Normalization

- Normalization is **the process of minimizing redundancy from a relation or set of relations**.
- Redundancy in relation may cause insertion, deletion, and update anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

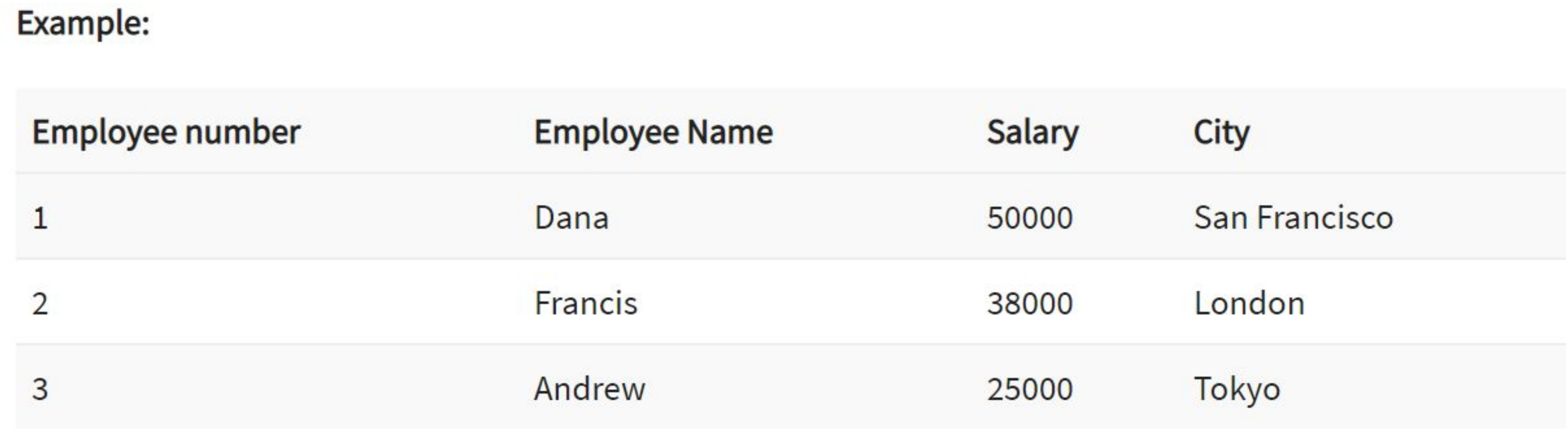
- Normalization is the process of reorganizing data in a database so that it meets two basic requirements:

1. There is no redundancy of data, all data is stored in only one place.
2. Data dependencies are logical,all related data items are stored together.

- Normalization is important for many reasons, but chiefly because it allows databases to take up as little disk space as possible, resulting in increased performance.
- Normalization is also known as data normalization.

Functional Dependancy

- A functional dependency is a constraint that specifies the relationship between two sets of attributes where one set can accurately determine the value of other sets.
- It is denoted as **X** → **Y**, where X is a set of attributes that is capable of determining the value of Y.
- Y is functionally dependent on X.
- The attribute set on the left side of the arrow, **X** is called **Determinant**, while on the right side, **Y** is called the **Dependent**.
- Functional Dependency helps to maintain the quality of data in the database. It plays a vital role to find the difference between good and bad database design.



Example:

Employee number	Employee Name	Salary	City
1	Dana	50000	San Francisco
2	Francis	38000	London
3	Andrew	25000	Tokyo

In this example, if we know the value of Employee number, we can obtain Employee Name, city, salary, etc. By this, we can say that the city, Employee Name, and salary are functionally depended on Employee number.

- Below are some rules needed to determine F+:

1. **Reflexivity:** If X is a superset of Y or Y is a subset of X, then X □ Y.
2. **Augmentation:** If X □ Y, then XZ □ YZ. Or If Z ⊆ W, and X □ Y, then XW □ YZ.
3. **Transitivity:** If X □ Y and Y □ Z, then X □ Z.
4. **Union:** If X □ Y and X □ Z, then X □ YZ.
5. **Decomposition:** If X □ YZ, then X □ Y and X □ Z.
6. **Pseudo-Transitivity:** If X □ Y and YW □ Z, then XW □ Z.

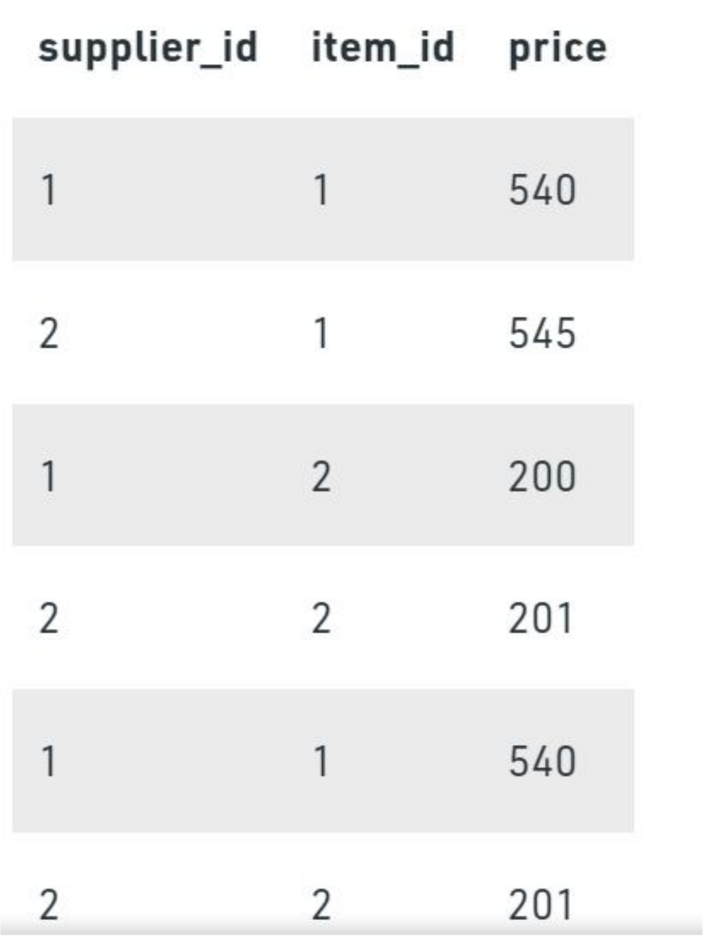
Fully Functional Dependency :

- An attribute is fully functional dependent on another attribute, if it is Functionally Dependent on that attribute and not on any of its proper subset.
- For example, an attribute Q is fully functional dependent on another attribute P, if it is Functionally Dependent on P and not on any of the proper subset of P.

Example –

In the relation ABC->D, attribute D is fully functionally dependent on ABC and not on any proper subset of ABC. That means that subsets of ABC like AB, BC, A, B, etc. cannot determine D.

