Normalization
(1NF, 2NF, 3NF)

USTF: Shi Ruolan
120090757@link.cuhk.edu.cn

CONTENTS

Why do we need normalization?

- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations.
- It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.
- Normalization divides the larger table into smaller and links them using relationships.

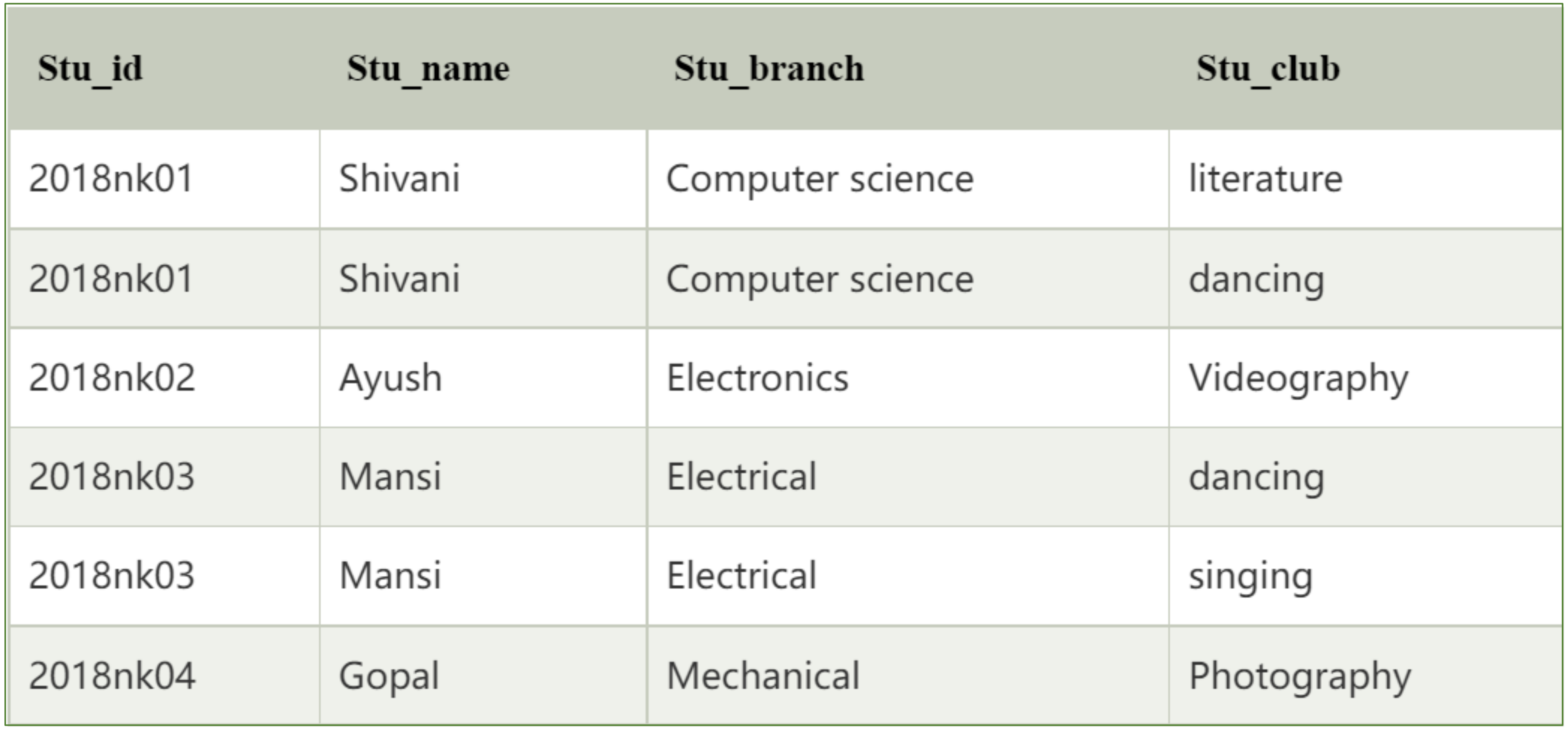
Data modification anomalies （3 types)

Insertion Anomaly: Insertion Anomaly refers to when one cannot insert a new tuple into a relationship due to lack of data.

Deletion Anomaly: The delete anomaly refers to the situation where the deletion of data results in the unintended loss of some other important data.

Updatation Anomaly: The update anomaly is when an update of a single data value requires multiple rows of data to be updated.

Data modification anomalies example



Stu_id	Stu_name	Stu_branch	Stu_club
2018nk01	Shivani	Computer science	literature
2018nk01	Shivani	Computer science	dancing
2018nk02	Ayush	Electronics	Videography
2018nk03	Mansi	Electrical	dancing
2018nk03	Mansi	Electrical	singing
2018nk04	Gopal	Mechanical	Photography

Data modification anomalies example

Update Anomaly
If Shivani changes her branch from Computer Science to Electronics, then we will have to update all the rows. If we miss any row, then Shivani will have more than one branch, which will create the update anomaly in the table.

