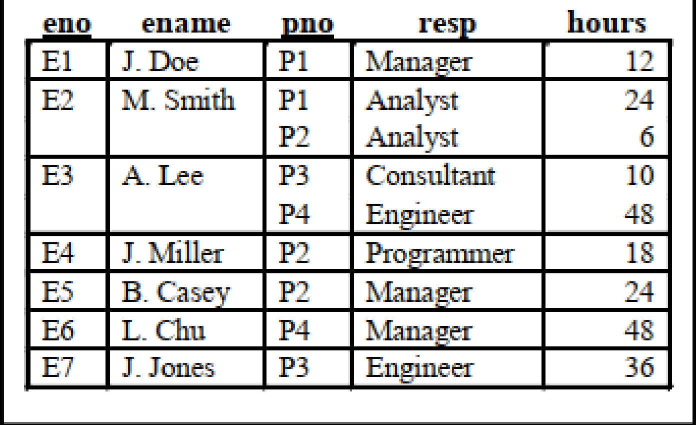
The first normal form (1 NF) states that every attribute type of a relation must be

atomic and **single valued**

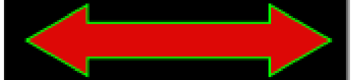
No Composite or **Multivalued** attribute types

SUPPLIER(SUPNR, NAME(FIRST NAME, LAST NAME), SUPSTATUS)

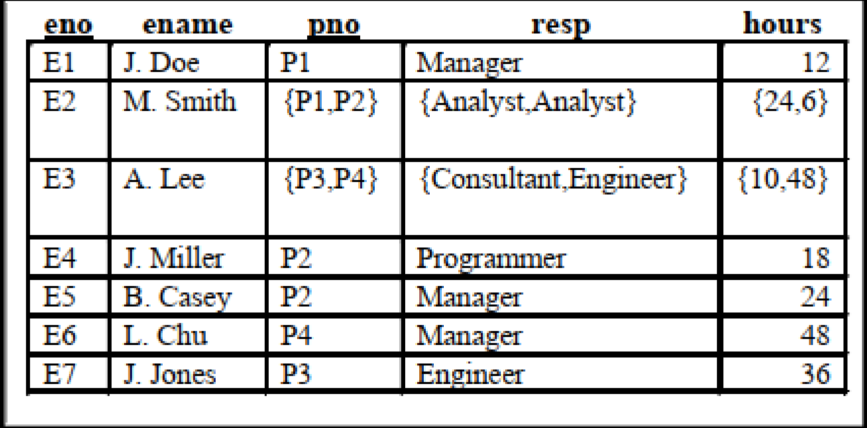
SUPPLIER(SUPNR, FIRST NAME, LAST NAME, SUPSTATUS)



eno	ename	pno	resp	hours
E1	J. Doe	P1	Manager	12
E2	M. Smith	P1 P2	Analyst Analyst	24 6
E3	A. Lee	P3 P4	Consultant Engineer	10 48
E4	J. Miller	P2	Programmer	18
E5	B. Casey	P2	Manager	24
E6	L. Chu	P4	Manager	48
E7	J. Jones	P3	Engineer	36