Let us take another example –



supplier_id	item_id	price
1	1	540
2	1	545
1	2	200
2	2	201
1	1	540
2	2	201

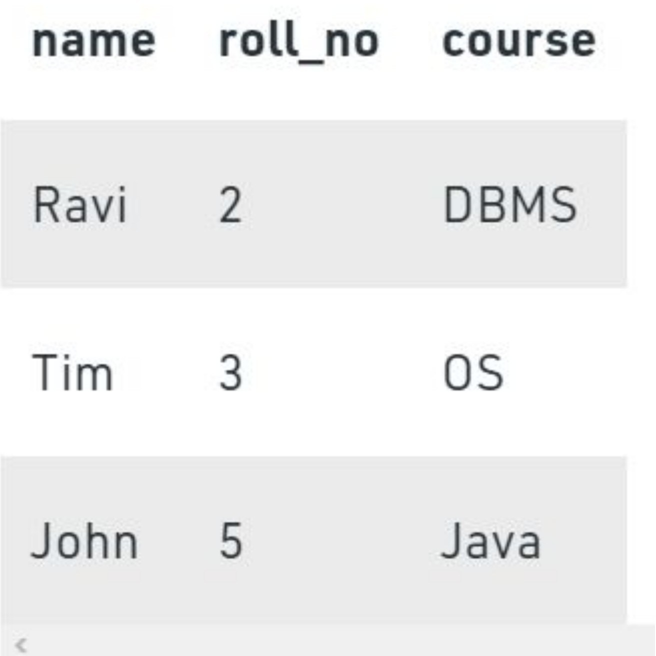
From the table, we can clearly see that neither supplier_id nor item_id can uniquely determine the price but both supplier_id and item_id together can do so. So we can say that price is fully functionally dependent on { supplier_id, item_id }. This summarizes and gives our fully functional dependency −

```
{ supplier_id , item_id } -> price
```

Partial Functional Dependency :

A functional dependency X->Y is a partial dependency if Y is functionally dependent on X and Y can be determined by any proper subset of X.

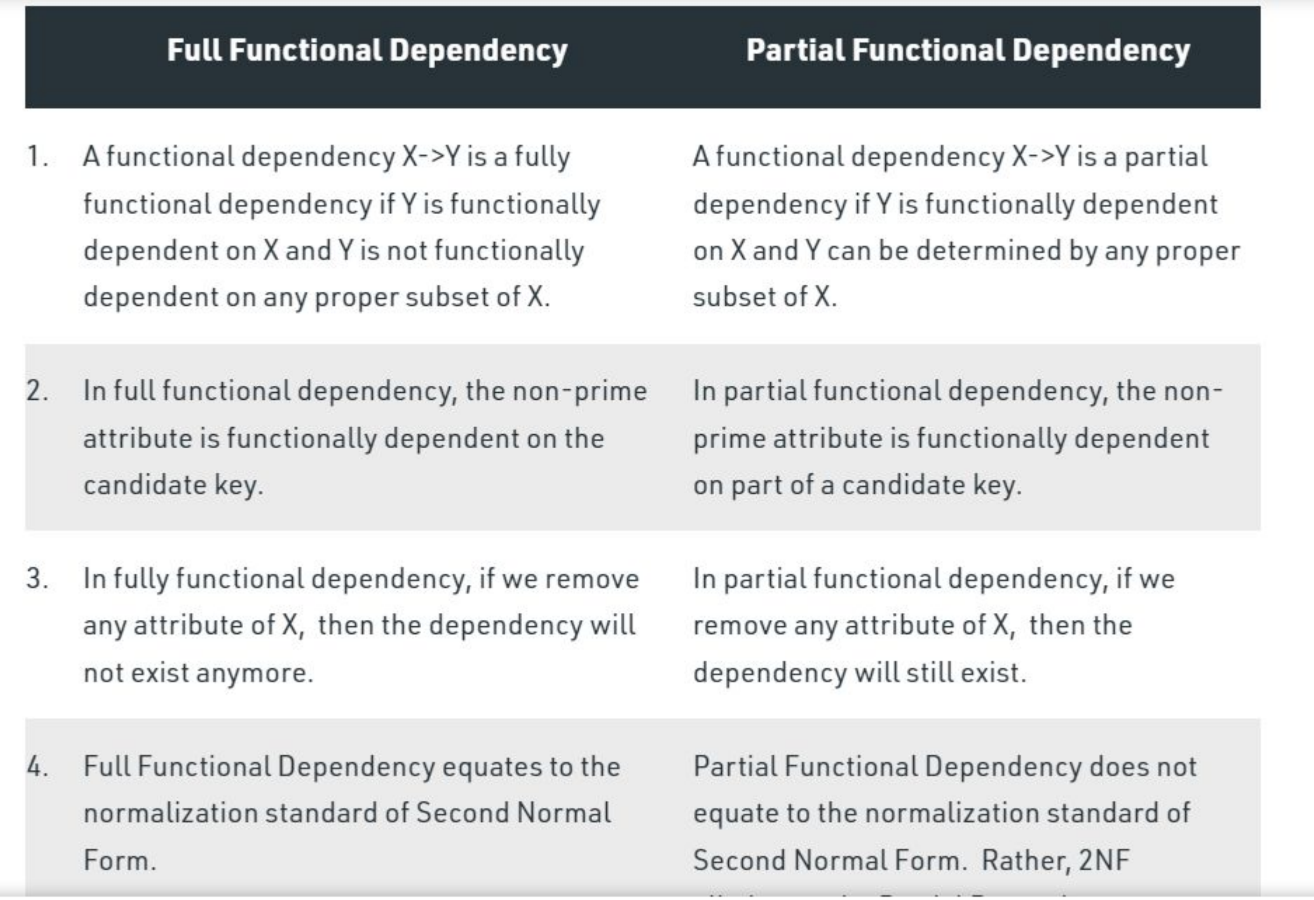
For example, we have a relationship AC->B, A->D, and D->B.
Now if we compute the closure of {A^+}=ADB
Here A is alone capable of determining B, which means B is partially dependent on AC.
Let us take another example –



name	roll_no	course
Ravi	2	DBMS
Tim	3	OS
John	5	Java

Student_id	rollno	course
1	5	BBA
2	6	BEDICT

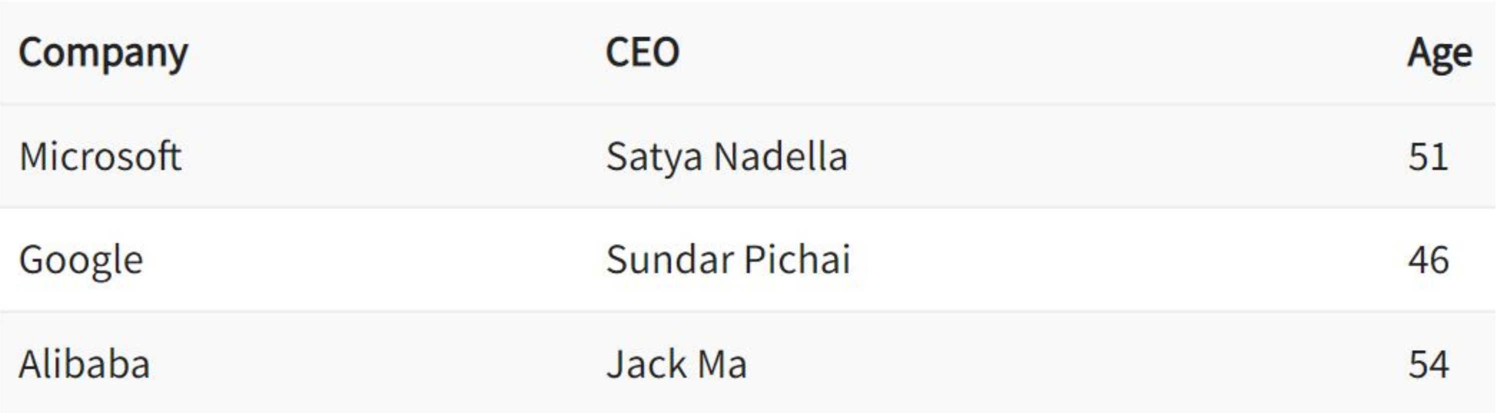
- Here, we can see that both the attributes name and roll_no alone are able to uniquely identify a course. Hence we can say that the relationship is partially dependent.