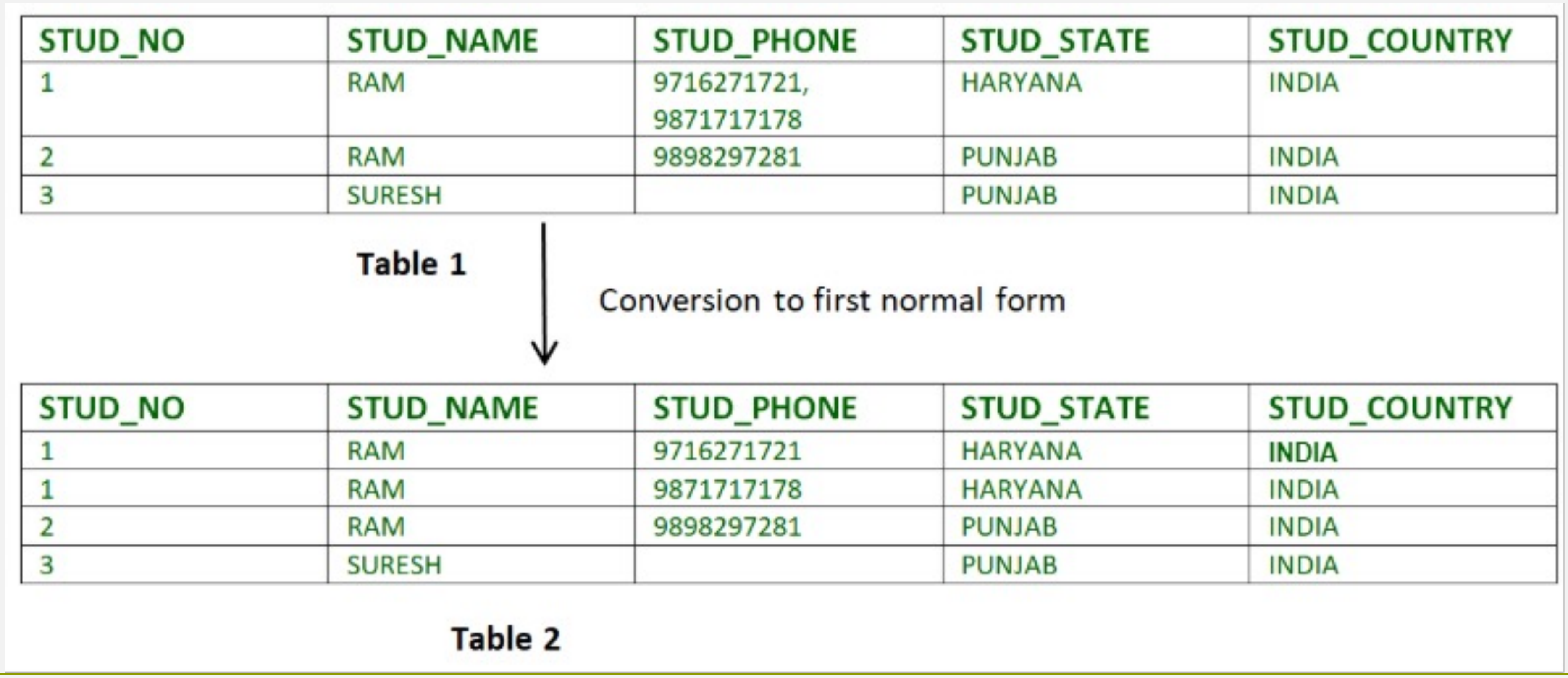
Insertion Anomaly
If we add a new row for student Ankit who is not a part of any club, we cannot insert the row into the table as we cannot insert null in the column of stu_club. This is called insertion anomaly.

Deletion Anomaly
If we remove the photography club from the college, then we will have to delete its row from the table. But it will also delete the table of Gopal and his details. So, this is called deletion anomaly and it will make the database inconsistent.

review: 1NF

- Each **attribute name** must be unique
- Each **attribute value** must be single
- Each **row** must be unique
- There is no **repeating groups**

Atomic



STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721, 9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 1

