Normalization

Database Normalization

- **Normalization:** The process of structuring data to minimize duplication and inconsistencies.
- The process usually involves breaking down a single Table into two or more tables and defining relationships between those tables.
- Normalization is usually done in stages, with each stage applying some rules to the types of information which can be stored in a table.

Well-Structured Relations

- Goal is to avoid anomalies
  - **Insertion Anomaly** – adding new rows forces user to create duplicate data
  - **Deletion Anomaly** – deleting rows may cause a loss of data that would be needed for other future rows
  - **Modification Anomaly** – changing data in a row forces changes to other rows because of duplication

a table should not have more than one entity type

Example

<u>SID</u>	Sname	Bdate	City	ZipCode	<u>Subject</u>	Grade	Teacher
1	Ahmed	1/1/1980	Cairo	1010	DB	A	Hany
1	Ahmed	1/1/1980	Cairo	1010	Math	B	Eman
1	Ahmed	1/1/1980	Cairo	1010	WinXP	A	khalid
2	Ali	1/1/1983	Alex	1111	DB	B	Hany
2	Ali	1/1/1983	Alex	1111	SWE	B	Heba
3	Mohamed	1/1/1990	Mansoura	1210	NC	C	Mona

Question – What's the primary key? Answer – Composite: SID, Subject

Why do these anomalies exist?

Because we've combined two themes (entity types) into one relation. This results in duplication, and an unnecessary dependency between the entities

Functional dependency

- a constraint between two attributes (columns) or two sets of columns
- A → B if "for every valid instance of A, that value of A uniquely determines the value of B"
- or …A →B if "existing of B depending on a value of A"