	Full Functional Dependency	Partial Functional Dependency
1.	A functional dependency X->Y is a fully functional dependency if Y is functionally dependent on X and Y is not functionally dependent on any proper subset of X.	A functional dependency X->Y is a partial dependency if Y is functionally dependent on X and Y can be determined by any proper subset of X.
2.	In full functional dependency, the non-prime attribute is functionally dependent on the candidate key.	In partial functional dependency, the non-prime attribute is functionally dependent on part of a candidate key.
3.	In fully functional dependency, if we remove any attribute of X, then the dependency will not exist anymore.	In partial functional dependency, if we remove any attribute of X, then the dependency will still exist.
4.	Full Functional Dependency equates to the normalization standard of Second Normal Form.	Partial Functional Dependency does not equate to the normalization standard of Second Normal Form. Rather, 2NF

Transitive Dependency in DBMS

- A Transitive Dependency is a type of functional dependency which happens when “t” is indirectly formed by two functional dependencies. Let’s understand with the following Transitive Dependency Example.



Company	CEO	Age
Microsoft	Satya Nadella	51
Google	Sundar Pichai	46
Alibaba	Jack Ma	54

{Company} -> {CEO} (if we know the compay, we know its CEO’s name)

{CEO } -> {Age} If we know the CEO, we know the Age

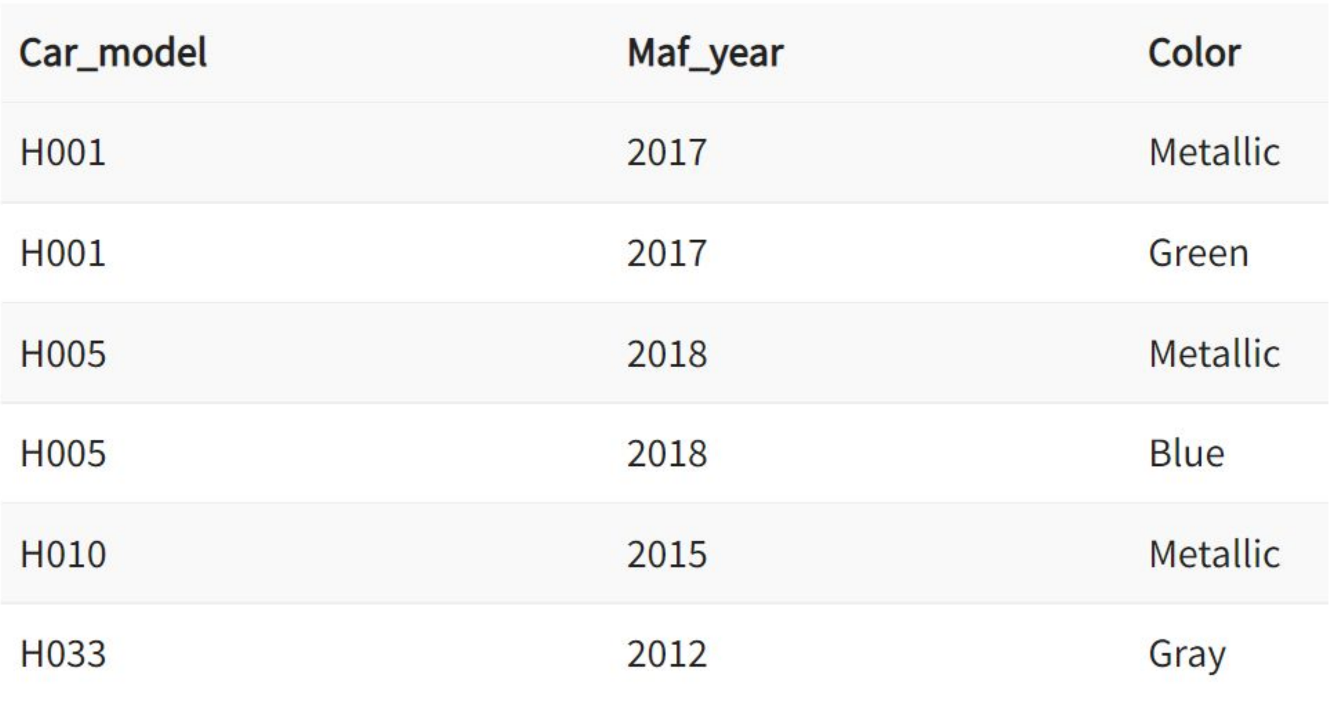
Therefore according to the rule of rule of transitive dependency:

{ Company} -> {Age} should hold, that makes sense because if we know the company name, we can know his age.

Note: You need to remember that transitive dependency can only occur in a relation of three or more attributes.

Multivalued Dependency in DBMS

- Multivalued dependency occurs in the situation where there are multiple independent multivalued attributes in a single table.
- A multivalued dependency is a complete constraint between two sets of attributes in a relation. It requires that certain tuples be present in a relation. Consider the following Multivalued Dependency Example to understand.