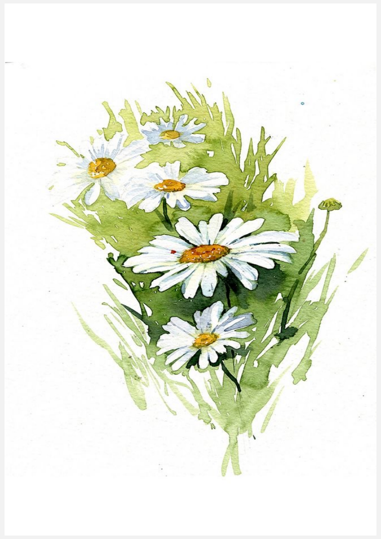
Conversion to first normal form

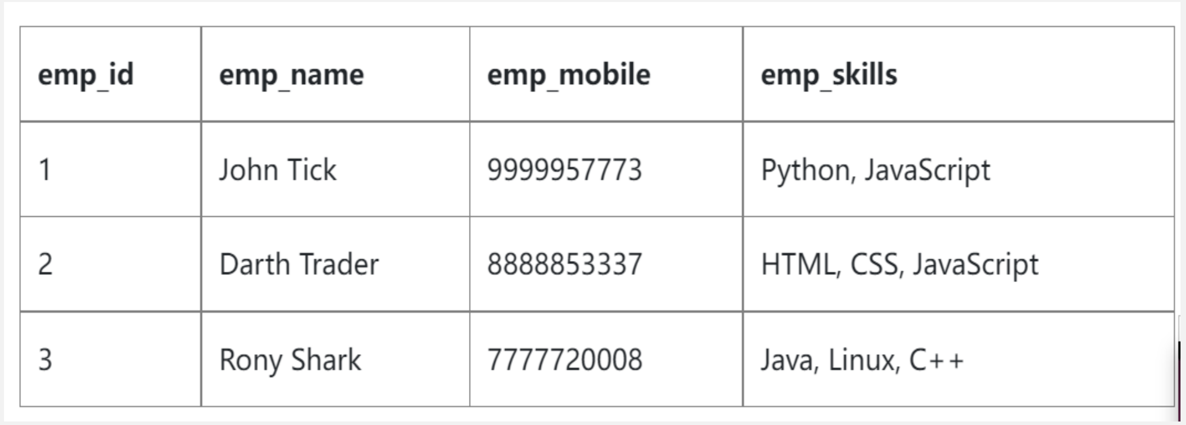
STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721	HARYANA	INDIA
1	RAM	9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 2

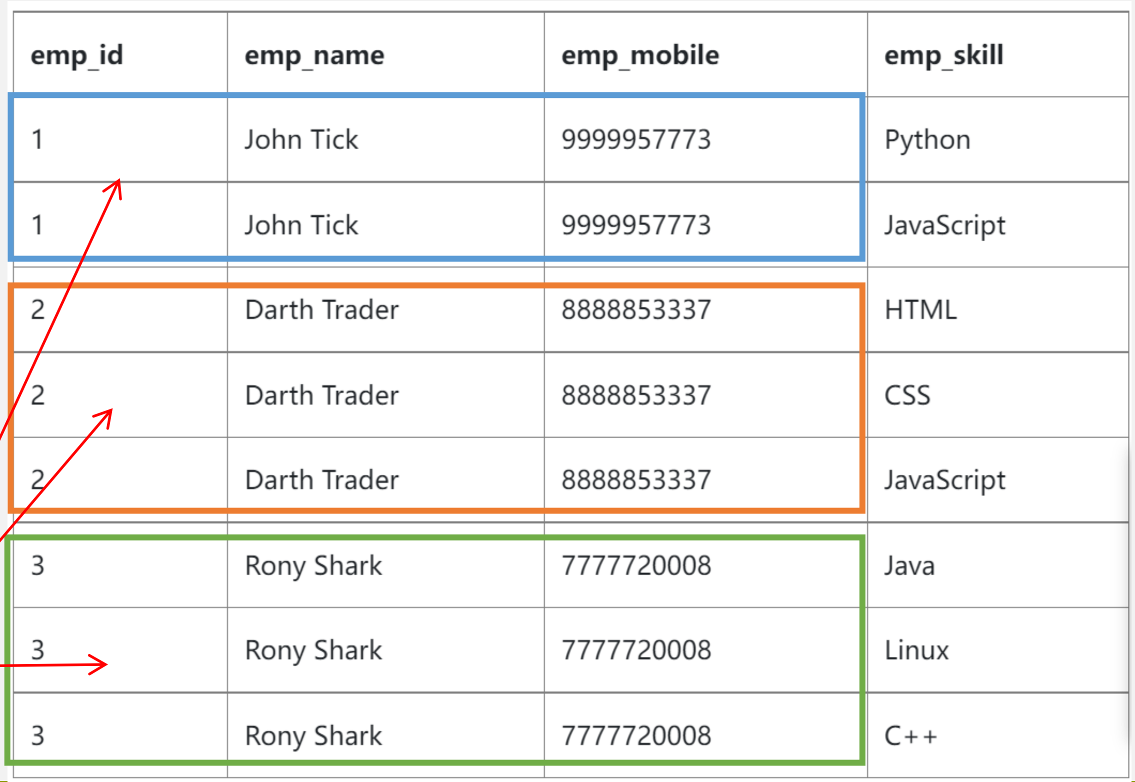
why do we need 2NF

First Normal Form (1NF) does not eliminate redundancy, but rather, it's that it **eliminates repeating groups**. Instead of having multiple columns of the same kind of data in a record, (0NF or Unnormalized form) you remove the repeated information into a separate relation and represent them as rows.





emp_id	emp_name	emp_mobile	emp_skills
1	John Tick	9999957773	Python, JavaScript
2	Darth Trader	8888853337	HTML, CSS, JavaScript
3	Rony Shark	7777720008	Java, Linux, C++