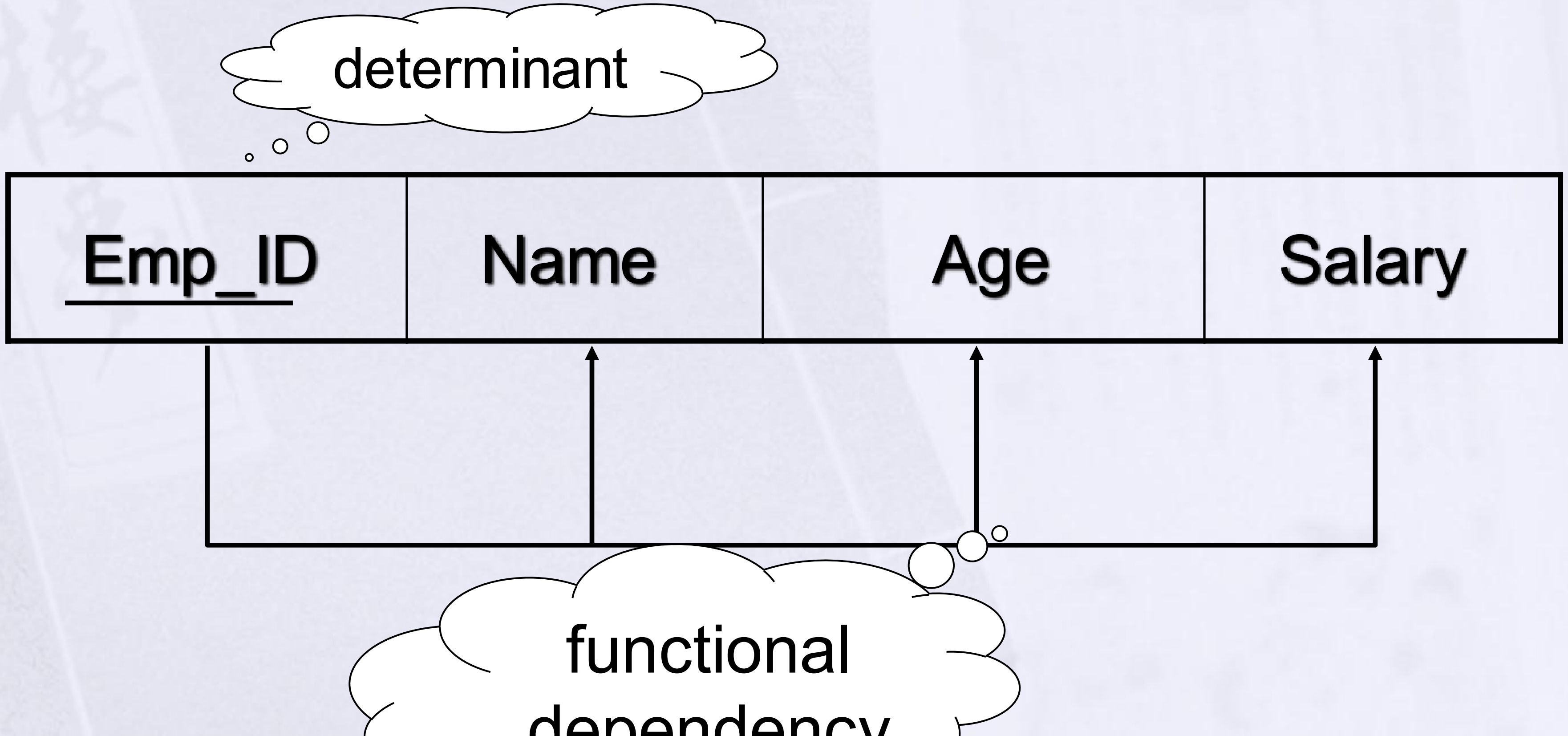
… functional dependency

some examples

- social security number determines employee name
  SSN -> ENAME
- project number determines project name and location
  PNUMBER -> {PNAME, PLOCATION}
- employee ssn and project number determines the hours per week that the employee works on the project
  {SSN, PNUMBER} -> HOURS

keys and dependencies

EMPLOYEE1 (Emp_ID, Name, Age, Salary)



Types of functional dependency

- **Full Functional Dependency**

  Attribute is fully Functional Dependency on a PK if its value is determined by the whole PK

- **Partial Functional Dependency**

  Attribute if has a Partially Functional Dependency on a PK if its value is determined by part of the PK(Composite Key)

- **Transitive Functional Dependency**

  Attribute is Transitively Functional Dependency on a table if its value is determined by anther non-key attribute which it self determined by PK

Example

<u>**SID**</u>	**SName**	**Birthdate**	**City**	**Zip Code**	<u>**Subject**</u>	**Grade**	**Teacher**
1	Ahmed	1/1/1980	Cairo	1010	DB	A	Hany
1	Ahmed	1/1/1980	Cairo	1010	Math	B	Eman
1	Ahmed	1/1/1980	Cairo	1010	WinXP	A	khalid
2	Ali	1/1/1983	Alex	1111	DB	B	Hany
2	Ali	1/1/1983	Alex	1111	SWE	B	Heba
3	Mohamed	1/1/1990	Cairo	1010	NC	C	Mona

Full Functional Dependency	Sid,Subject → Grade
Partial Functional Dependency	Sid → SName Subject →Teacher
Transitive Functional Dependency	ZipCode → City