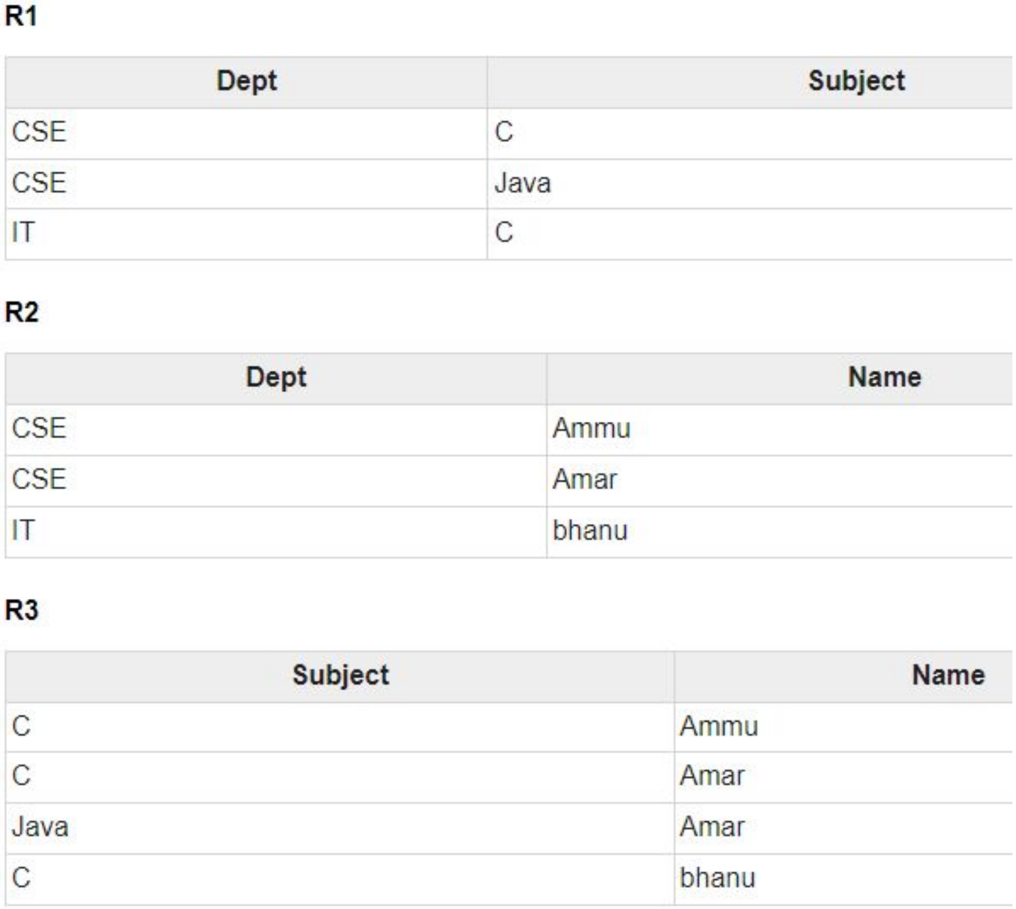


Car_model	Maf_year	Color
H001	2017	Metallic
H001	2017	Green
H005	2018	Metallic
H005	2018	Blue
H010	2015	Metallic
H033	2012	Gray

- In this example, maf_year and color are independent of each other but dependent on car_model. In this example, these two columns are said to be multivalue dependent on car_model.
- This dependence can be represented like this:
- car_model -> maf_year
- car_model-> colour

Join Dependency

- If a table can be recreated by joining multiple tables and each of this table have a subset of the attributes of the table, then the table is in Join Dependency. It is a generalization of Multivalued Dependency
- Join Dependency can be related to 5NF, wherein a relation is in 5NF, only if it is already in 4NF and it cannot be decomposed further.



R1

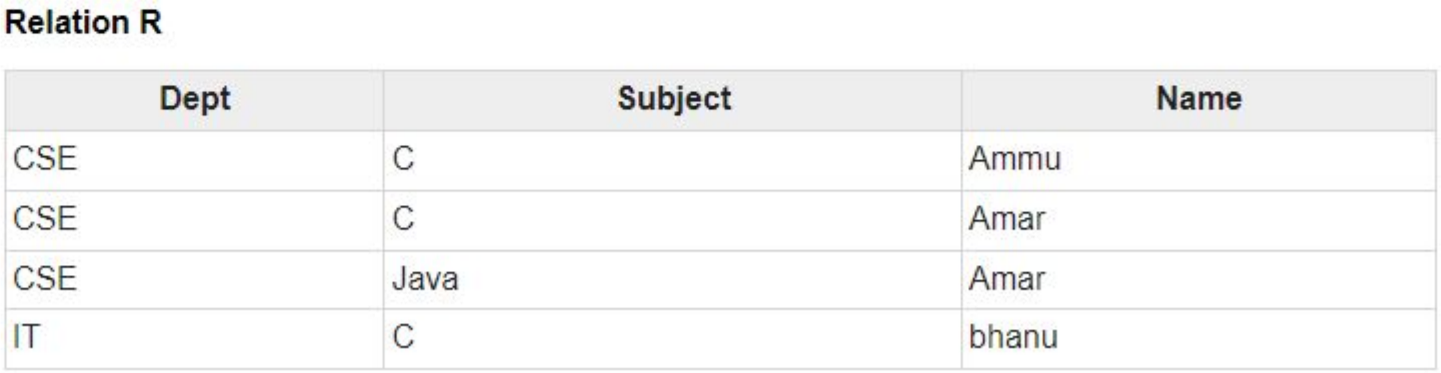
Dept	Subject
CSE	C
CSE	Java
IT	C

R2

Dept	Name
CSE	Ammu
CSE	Amar
IT	bhanu

R3

Subject	Name
C	Ammu
C	Amar
Java	Amar
C	bhanu



Relation R

Dept	Subject	Name
CSE	C	Ammu
CSE	C	Amar
CSE	Java	Amar
IT	C	bhanu

Anomalies

- **Data anomalies** are inconsistencies in the **data** stored in a **database** as a result of an operation such as update, insertion, and/or deletion
- Database anomaly is **normally the flaw in databases** which occurs because of poor planning and storing everything in a flat database.
- Generally this is removed by the process of normalization which is performed by splitting/joining of tables.

Anomalies

- There are three types of anomalies: update, deletion, and insertion anomalies.
- For example, each employee in a company has a department associated with them as well as the student group they participate in.

Insertion Anomaly

- **Insertion Anomalies** happen when inserting vital data into the database is not possible because other data is not already there.
- For example, if a system is designed to require that a customer be on file before a sale can be made to that customer, but you cannot add a customer until they have bought something, then you have an insert anomaly.

id	Student	Course id	Course name	
001	Samikshya	002	Java	
002	Samjhana	005	C++	
002	Samjhana	005	C++	
003	Saru	NULL	NULL	

If course is new then first we must add the course in databse and then only assign course id and name to student . This creates insertion namolies