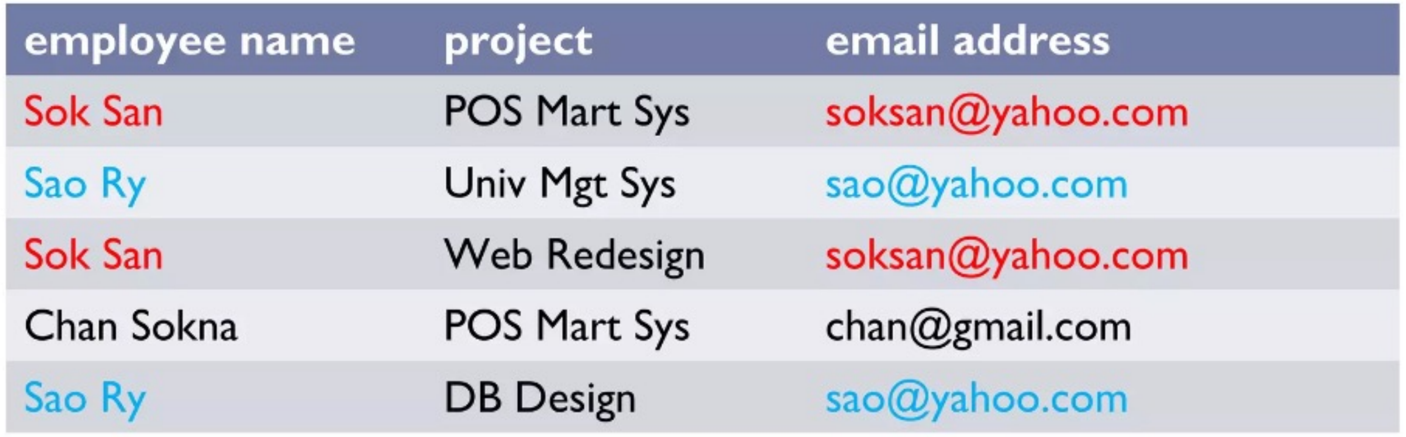
emp_id	emp_name	emp_mobile	emp_skill
1	John Tick	9999957773	Python
1	John Tick	9999957773	JavaScript
2	Darth Trader	8888853337	HTML
2	Darth Trader	8888853337	CSS
2	Darth Trader	8888853337	JavaScript
3	Rony Shark	7777720008	Java
3	Rony Shark	7777720008	Linux
3	Rony Shark	7777720008	C++

redundancy!

Review: Functional Dependencies

We say an atttribute, B, has a functional dependency on another attribute, A, if for any records, which have the same value for A, then the values for B in these two records must be the same.

A -> B (A determines B or B depends on A)



employee name	project	email address
Sok San	POS Mart Sys	soksan@yahoo.com
Sao Ry	Univ Mgt Sys	sao@yahoo.com
Sok San	Web Redesign	soksan@yahoo.com
Chan Sokna	POS Mart Sys	chan@gmail.com
Sao Ry	DB Design	sao@yahoo.com

(Assuming each employee name is unique!)

employee name → email address

Review: Partial Dependency

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.

Example:
we have a relationship like MO->N, M->P, and P->N. In this case, M is alone capable of determining N. It means that N is dependent partially on MO.

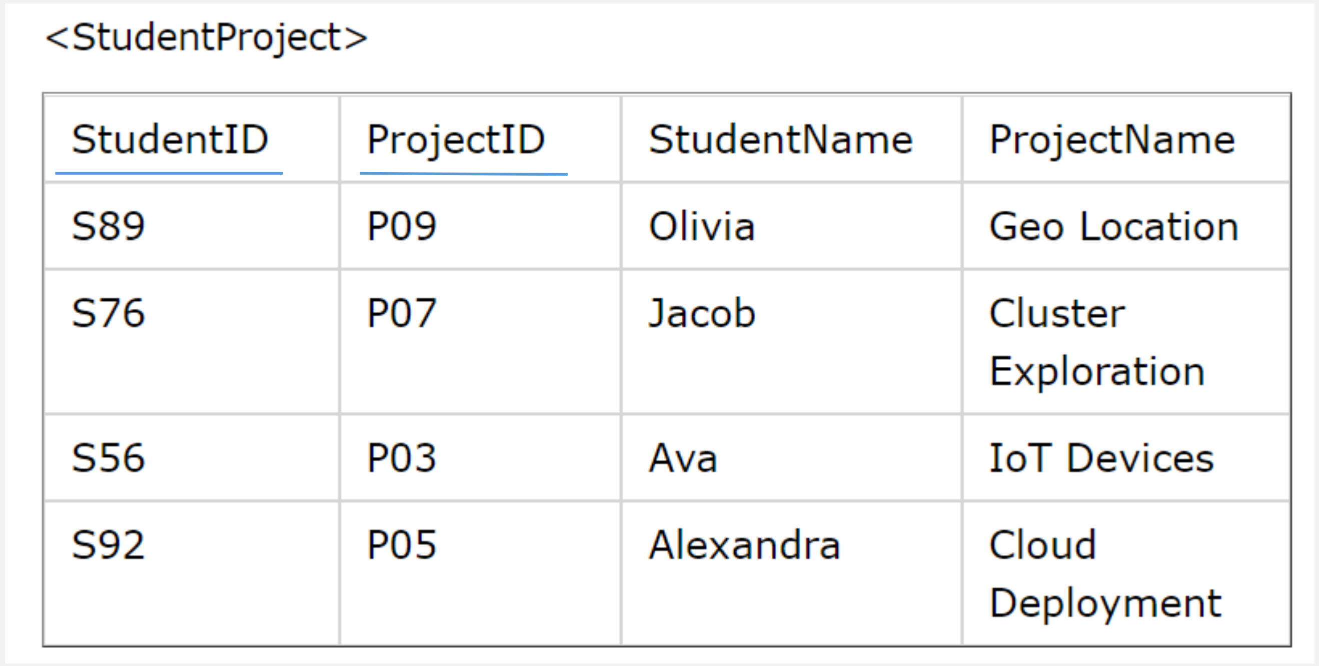
What is 2NF ?