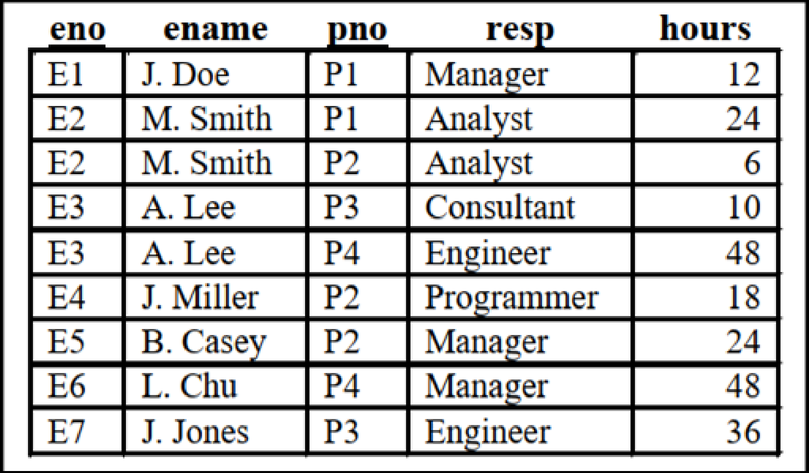


Two equivalent representations



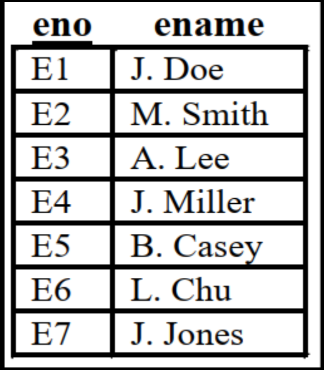
eno	ename	pno	resp	hours
E1	J. Doe	P1	Manager	12
E2	M. Smith	{P1,P2}	{Analyst,Analyst}	{24,6}
E3	A. Lee	{P3,P4}	{Consultant,Engineer}	{10,48}
E4	J. Miller	P2	Programmer	18
E5	B. Casey	P2	Manager	24
E6	L. Chu	P4	Manager	48
E7	J. Jones	P3	Engineer	36

Flattening

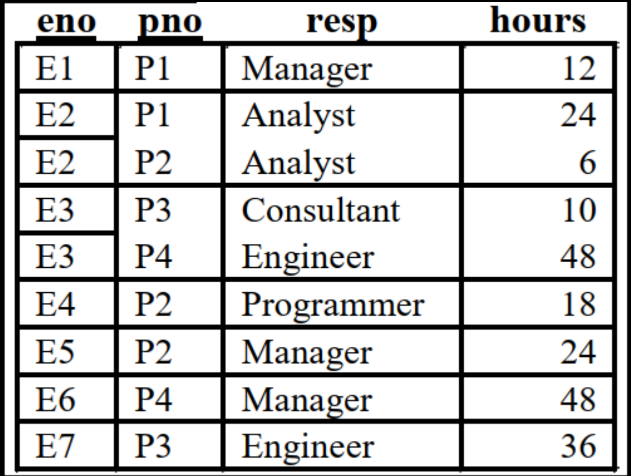


eno	ename	pno	resp	hours
E1	J. Doe	P1	Manager	12
E2	M. Smith	P1	Analyst	24
E2	M. Smith	P2	Analyst	6
E3	A. Lee	P3	Consultant	10
E3	A. Lee	P4	Engineer	48
E4	J. Miller	P2	Programmer	18
E5	B. Casey	P2	Manager	24
E6	L. Chu	P4	Manager	48
E7	J. Jones	P3	Engineer	36

Splitting



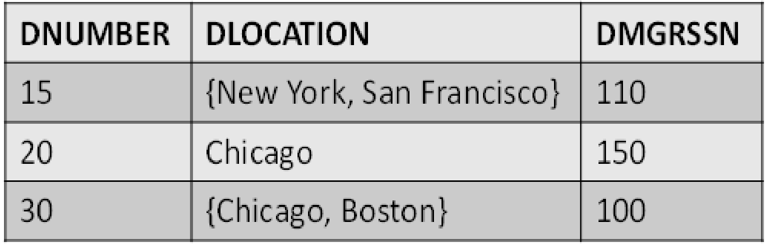
eno	ename
E1	J. Doe
E2	M. Smith
E3	A. Lee
E4	J. Miller
E5	B. Casey
E6	L. Chu
E7	J. Jones



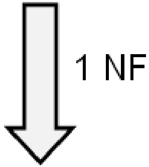
eno	pno	resp	hours
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36

DEPARTMENT(DNUMBER, DLOCATION, *DMGRSSN*)

Assumption: a department can have multiple locations and multiple departments are possible at a given location

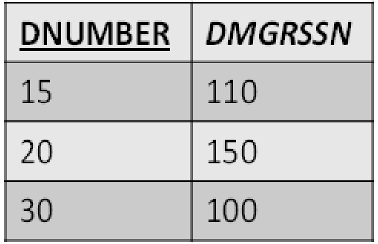


DNUMBER	**DLOCATION**	**DMGRSSN**
15	{New York, San Francisco}	110
20	Chicago	150
30	{Chicago, Boston}	100