Update Anomaly

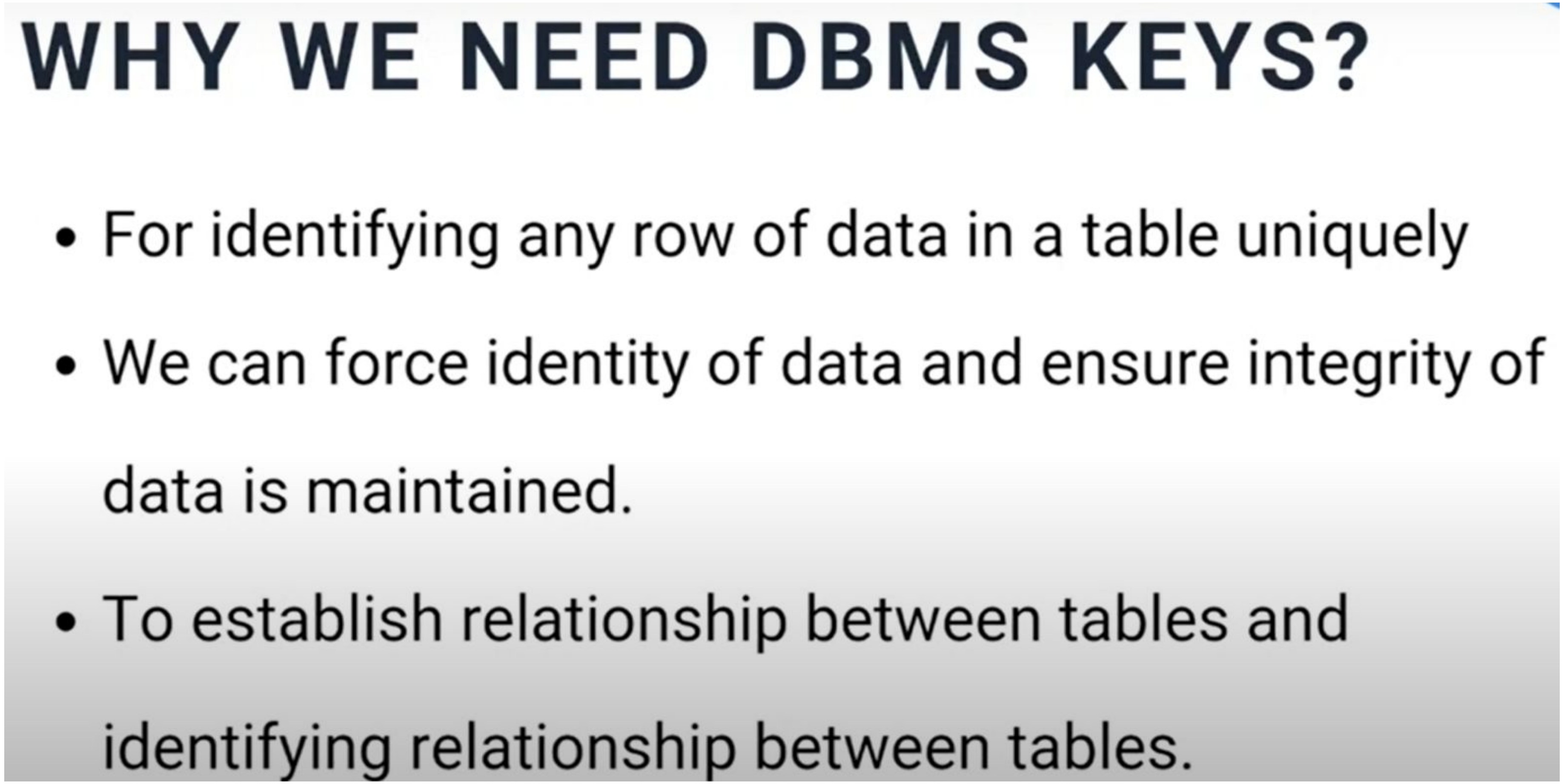
- An update anomaly is a data inconsistency that results from data redundancy and a partial update.
- For example, to change an employee’s title due to a promotion.
- If the data is stored redundantly in the same table, and the person misses any of them, then there will be multiple titles associated with the employee. The end user has no way of knowing which is the correct title.

Deletion Anomaly

- A deletion anomaly is the unintended loss of data due to deletion of other data.
- For example, For example, if a single database record contains information about a particular product along with information about a salesperson for the company and the salesperson quits, then information about the product is deleted along with salesperson information.

Key

- A key in DBMS is **an attribute or a set of attributes that help to uniquely identify a tuple (or row) in a relation (or table)**. Keys are also used to establish relationships between the different tables and columns of a relational database. Individual values in a key are called key values.

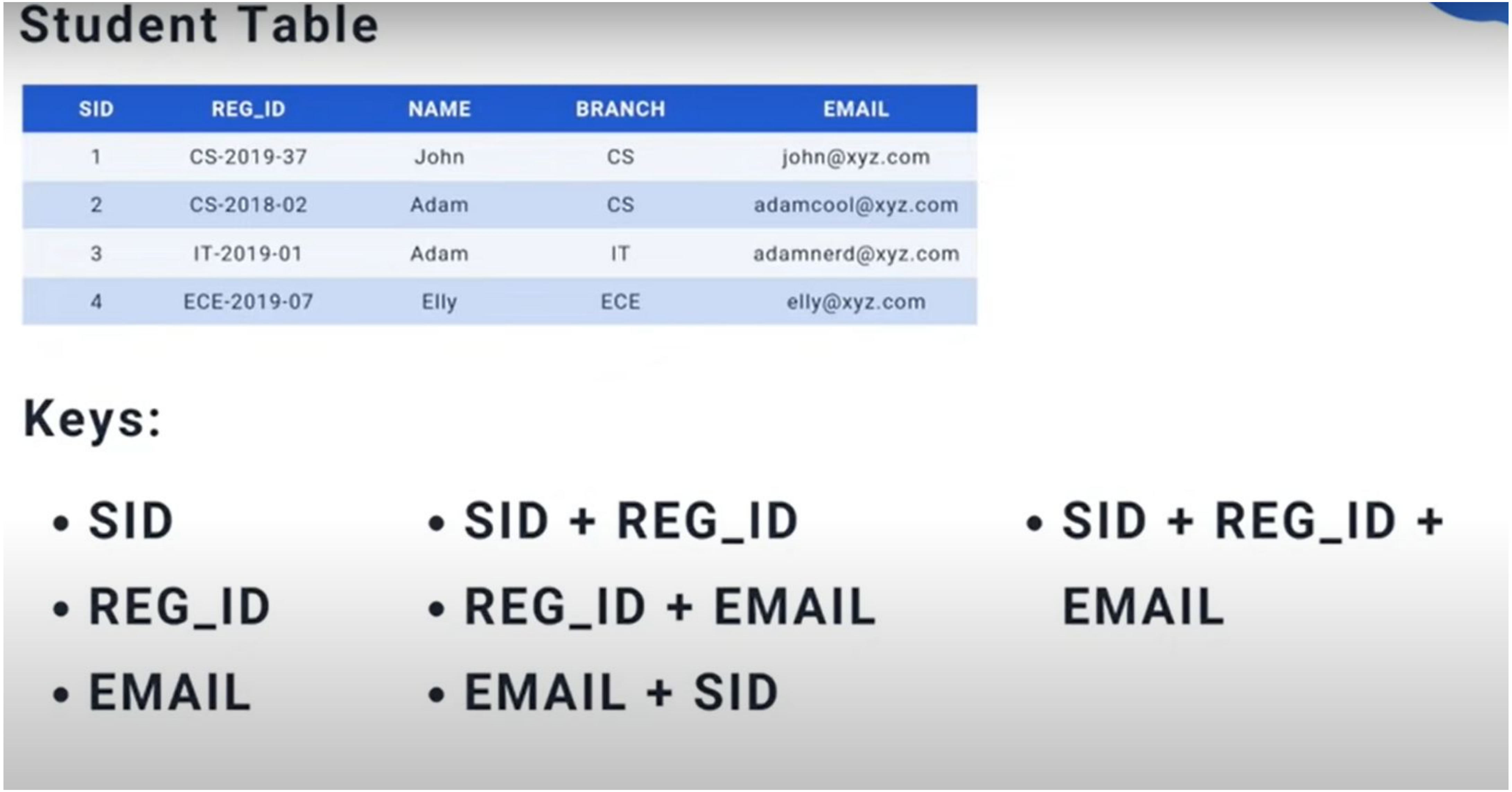


WHY WE NEED DBMS KEYS?

- For identifying any row of data in a table uniquely
- We can force identity of data and ensure integrity of data is maintained.
- To establish relationship between tables and identifying relationship between tables.

Super Key

- Super key is an attribute or set of attribute that can be used to identify the row in a table .
- Super Key is the set of all the keys which help to identify rows in a table uniquely.
- This means that all those columns of a table than capable of identifying the other columns of that table uniquely will all be considered super keys.
- Super Key is the superset of a candidate key (explained below). The Primary Key of a table is picked from the super key set to be made the table's identity attribute.