A table is said to be in 2NF if it meets the following criteria:

- it's already **in 1NF**
- has **no partial dependency**. That is, all non-key attributes are **fully dependent** on a primary key.

All partial dependencies are removed to place in another table!

All partial dependencies are removed to place in another table!


<StudentProject>

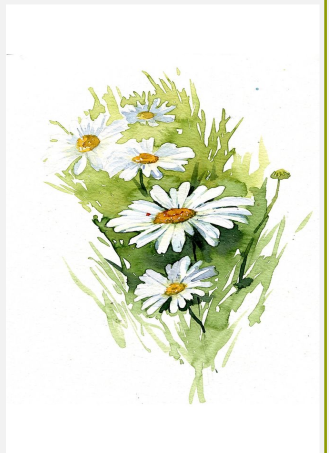
StudentID	ProjectID	StudentName	ProjectName
S89	P09	Olivia	Geo Location
S76	P07	Jacob	Cluster Exploration
S56	P03	Ava	IoT Devices
S92	P05	Alexandra	Cloud Deployment

primary key: {StudentID, ProjectID}

We have **partial dependencies**:

- The *StudentName* can be determined by *StudentID(part of primary key)*
- The *ProjectName* can be determined by *ProjectID(part of primary key)*

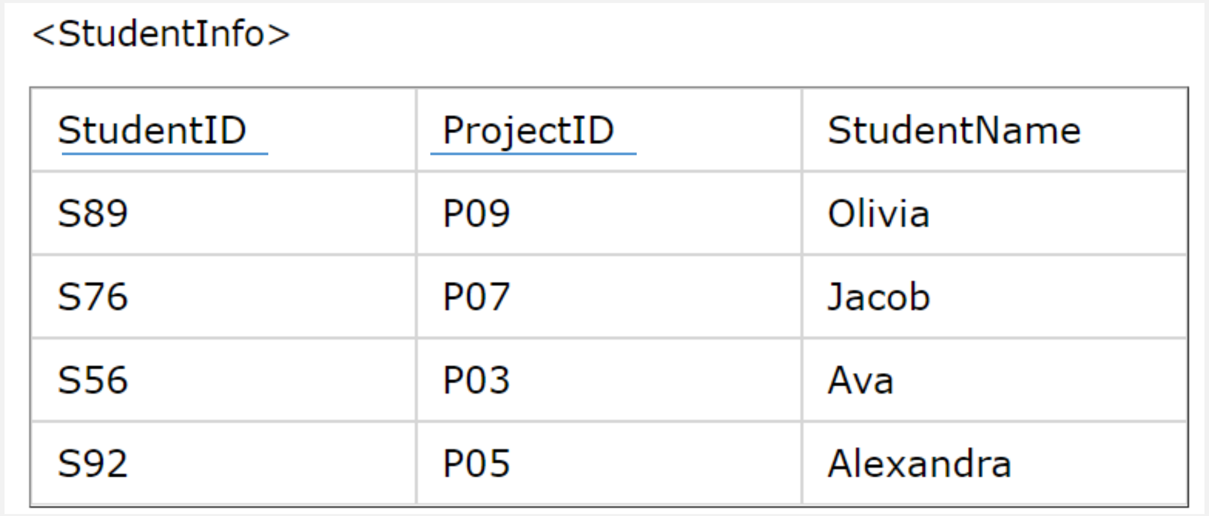
Solution?



All partial dependencies are removed to place in another table!

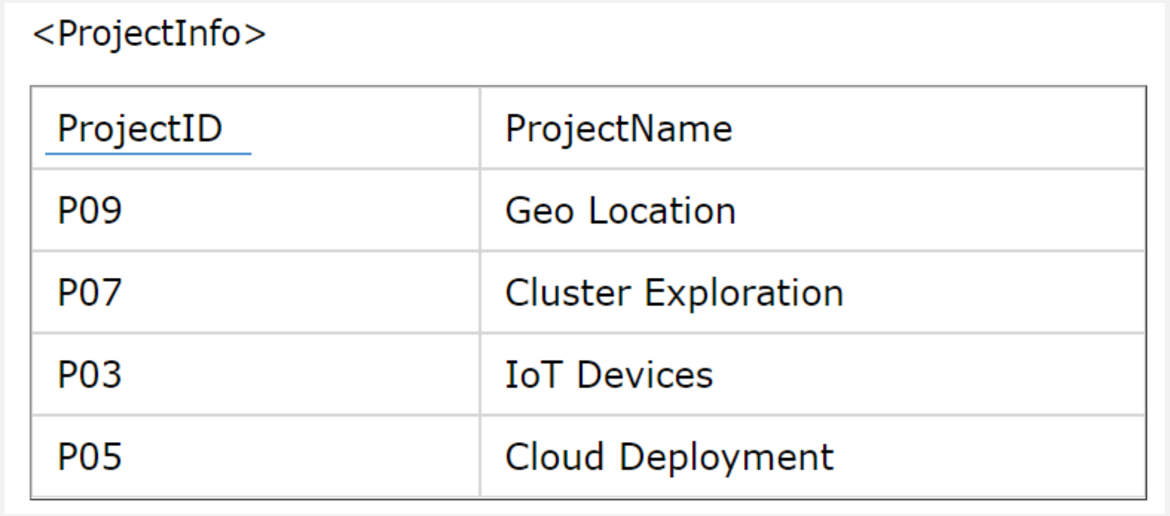
Solution!