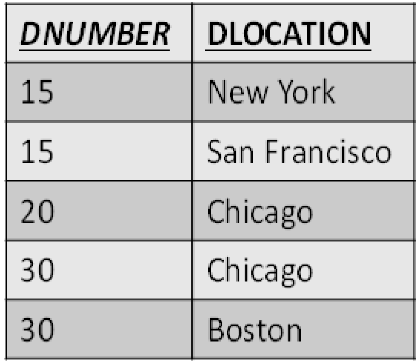
1 NF

DEPARTMENT



DNUMBER	***DMGRSSN***
15	110
20	150
30	100

DEP-LOCATION



DNUMBER	**DLOCATION**
15	New York
15	San Francisco
20	Chicago
30	Chicago
30	Boston

Full Functional Dependency

A functional dependency $A \rightarrow B$ is a ***full functional dependency*** if removal of any attribute from *A* results in the dependency not existing any more.

- We say that *B* is *fully functionally dependent* on *A*.
- If remove an attribute from *A* and the functional dependency still exists, we say that *B* is partially dependent on *A*.

eno → ename	(full FD)
eno, ename → salary, title	(partial FD - can remove ename)
eno, pno → hours, resp	(full FD)

Second Normal Form (2NF)

A relation R is in the second normal form (2 NF) if it **satisfies 1 NF** and every non-prime attribute A in R is **fully functional dependent on the primary key of R**

A prime attribute is an attribute in any candidate key.

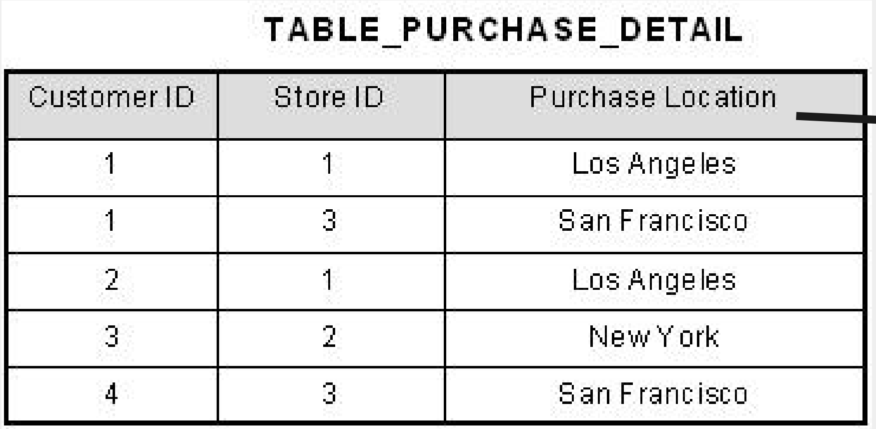
If the relation is not in second normal form, we must:

- Keep a relation with the original primary key and any attribute types that are fully functional dependent on it
- Set up a new relation for each partial key together with its dependent attribute types

Simple Example

This table has a composite primary key [Customer ID, Store ID]

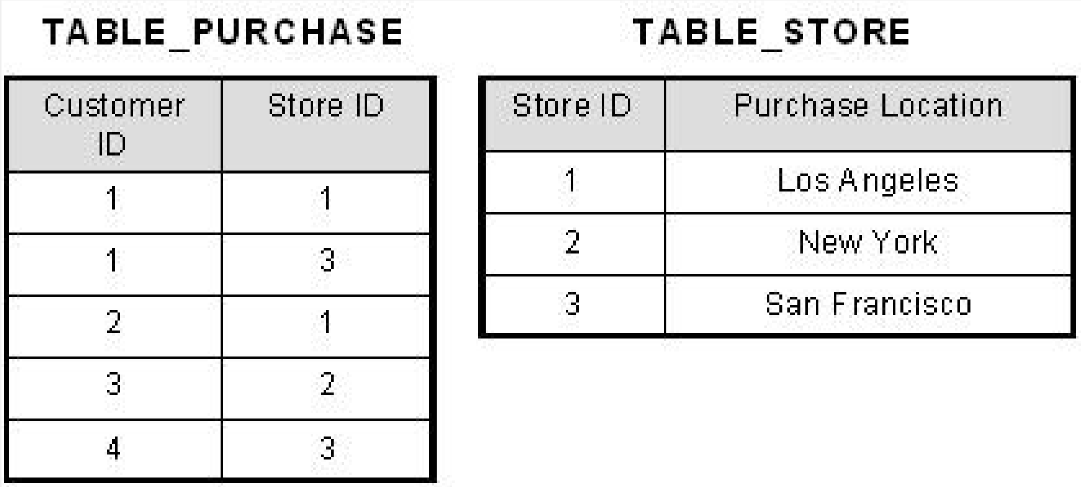
1NF



TABLE_PURCHASE_DETAIL

Customer ID	Store ID	Purchase Location
1	1	Los Angeles
1	3	San Francisco
2	1	Los Angeles
3	2	New York
4	3	San Francisco

