Normalization

Khalid Mahmood
Khalid.mahmood@it.uu.se

Issues with DBMS Design

- Data integrity failure, e.g., key does not exists

Customer_ID	Customer_Date_Of_Birth
1001	17-FEB-1994
1001	06-AUG-1988

Not a PK

Normalization

- The process of breaking "bad" relations/tables smaller ”good” relations → leads to a better DB design
- *Avoids **update**, **delete** and **insert** anomalies*
- **Normalization** process involves achieving different **n*ormal forms*** *step by step:*
  - - 1NF (First Normal Form)
  - - 2NF (Second Normal Form)
  - - 3NF (Third Normal Form)
  - - BCNF (Boyce-Codd Normal Form) [not going to cover]
  - …. (and more)

The more normal form we process, the better the relational database we produce.

Normalization

- 2 Tools for achieving 3 different Normal Forms:

  (1) Key: what is the key/PK of a relation?

  (2) Functional Dependency

Functional Dependency

- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y
  - Simple saying, X determines Y; written as *X->Y*
  - Possible FD: Text → Course, [what is special about *Text*?]
  - Not FDs : Teacher → Course, Teacher → Text, Couse → Text
  - What about : Text→ Teacher?

TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

Functional Dependency

- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y
  - Simple saying, X determines Y; written as *X->Y*
  - Possible FD: Text → Course, [what is special about *Text*?] PK forms FD
  - Not FDs : Teacher → Course, Teacher → Text, Couse → Text
  - What about : Text→ Teacher? *Not possible*

TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

Task

- Which FDs may exist in this relation?

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

Task:

- Which FDs may exist in this relation?
  - B→C, C→B

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

1NF (1st Normal Form)

- **Composite Attributes** or **Attributes with different data-types**) is not allowed in a relation

Beatle_Height

Beatle	Height_In_Cm (integer)
George	178
John	179
Ringo	Somewhere between 168 and 171
Paul	180

Mixing data types within the same column violates INF

(and the DB platform won't let you do it anyway)

1NF (1st Normal Form)

- A relation must have a **primary key**

We should make this the primary key...

Beatle_Height

Beatle	Height_In_Cm
George	178
John	179
Ringo	170
Paul	180

```
ALTER TABLE Beatle_Height
ADD PRIMARY KEY (Beatle);
```

Making "Beatle" the primary key prevents this:

Beatle	Height_In_Cm
George	178
John	179
John	186
Ringo	170
Paul	180

1NF (1st Normal Form)

- **Nested relations** is not allowed

(a)

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours

(b)

EMP_PROJ

Ssn	Ename	Pnumber	Hours
123456789	Smith, John B.	1	32.5
		2	7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1	20.0
		2	20.0
333445555	Wong, Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0
999887777	Zelaya, Alicia J.	30	30.0
		10	10.0
987987987	Jabbar, Ahmad V.	10	35.0
		30	5.0
987654321	Wallace, Jennifer S.	30	20.0
		20	15.0
888665555	Borg, James E.	20	NULL

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

1st normal form

- **Multivalued Attribute** is not allowed

(a)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations

(b)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

1NF Summery

- Must have a PK
- No Composite Attribute
- No Multivalued Attribute
- No Nested Relationship

Issues with DBMS Design

- Redundant Information in Tuples

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples: ***Delete Anomaly***
  - Example: what if we delete *'gilal9'?*

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gilal9	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples: ***Delete Anomaly***
  - Example: what if we delete *'gilal9'* ?
    - *The Player_Rating of gilal9 (i.e., Beginner) will be deleted, which is not the case for other players.*

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
** deletion anomaly **			
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples: ***Update Anomaly***
  - Example: what if we change *Player_Rating* of *'jdog21'?*

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples: ***Update Anomaly***
  - Example: what if we change *Player_Rating* of *'jdog21'?*
    - *The Player_Rating might display two different ratings.*

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Advanced
jdog21	rings	4	Intermediate
gila19	copper coins	18	
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

** update anomaly **

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples: ***Insert Anomaly***
  - Example: what if we insert '*tine42*' with *Player_Rating* = *Beginner* ?