Steps in normalization

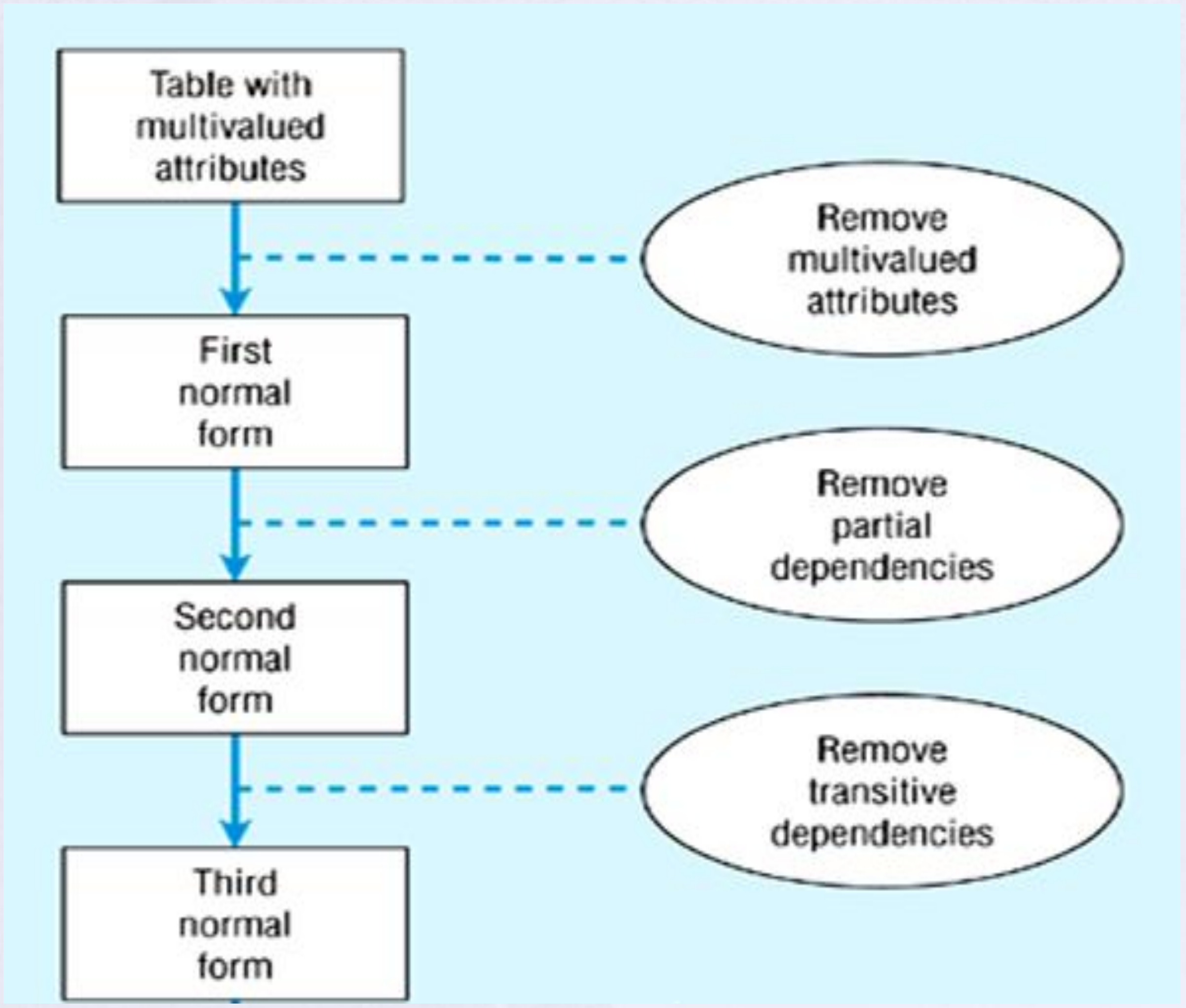


Table with multivalued attributes
Remove multivalued attributes
First normal form
Remove partial dependencies
Second normal form
Remove transitive dependencies
Third normal form

1NF

- relation is in **first normal form** if it contains no multivalued or composite attributes
- remove repeating groups to a new table as already demonstrated, “carrying” the PK as a FK
- All columns (fields) must be atomic
  - Means : no repeating items in columns

Example

<u>SID</u>	SName	Birthdate	City	Zip Code	<u>Subject</u>	Grade	Teacher
1	Ahmed	1/1/1980	Cairo	1010	DB	A	Hany
1	Ahmed	1/1/1980	Cairo	1010	Math	B	Eman
1	Ahmed	1/1/1980	Cairo	1010	WinXP	A	khalid
2	Ali	1/1/1983	Alex	1111	DB	B	Hany
2	Ali	1/1/1983	Alex	1111	SWE	B	Heba
3	Mohamed	1/1/1990	Cairo	1010	NC	C	Mona

<u>SID</u>	SName	Birthdate	City	Zip Code	<u>Subject</u>	Grade	Teacher
1	Ahmed	1/1/1980	Cairo	1010	DB	A	Hany
					Math	B	Eman
					WinXP	A	khalid
2	Ali	1/1/1983	Alex	1111	DB	B	Hany
					SWE	B	Heba
3	Mohamed	1/1/1990	Cairo	1010	NC	C	Mona

Repeating Groups
Or multivalued

1NF

Student(<u>SID</u>, Sname, Birthdate, City, Zip Code)

<u>SID</u>	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Cairo	1010

Stud_Subject (<u>SID, Subject</u>, Grade, Teacher)

<u>SID</u>	<u>Subject</u>	Grade	Teacher
1	DB	A	Hany
1	Math	B	Eman
1	WinXP	A	khalid
2	DB	B	Hany
2	SWE	B	Heba
3	NC	C	Mona