[Purchase Location] only depends on [Store ID]



TABLE_PURCHASE

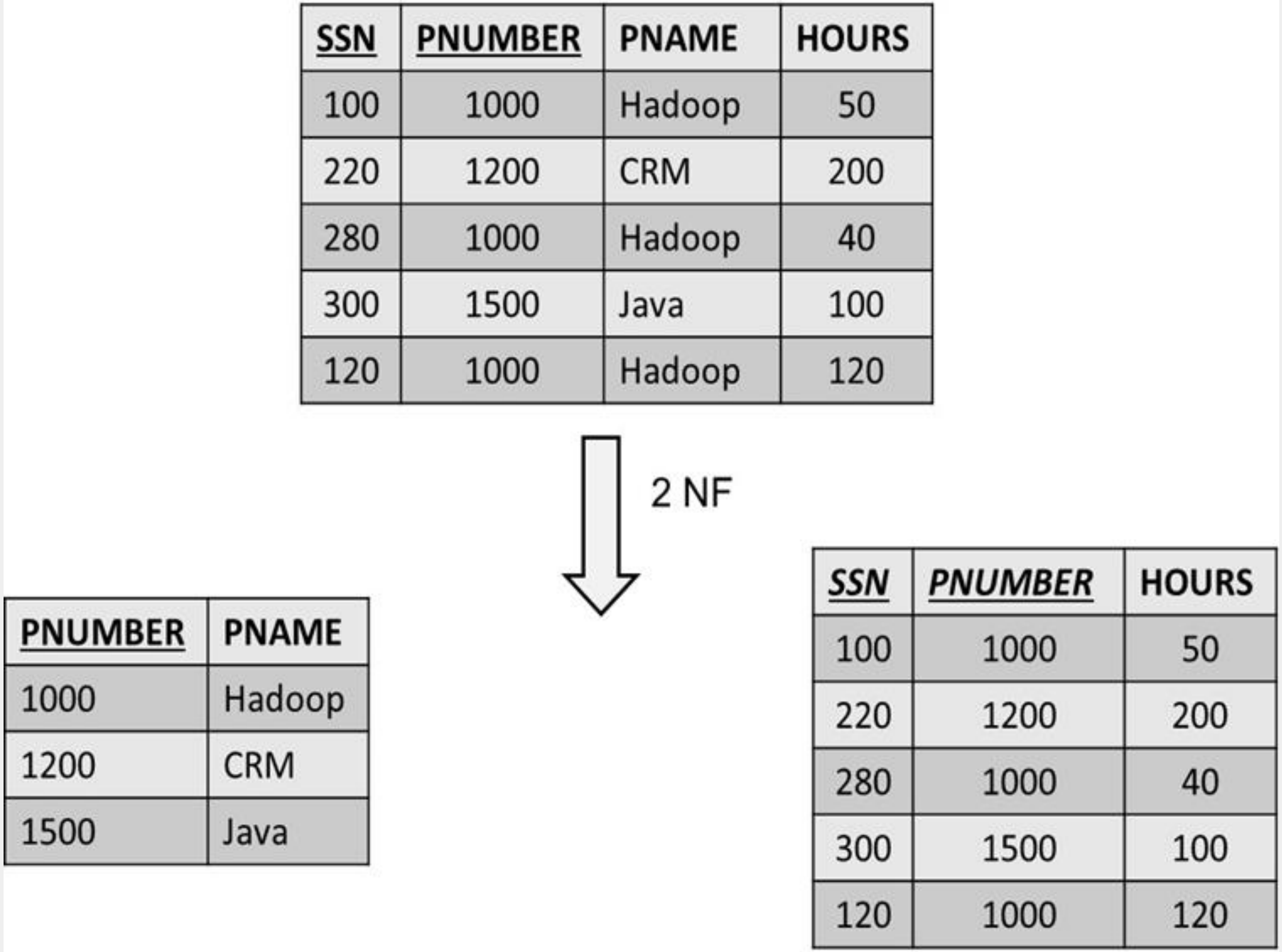
Customer ID	Store ID
1	1
1	3
2	1
3	2
4	3