- Remove *StudentName* and *StudentID* together to create a new table
  OR Remove *ProjectName* and *ProjectID* together to create a new table
- Remove the non-primary attributes that was removed in the former steps, and leave the remaining table as a new table



<StudentInfo>

StudentID	ProjectID	StudentName
S89	P09	Olivia
S76	P07	Jacob
S56	P03	Ava
S92	P05	Alexandra

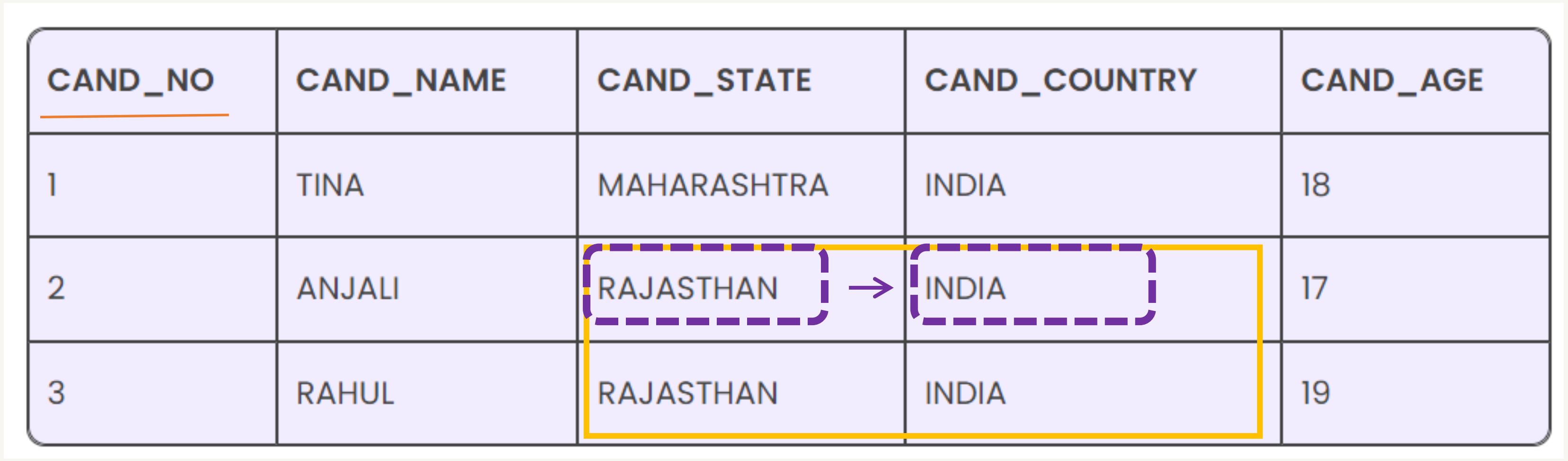


<ProjectInfo>

ProjectID	ProjectName
P09	Geo Location
P07	Cluster Exploration
P03	IoT Devices
P05	Cloud Deployment

why do we need 3NF

- We use the 3NF to **reduce any duplication of data and achieve data integrity** in a database.
- Although Second Normal Form (2NF)relations have less redundancy than those in 1NF, they may **still suffer from update anomalies**. If we update only one tuple and not the other, the database will be in an inconsistent state. This update anomaly is caused by a transitive dependency. We need to remove such dependencies by progressing to the Third Normal Form (3NF).



CAND_NO	CAND_NAME	CAND_STATE	CAND_COUNTRY	CAND_AGE
1	TINA	MAHARASHTRA	INDIA	18
2	ANJALI	RAJASTHAN	INDIA	17
3	RAHUL	RAJASTHAN	INDIA	19

- - - - **update anomaly**

— **redundancy**

What is 3NF?

A table is said to be in 3NF if it meets the following criteria:

- it's already **in 2NF**
- has **no transitive partial dependency,** i.e., non-primary keys don't depend on other non-primary keys.

➡ 3NF ensures that non-key attributes only depend on the primary key

All transtivity dependency are moved to another table!

Detailed process:

- Eliminate all dependent attributes in transitive relationship(s) from each of the tables that have a transitive relationship.
- Create new table(s) with removed dependency.
- Check new table(s) as well as table(s) modified to make sure that each table has a determinant and that no table contains inappropriate dependencies.