Student Table

SID	REG_ID	NAME	BRANCH	EMAIL
1	CS-2019-37	John	CS	john@xyz.com
2	CS-2018-02	Adam	CS	adamcool@xyz.com
3	IT-2019-01	Adam	IT	adamnerd@xyz.com
4	ECE-2019-07	Elly	ECE	elly@xyz.com

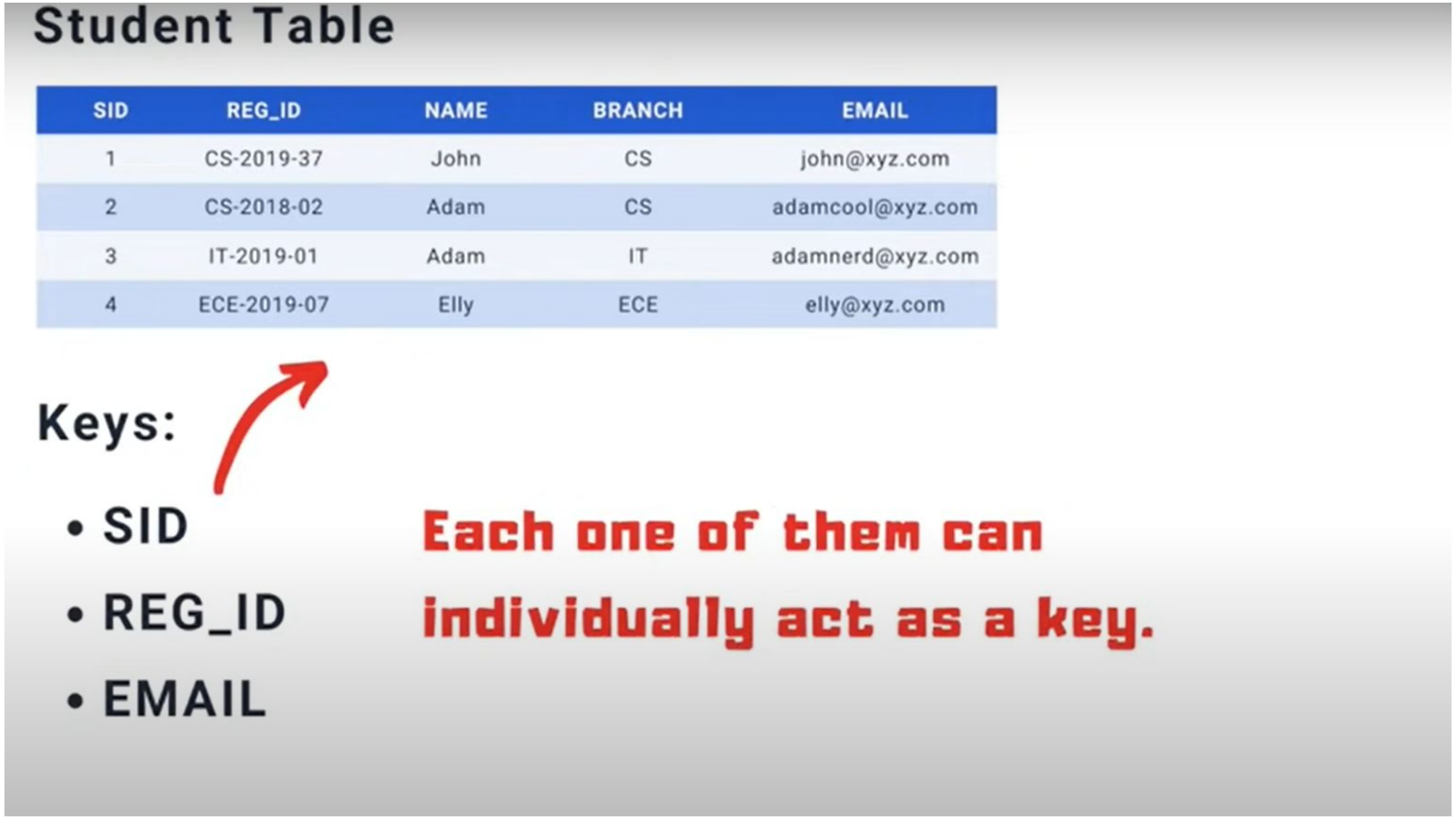
Keys:

- SID
- REG_ID
- EMAIL
- SID + REG_ID
- REG_ID + EMAIL
- EMAIL + SID
- SID + REG_ID + EMAIL

Candidate key

- Minumum subset of super key.
- A candidate key is a specific type of field in a relational database that can identify each unique record independently of any other data.
- Candidate keys are those attributes that uniquely identify rows of a table.
- The Primary Key of a table is selected from one of the candidate keys. So, candidate keys have the same properties as the primary keys explained above. There can be more than one candidate keys in a table.

Candidate key



Student Table

SID	REG_ID	NAME	BRANCH	EMAIL
1	CS-2019-37	John	CS	john@xyz.com
2	CS-2018-02	Adam	CS	adamcool@xyz.com
3	IT-2019-01	Adam	IT	adamnerd@xyz.com
4	ECE-2019-07	Elly	ECE	elly@xyz.com

Keys:

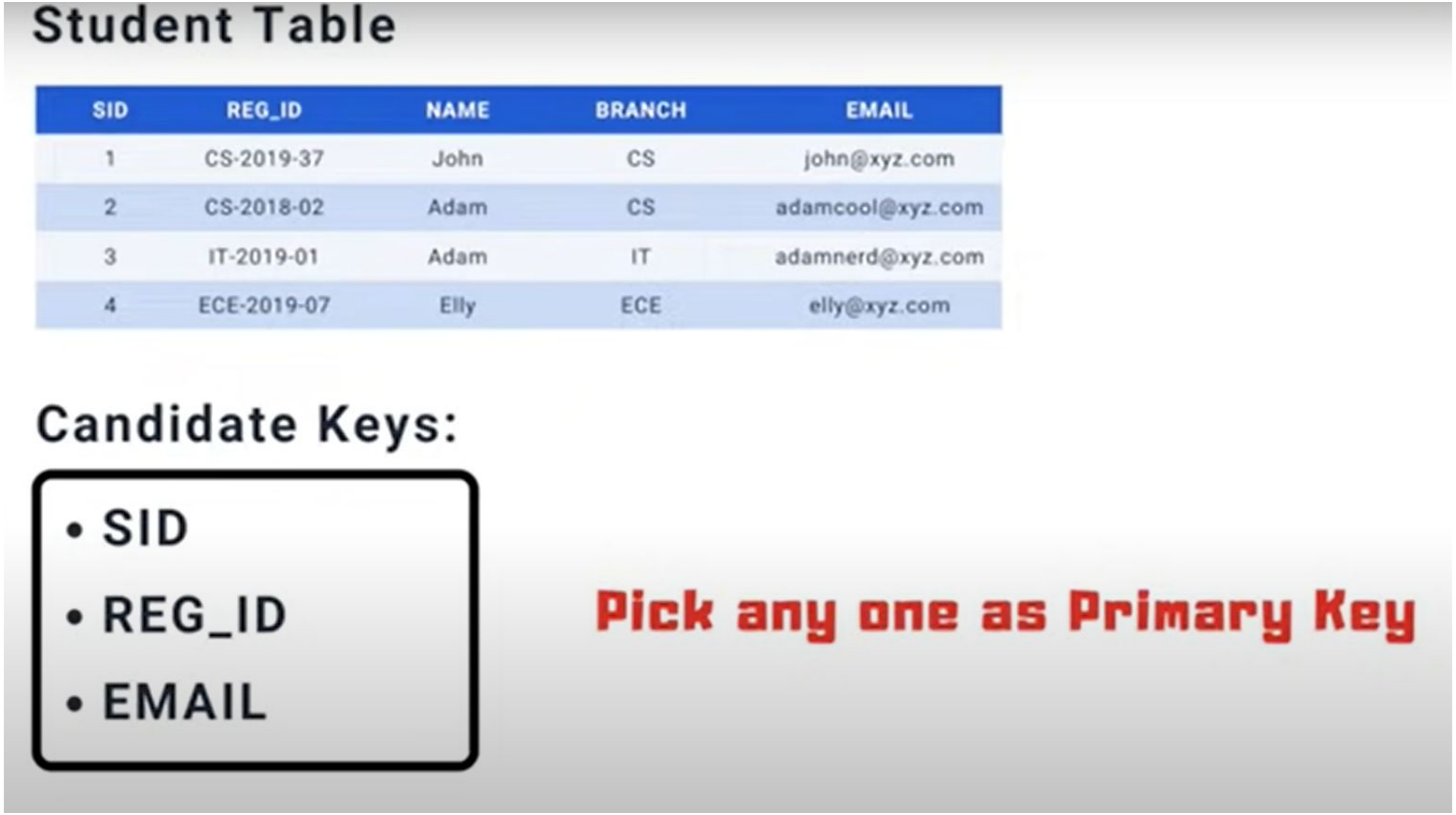
- SID
- REG_ID
- EMAIL

Each one of them can individually act as a key.

Keys

Primary Key

- A primary is a single column value used to identify a database record uniquely.
- It has following attributes
- A primary key cannot be NULL
- A primary key value must be unique
- The primary key values should rarely be changed
- The primary key must be given a value when a new record is inserted.



Student Table

SID	REG_ID	NAME	BRANCH	EMAIL
1	CS-2019-37	John	CS	john@xyz.com
2	CS-2018-02	Adam	CS	adamcool@xyz.com
3	IT-2019-01	Adam	IT	adamnerd@xyz.com
4	ECE-2019-07	Elly	ECE	elly@xyz.com

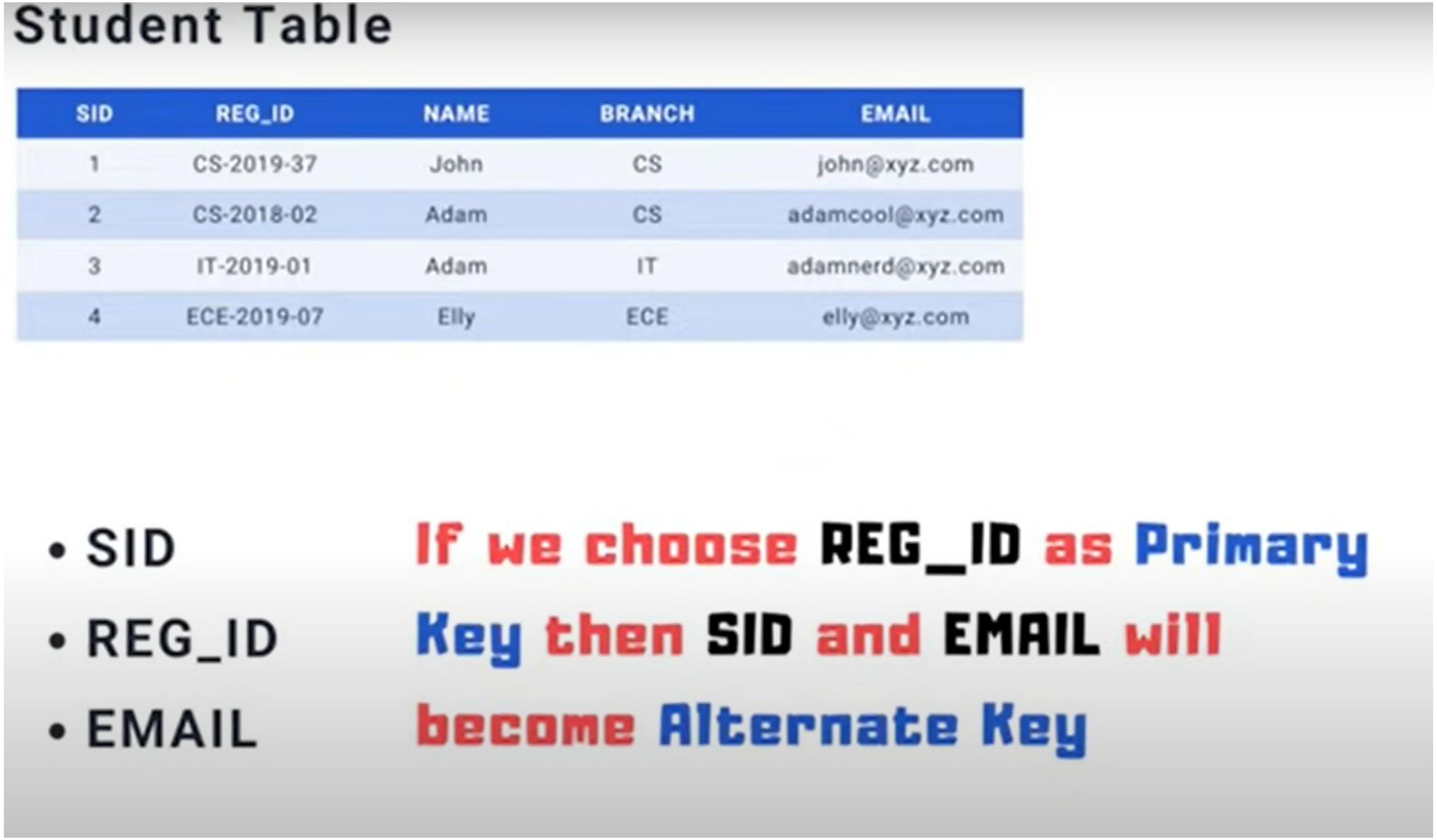
Candidate Keys:

- SID
- REG_ID
- EMAIL

Pick any one as Primary Key

Alternate Key

- As stated above, a table can have multiple choices for a primary key; however, it can choose only one. So, all the keys which did not become the primary Key are called alternate keys.