TABLE_STORE

Store ID	Purchase Location
1	Los Angeles
2	New York
3	San Francisco

2NF

Less Simple Example



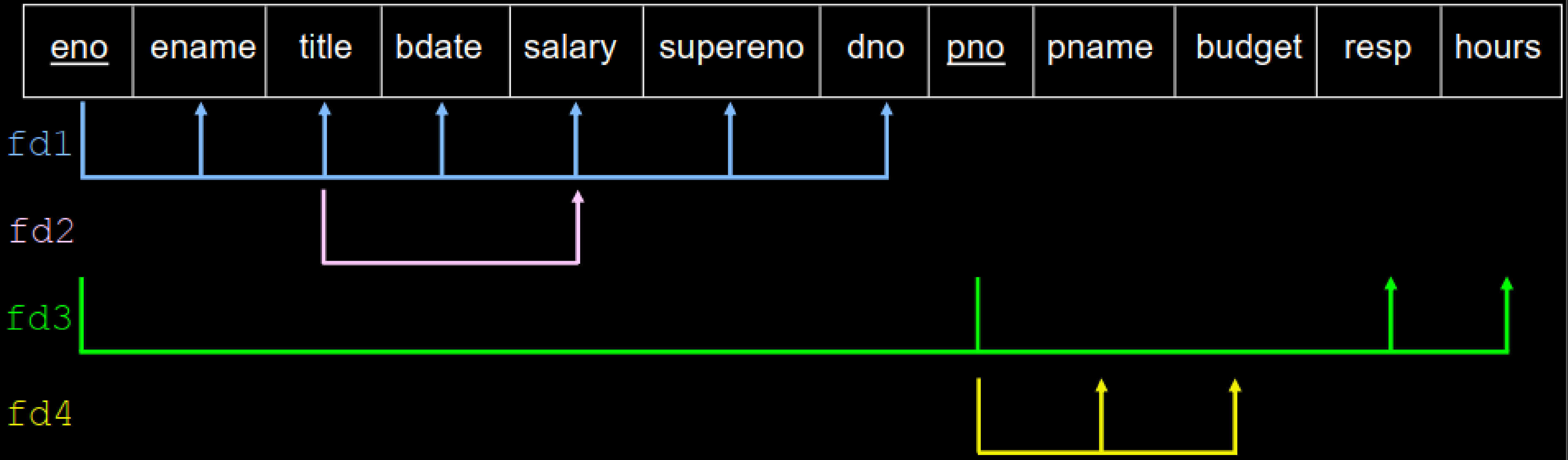
SSN	PNUMBER	PNAME	HOURS
100	1000	Hadoop	50
220	1200	CRM	200
280	1000	Hadoop	40
300	1500	Java	100
120	1000	Hadoop	120

PNUMBER	PNAME
1000	Hadoop
1200	CRM
1500	Java

SSN	*PNUMBER*	HOURS
100	1000	50
220	1200	200
280	1000	40
300	1500	100
120	1000	120

Not So Simple Example

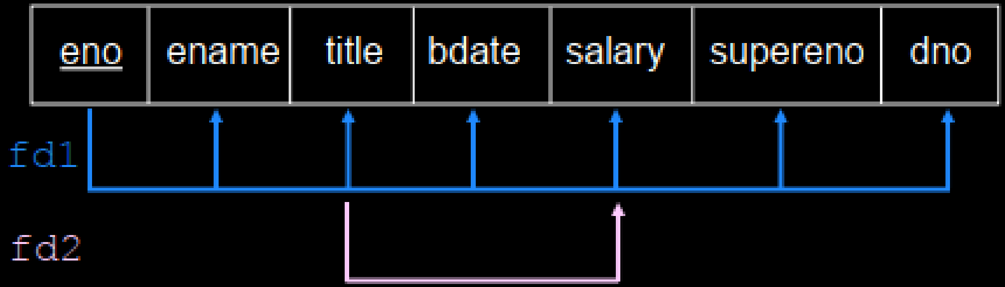
EmpProj relation:



eno	ename	title	bdate	salary	supereno	dno	pno	pname	budget	resp	hours

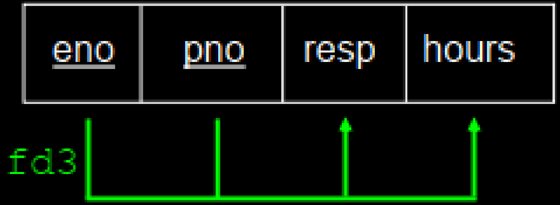
- Emp (eno, ename, title, bdate, salary, supereno, dno)
- WorksOn (eno, pno, resp, hours)
- Proj (pno, pname, budget)