2NF

- a relation is in **second normal form** if it is in first normal form AND every nonkey attribute is fully functionally dependant on the primary key
- i.e. remove partial functional dependencies, so no nonkey attribute depends on just part of the key

2NF

Student(<u>SID</u>, Sname, Birthdate, City, Zip Code)

<u>SID</u>	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Cairo	1010

2NF
Because there is no
Composite PK

Stud_Subject (<u>SID, Subject</u>, Grade, Teacher)

<u>SID</u>	<u>Subject</u>	Grade	Teacher
1	DB	A	Hany
1	Math	B	Eman
1	WinXP	A	khalid
2	DB	B	Hany
2	SWE	B	Heba
3	NC	C	Mona

SID, Subject → Grade……FFD

Subject → Teacher……PFD

2NF

Student(SID, Sname, Birthdate, City, Zip Code)

SID	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Mansoura	1210

Stud_Subject (SID, Subject, Grade)

SID	Subject	Grade
1	DB	A
1	Math	B
1	WinXP	A
2	DB	B
2	SWE	B
3	NC	C

Subject (Subject,Teacher)

Subject	Teacher
DB	Hany
Math	Eman
WinXP	khalid
SWE	Heba
NC	Mona

Third Normal Form

- 2NF PLUS ***no transitive dependencies*** (one attribute functionally determines a second, which functionally determines a third)

2NF

Student(SID, Sname, Birthdate, City, Zip Code)

SID	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Cairo	1010

Zip Code ->City …….TFD

Stud_Subject (SID, Subject, Grade)

SID	Subject	Grade
1	DB	A
1	Math	B
1	WinXP	A
2	DB	B
2	SWE	B
3	NC	C

Subject (Subject,Teacher)

Subject	Teacher
DB	Hany
Math	Eman
WinXP	khalid
SWE	Heba
NC	Mona

3NF

Because there is no Transtive Functional Dependency

3NF

Student(<u>SID</u>, Sname, Birthdate,)

<u>SID</u>	SName	Birthdate	ZipCode
1	Ahmed	1/1/1980	1010
2	Ali	1/1/1983	1111
3	Mohamed	1/1/1990	1010

Stud_City(City, <u>Zip Code</u>)

City	<u>Zip Code</u>
Cairo	1010
Alex	1111

Stud_Subject (<u>SID, Subject</u>, Grade)

<u>SID</u>	<u>Subject</u>	Grade
1	DB	A
1	Math	B
1	WinXP	A
2	DB	B
2	SWE	B
3	NC	C

Subject (<u>Subject</u>,Teacher)

<u>Subject</u>	Teacher
DB1	Hany
Math	Eman
WinXP	khalid
DB2	Hany
SWE	Heba
NC	Mona

ITI Example

ITI Students Sheet

Platform Name : SWE **Platform Description:** Software Engineering

Graduate Manager: Dr. Baha

Appno	Name	F-code	Faculty	Address	Telno	Grade	Att. Hrs	Sdate
123	Ahmed	SC-phy	Science	Haram	3386842	A	600	14 Sep
124	Mona	Eng-cs	Engineering	Dokki	3389745, 3389744, 5123445	B	591	15 Sep
127	Ali	Com-ac	Commerce	Nasr City	2241593, 2222345	A	550	21 Sep
223	Karim	Med-bio	Medicine	Sheraton	2286845	C	600	14 Sep

1NF :

- **Platform** :<u>pfname</u> , pfdesc , pfManager
- **Students_pf**: <u>pfname, appno</u>, name , faculty , F-Code, address, grade, attd , start_date
- **Std_Tel**: <u>appno, telno</u>

2NF

- **Students**: appno, name , faculty , FCode, address
- **Students_pf**: pfname,appno, grade, attd , start_date

Unchanged Tables

- **Platform** :pfname , pfdesc , pfManager
- **Std_Tel**: appno, telno

3NF

- **Students**: appno, name , FCode, address
- **Fac_majors**:faculty , FCode

Unchanged Tables

- **Platform** :pfname , pfdesc , pfManager
- **Std_Tel**: appno, telno
- **Students_pf**: pfname,appno, grade, attd , start_date

Thank You !!!