What is 3NF?

All transtivity dependency are moved to another table!

Example 1:

A: primary key   B,C,D,E: non-primary key

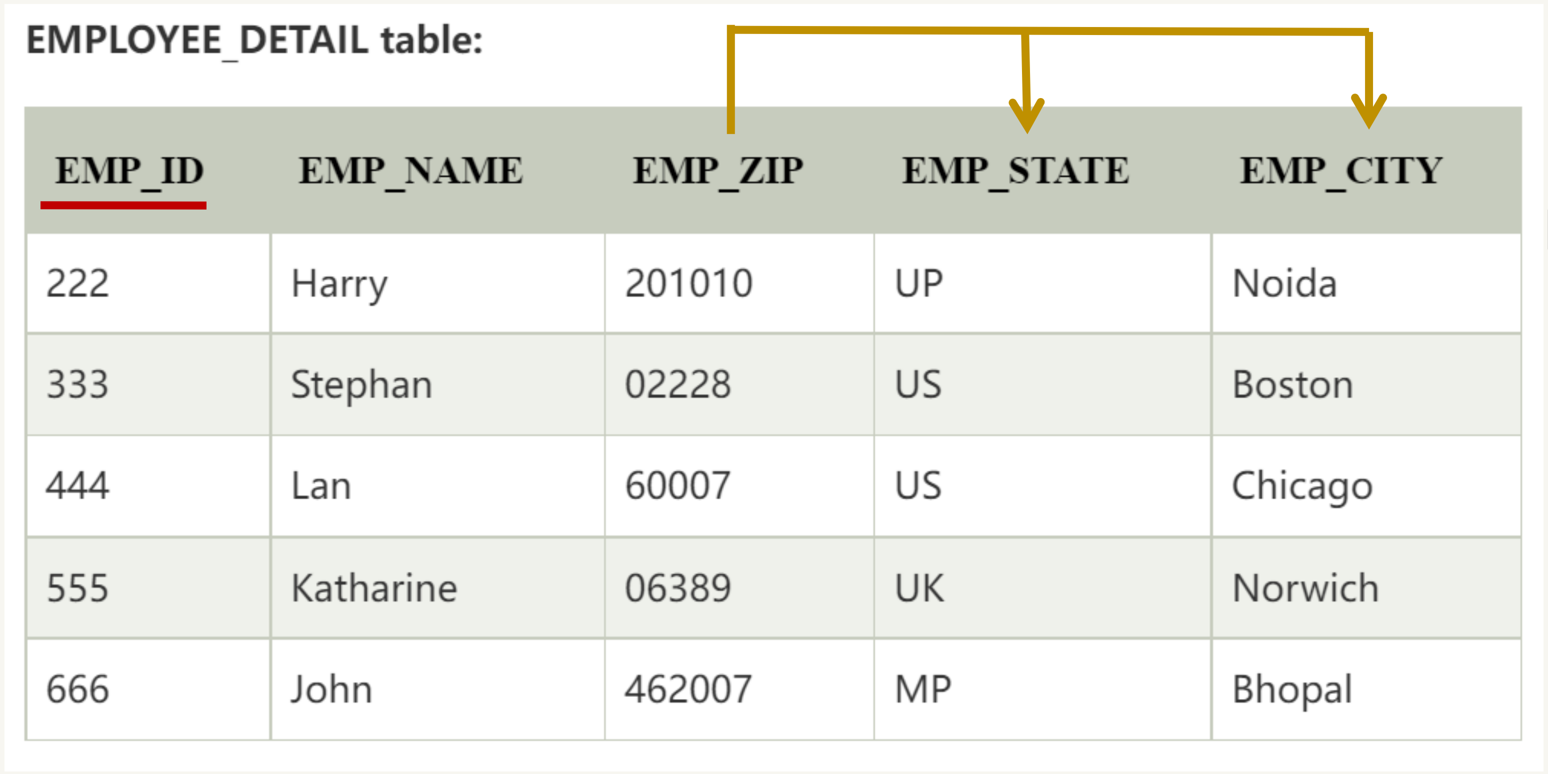
Dependency: {A->BCDE, B->C}   transtivity dependency!

Solution: Move B and C to another table

What is 3NF?

All transtivity dependency are moved to another table!

Example 2:



EMPLOYEE_DETAIL table:

EMP_ID	EMP_NAME	EMP_ZIP	EMP_STATE	EMP_CITY
222	Harry	201010	UP	Noida
333	Stephan	02228	US	Boston
444	Lan	60007	US	Chicago
555	Katharine	06389	UK	Norwich
666	John	462007	MP	Bhopal