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples: ***Insert Anomaly***
  - Example: what if we insert '*tine42*' with *Player_Rating* = *Beginner* ?
    - Not possible as part of PK (*Item_Type)* is missing

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Composite PK = (Player_ID, Item_Type)

Issues with DBMS Design

- Redundant Information in Tuples are quite common.

Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Redundancy Redundancy

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

2NF (2nd Normal Form)

- 1NF + Each **non-key attribute** must depended on the **entire primary key**

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

2NF

- Each **non-key attribute** must depended on the **entire primary key:** avoid Partial Dependencies

Composite PK = (Player_ID, Item_Type)

Non-Key Attribute:
- Item_Quantity
- Player_Rating

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

Functional Dependencies:

{Player_ID, Item_Type} → {Item_Quantity} ✅ *(in 2NF)*

{Player_ID} → {Player_Rating} ❌ *(not in 2NF)*

2NF

- How to achieve 2NF?
  - Draw the *Functional Dependencies*
    - *PK in this relation is {Ssn, Pnumber}*

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation

2NF

- How to achieve 2NF?
  - Draw the *Functional Dependencies*
    - *PK in this relation is {Ssn, Pnumber}*

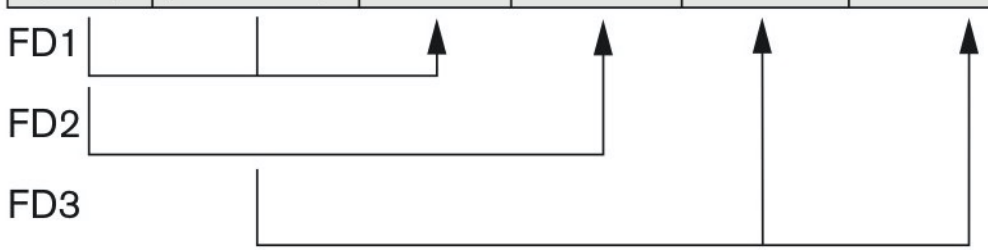
EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation

Functional Dependencies

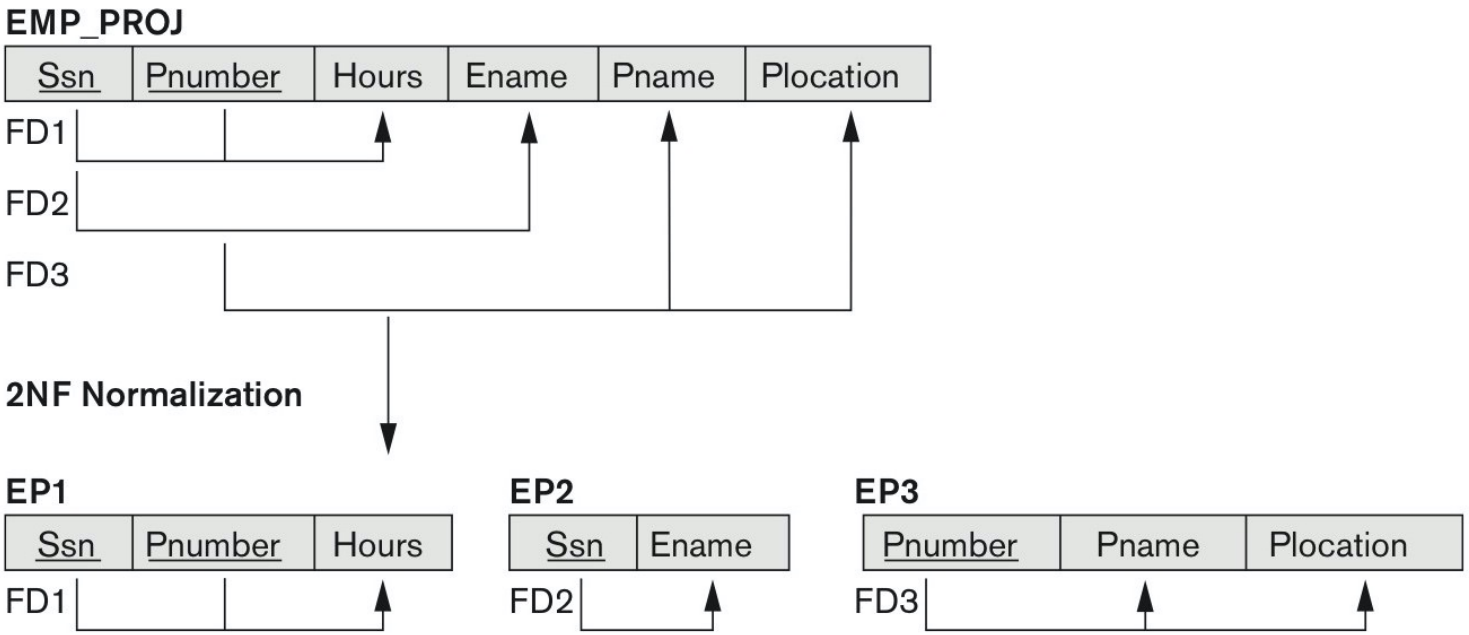
EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation



2NF

- How to achieve 2NF?
  - Draw the *Functional Dependencies*
    - *PK in this relation is {Ssn, Pnumber}*
  - Decompose it into different relations



2NF

- 1NF +
- Each **non-key attribute** must depended on the **entire primary key**
  - ***Partial Dependencies*** of PK is not allowed

3NF (3rd Normal Form)

- Is the table in 2NF?
- What if we UPDATE *Player_Skill_Level* of **jdog21** from **4** to **8**?

Player

Player_ID	Player_Rating	Player_Skill_Level
jdog21	Intermediate	4
gila19	Beginner	3
trev73	Advanced	8
tina42	Beginner	1

SKILL LEVEL

1 2 3	4 5 6	7 8 9
Rating: Beginner	Rating: Intermediate	Rating: Advanced

3NF (3rd Normal Form)

- Is the table in 2NF?
- What if we UPDATE *Player_Skill_Level* of **jdog21** from **4** to **8**?

Player

Player_ID	Player_Rating	Player_Skill_Level
jdog21	Intermediate	8
gila19	Beginner	3
trev73	Advanced	8
tina42	Beginner	1

Will update to “Advanced”

SKILL LEVEL

1 2 3	4 5 6	7 8 9
Rating: Beginner	Rating: Intermediate	Rating: Advanced

3NF (3rd Normal Form)

- 2NF + Every **non-key attribute** depends on ***only the Key***
  - Avoid *Transitive Dependency*

Player

Player_ID	Player_Rating	Player_Skill_Level
jdog21	Intermediate	4
gila19	Beginner	3
trev73	Advanced	8
tina42	Beginner	1

SKILL LEVEL

1 2 3	4 5 6	7 8 9
Rating: Beginner	Rating: Intermediate	Rating: Advanced

3NF (3rd Normal Form)

- 2NF + Every **non-key attribute** depends on ***only the Key***
  - Avoid *Transitive Dependency*

Player

Player_ID	Player_Rating	Player_Skill_Level
jdog21	Intermediate	4
gila19	Beginner	3
trev73	Advanced	8
tina42	Beginner	1

SKILL LEVEL

1 2 3	4 5 6	7 8 9
Rating: Beginner	Rating: Intermediate	Rating: Advanced

Functional Dependencies:

{Player_ID} → {Player_Rating}

{Player_ID} → {Player_Skill_Level} → {Player_Rating} ✗ *(not in 3NF)*

Transitive Dependency

3NF

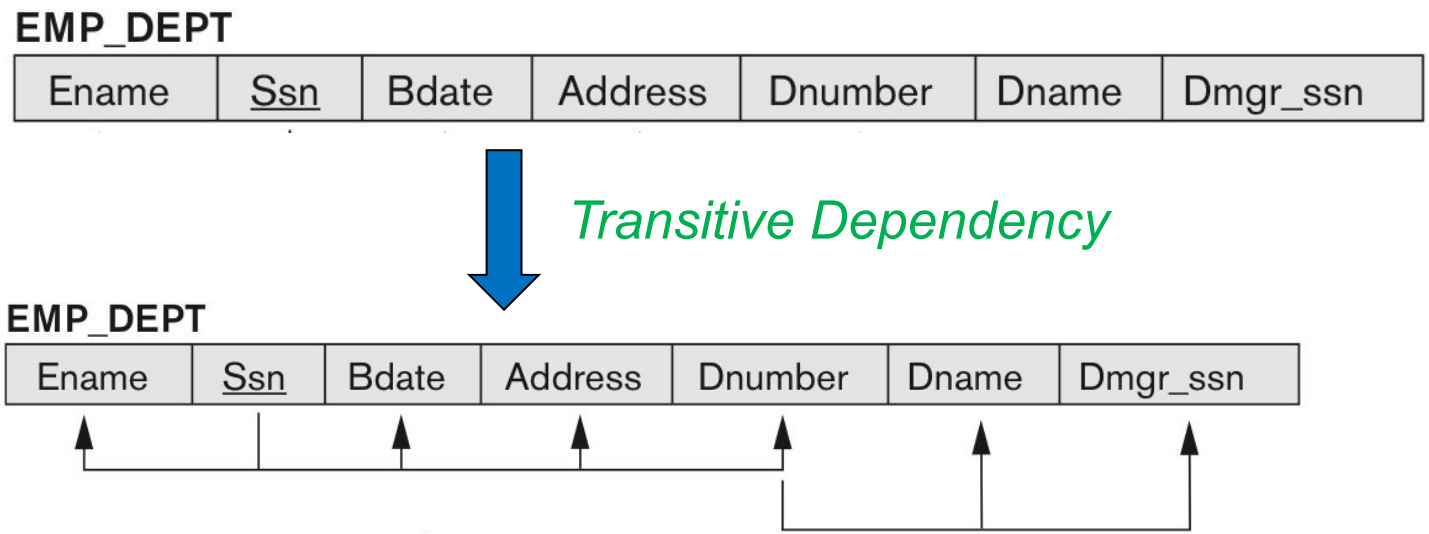
- How to achieve 3NF?
  - Draw *Functional Dependencies and find the Transitive Dependencies*

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn

3NF

- How to achieve 3NF?
  - Draw *Functional Dependencies and find the Transitive Dependencies*