Emp relation:



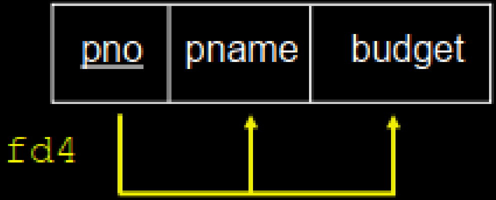
eno	ename	title	bdate	salary	supereno	dno

WorksOn relation:



eno	pno	resp	hours

Proj relation:



pno	pname	budget

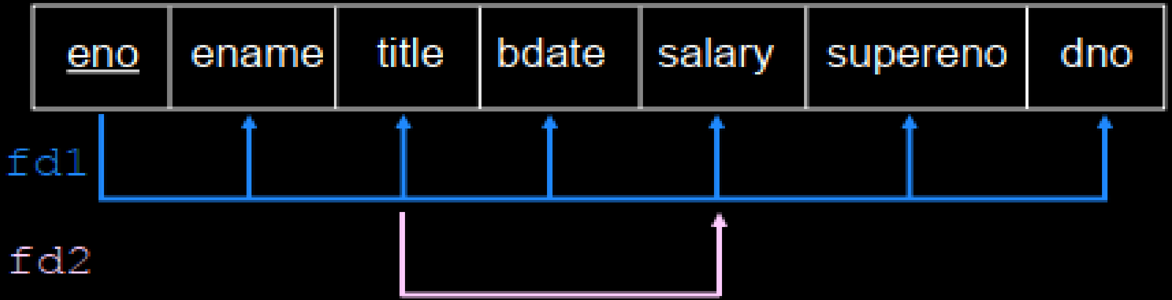
Third Normal Form (3NF)

Functional Dependency Rules :

Transitive Rule: If $A \rightarrow B$ and $B \rightarrow C$ then $A \rightarrow C$

A relation is in the third normal form (3 NF) if it satisfies 2 NF and no non-prime attribute type of R is transitively dependent on the primary key.

Emp relation:



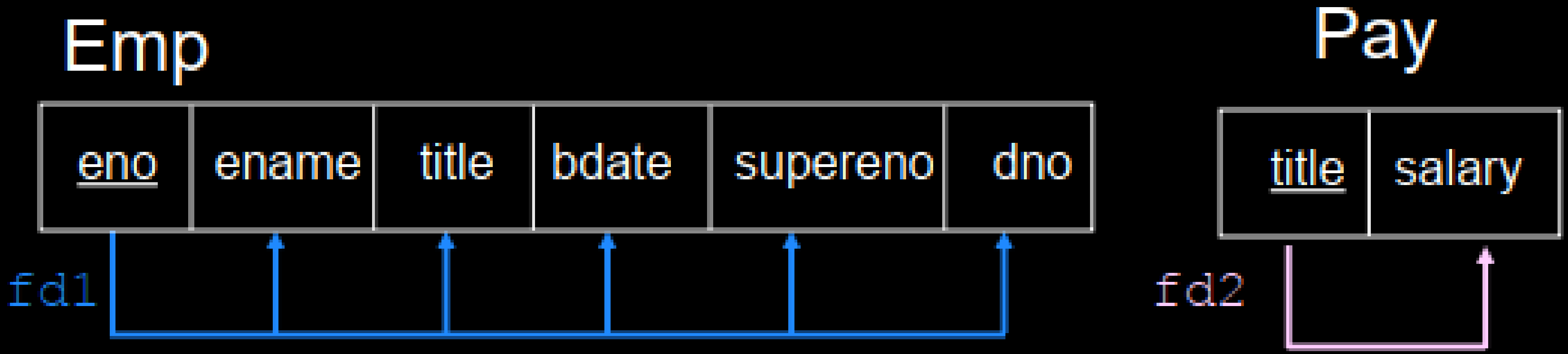
fd2 results in a transitive dependency

eno → salary

Third Normal Form (3NF)