primary key: EMP_ID

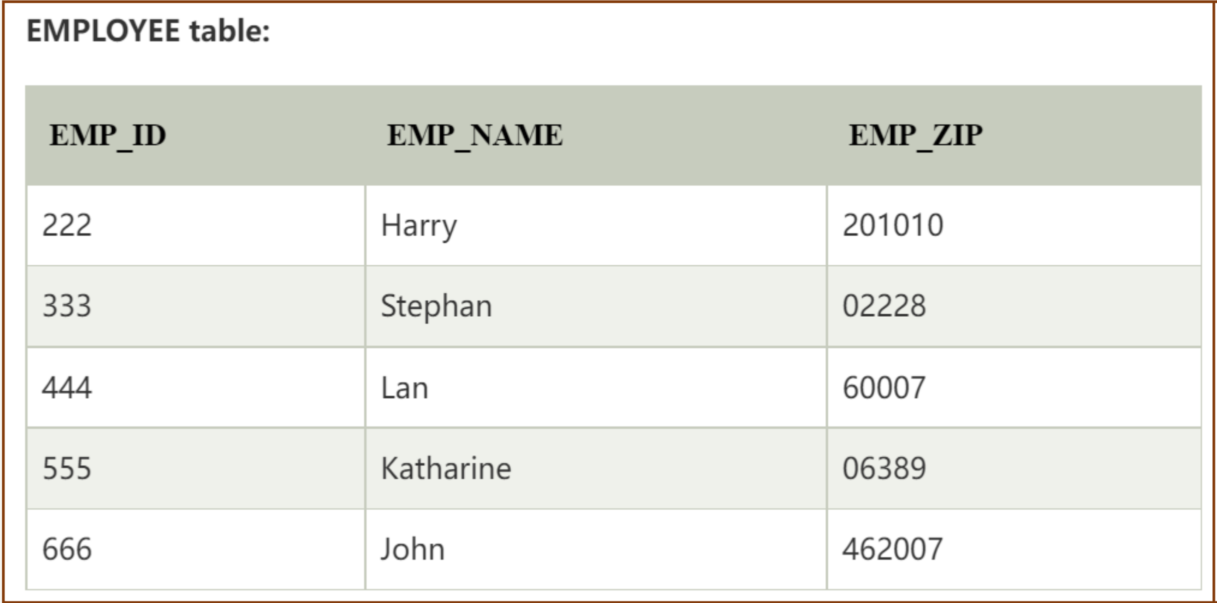
EMP_ZIP->EMP_STATE, EMP_CITY

Is this 3NF?

What is 3NF?

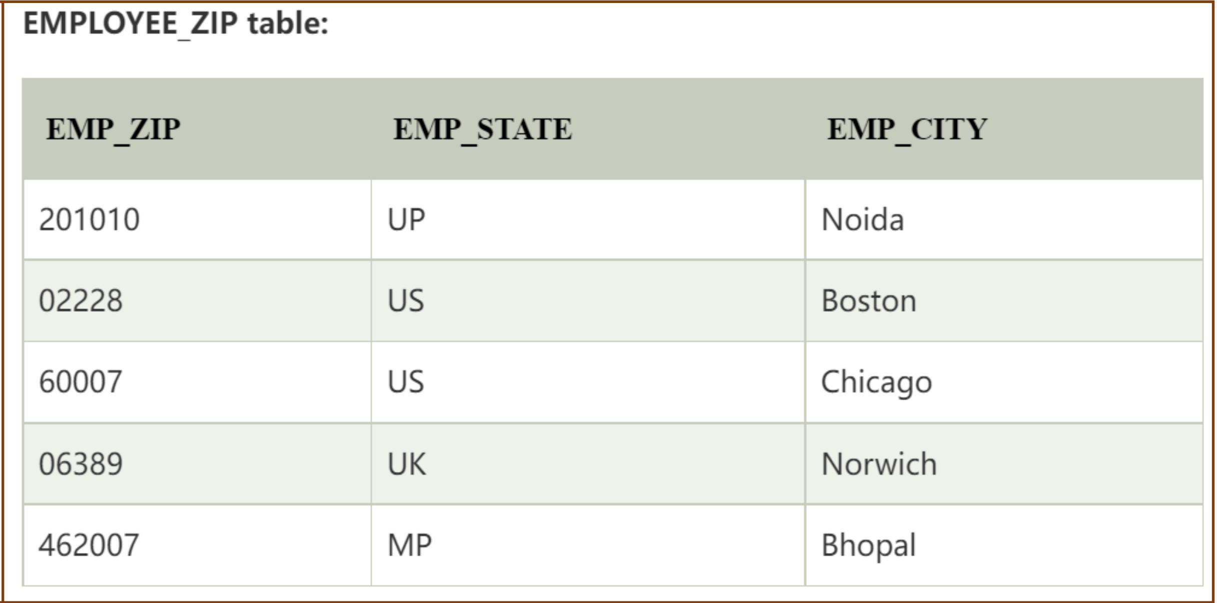
All transtivity dependency are moved to another table!

Recall: EMP_ZIP->EMP_STATE, EMP_CITY
Remove ***EMP_ZIP, EMP_STATE, EMP_CITY*** together to create a new table



EMPLOYEE table:

EMP_ID	EMP_NAME	EMP_ZIP
222	Harry	201010
333	Stephan	02228
444	Lan	60007
555	Katharine	06389
666	John	462007



EMPLOYEE_ZIP table:

EMP_ZIP	EMP_STATE	EMP_CITY
201010	UP	Noida
02228	US	Boston
60007	US	Chicago
06389	UK	Norwich
462007	MP	Bhopal

Thanks!