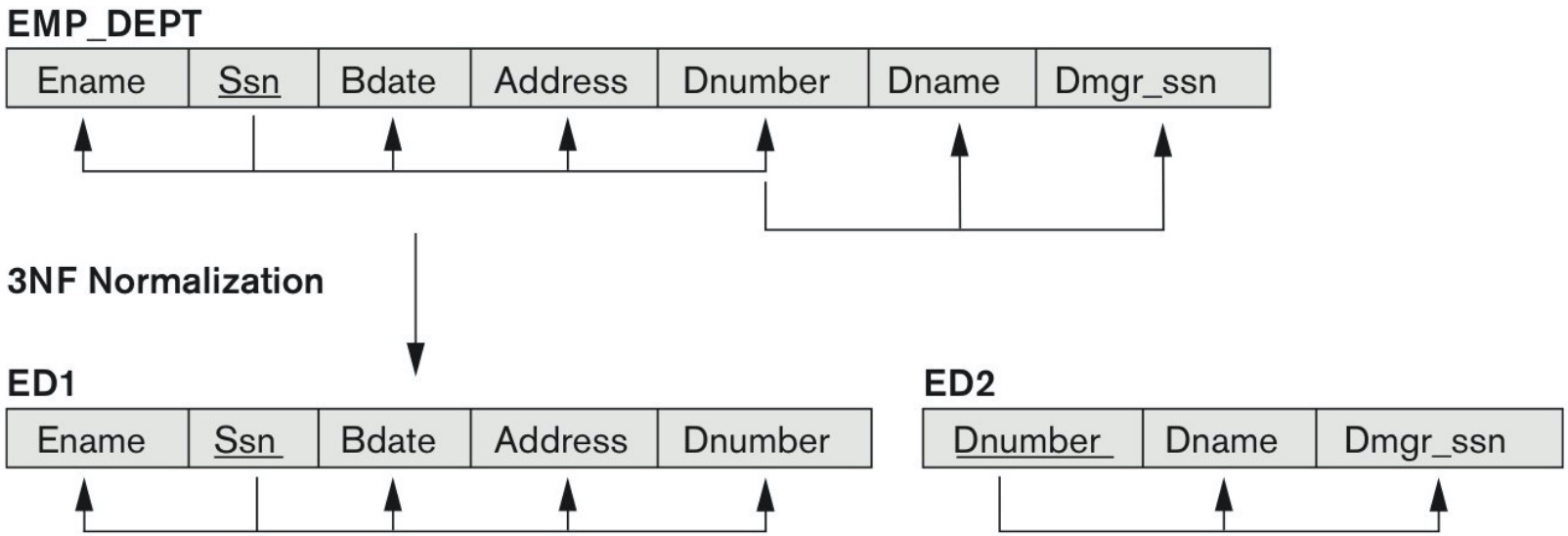
Functional Dependencies:

Ssn → Ename, ***Ssn → Dnumber****, Ssn → Bdate , Ssn → Address*

Dnumber→ Dname, Dnumber→ Dmgr_ssn ; => Transitive Dependencies

3NF

- How to achieve 3NF?
  - Draw *Functional Dependencies and find the Transitive Dependencies*
  - Decompose *Transitive dependency* into different relations



3NF

- 2NF +
- Every **non-key attribute** depends on ***only the Key***: Avoid *Transitive Dependency*

1NF & 2NF & 3NF

- 1NF: All attributes depend on **the key**
- 2NF: All attributes depend on **the whole key**
- 3NF: All attributes depend on **nothing but the key**

Formal Normalization Jargons

- ***Candidate Key**:* an attribute (or multiple attributes) uniquely identifies a row
- ***Prime Attribute:*** an attribute belongs to at least one candidate key
- ***Non-prime Attribute**:* an attribute that doesn't belong to any candidate key

Player_Inventory

Player_ID	Item_Type	Item_Quantity	Player_Rating
jdog21	amulets	2	Intermediate
jdog21	rings	4	Intermediate
gila19	copper coins	18	Beginner
trev73	shields	3	Advanced
trev73	arrows	5	Advanced
trev73	copper coins	30	Advanced
trev73	rings	7	Advanced

- **Candidate key**: {*Player_ID, Item_Type*}
- **Prime-Attributes**: (1) *Player_ID* (2) *Item_Type*
- **Non-prime attributes:** (1) *Item_Quantity* (2) *Player_Rating*

Task: Understand the formal definitions of 2NF and 3NF

- ***Definition of 2NF:*** A relation schema R is in **second normal form (2NF)** if every ***non-prime attribute*** A in R is ***fully functionally dependent*** on the primary key

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

Exercise: *Ex-4-Normalisation-Steps.pdf*

Full slide from book authors:

Chapter14.pdf

Acknowledgements

- Pictures are taken from:
  - *Learn Database Normalization - 1NF, 2NF, 3NF, 4NF, 5NF* | Youtube Video
  - *Chapter14.pdf* of our text book