If the relation is not in third normal form

Decompose the relation R and set up a relation that includes the non-key attributes **that functionally determine** the other non-key attributes



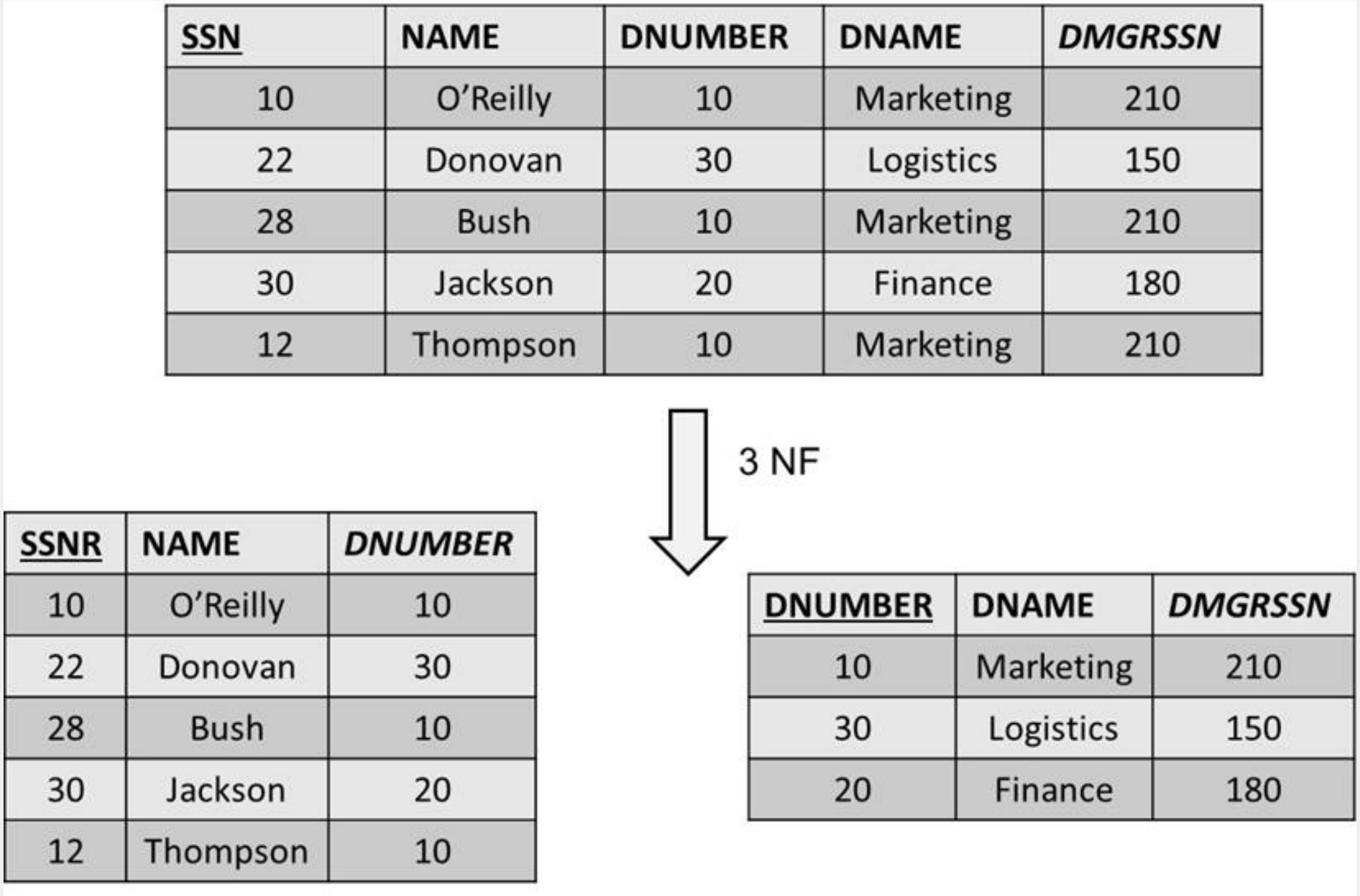
Emp

eno	ename	title	bdate	supereno	dno

Pay

title	salary

Another Example



SSN	NAME	DNUMBER	DNAME	*DMGRSSN*
10	O’Reilly	10	Marketing	210
22	Donovan	30	Logistics	150
28	Bush	10	Marketing	210
30	Jackson	20	Finance	180
12	Thompson	10	Marketing	210

3 NF

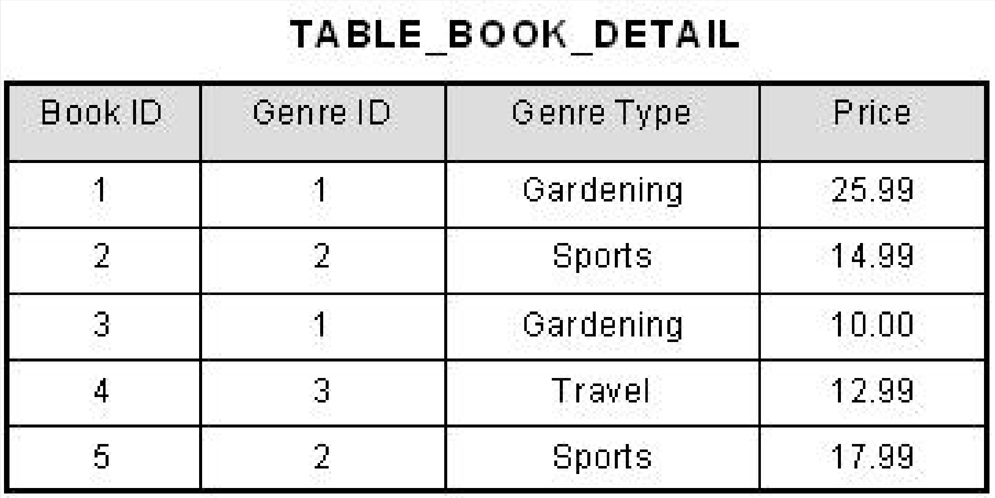
SSNR	NAME	*DNUMBER*
10	O’Reilly	10
22	Donovan	30
28	Bush	10
30	Jackson	20
12	Thompson	10

DNUMBER	DNAME	*DMGRSSN*
10	Marketing	210
30	Logistics	150
20	Finance	180

Example

2NF ?

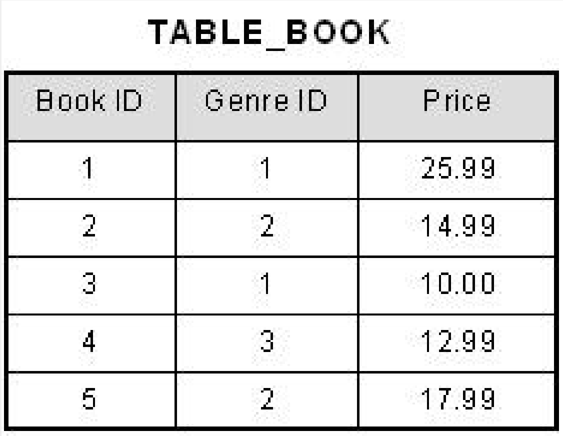
[Book ID] determines [Genre ID]
[Genre ID] determines [Genre Type]



TABLE_BOOK_DETAIL

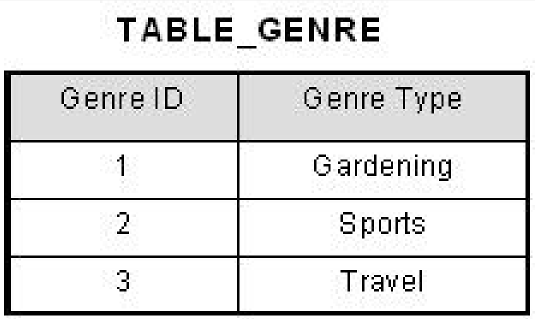
Book ID	Genre ID	Genre Type	Price
1	1	Gardening	25.99
2	2	Sports	14.99
3	1	Gardening	10.00
4	3	Travel	12.99
5	2	Sports	17.99

3NF



TABLE_BOOK

Book ID	Genre ID	Price
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99



TABLE_GENRE

Genre ID	Genre Type
1	Gardening
2	Sports
3	Travel