Student Table

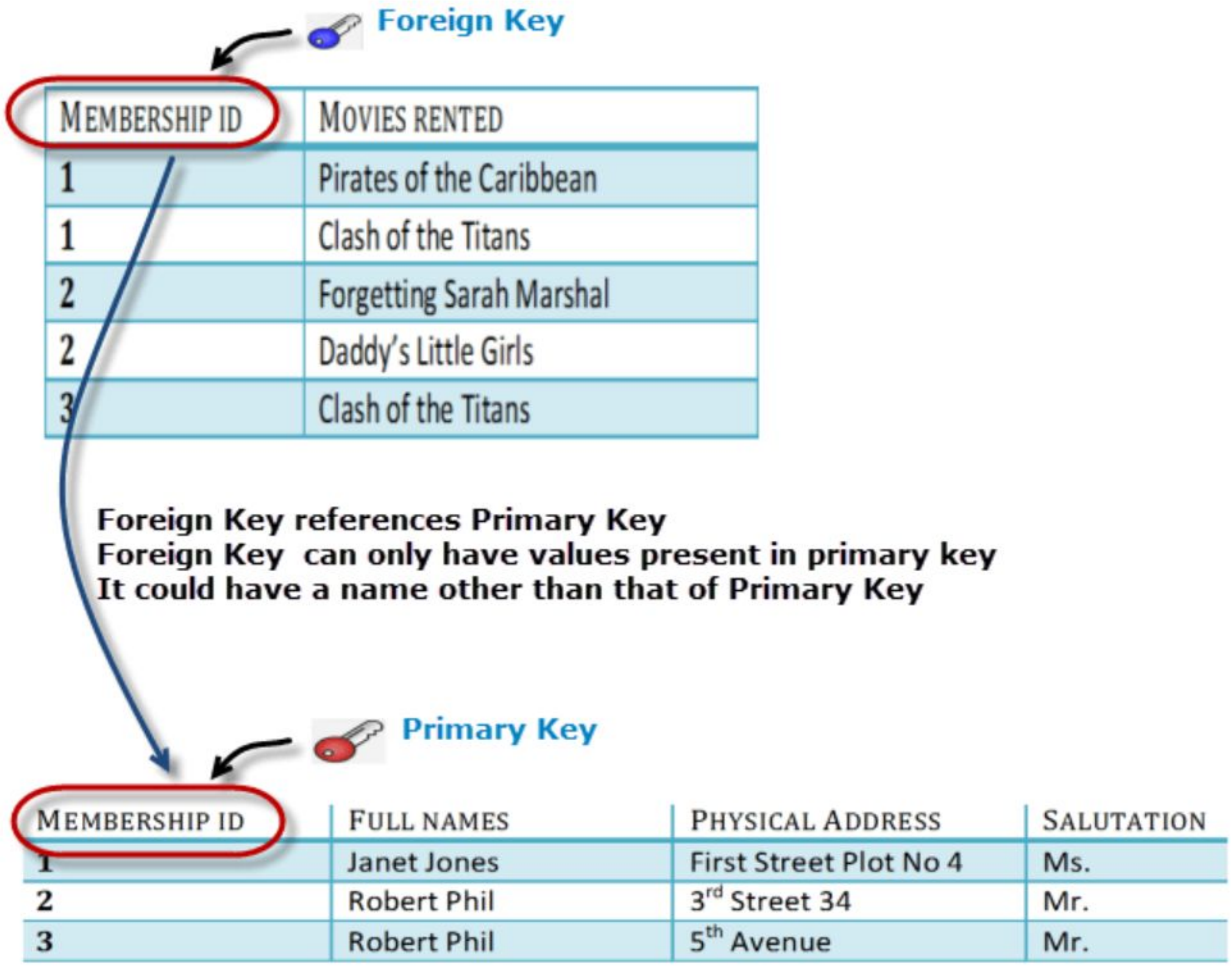
SID	REG_ID	NAME	BRANCH	EMAIL
1	CS-2019-37	John	CS	john@xyz.com
2	CS-2018-02	Adam	CS	adamcool@xyz.com
3	IT-2019-01	Adam	IT	adamnerd@xyz.com
4	ECE-2019-07	Elly	ECE	elly@xyz.com

- SID
- REG_ID
- EMAIL

If we choose REG_ID as Primary Key then SID and EMAIL will become Alternate Key

Foreign Key

- Foreign Key references the primary key of another Table! It helps connect your Tables
- A foreign key can have a different name from its primary key
- It ensures rows in one table have corresponding rows in another
- Unlike the Primary key, they do not have to be unique. Most often they aren’t
- Foreign keys can be null even though primary keys can not
-



Foreign Key

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

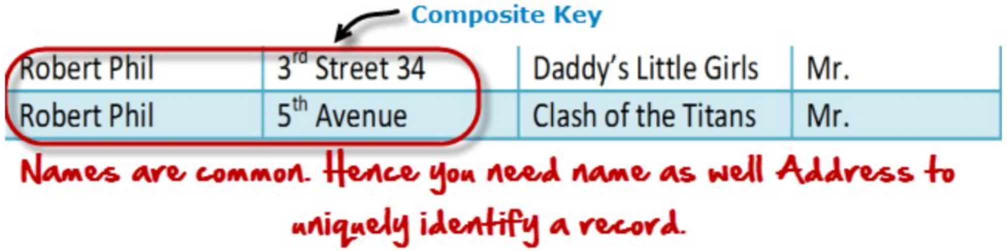
Foreign Key references Primary Key
Foreign Key can only have values present in primary key
It could have a name other than that of Primary Key

Primary Key

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Composite Key

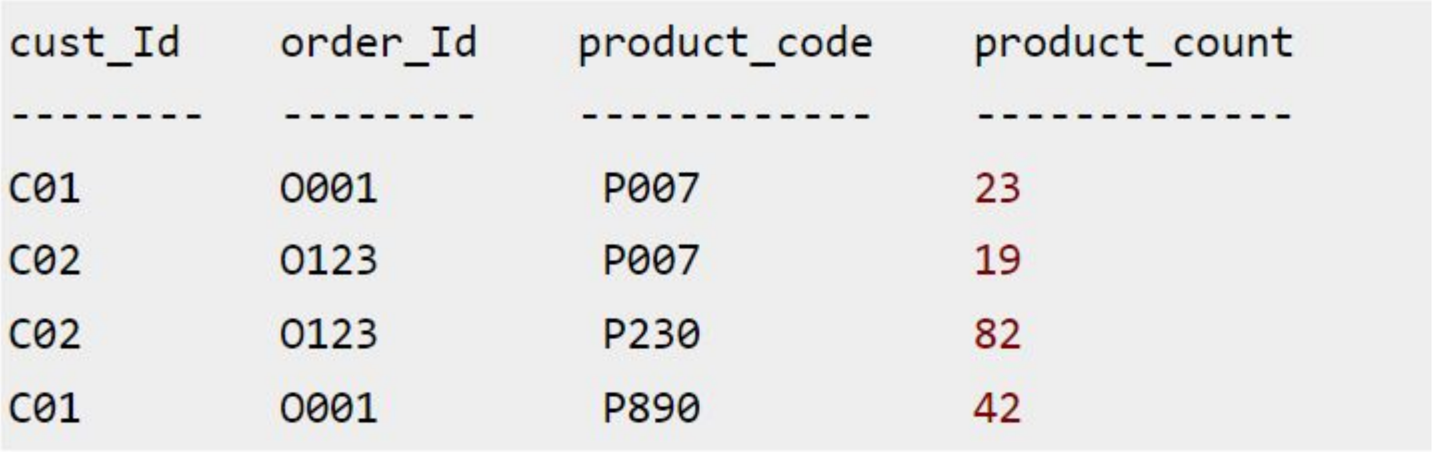
- A composite key is a primary key composed of multiple columns used to identify a record uniquely
- In our database, we have two people with the same name Robert Phil, but they live in different places.
- Hence, we require both Full Name and Address to identify a record uniquely. That is a composite key.


Composite Key

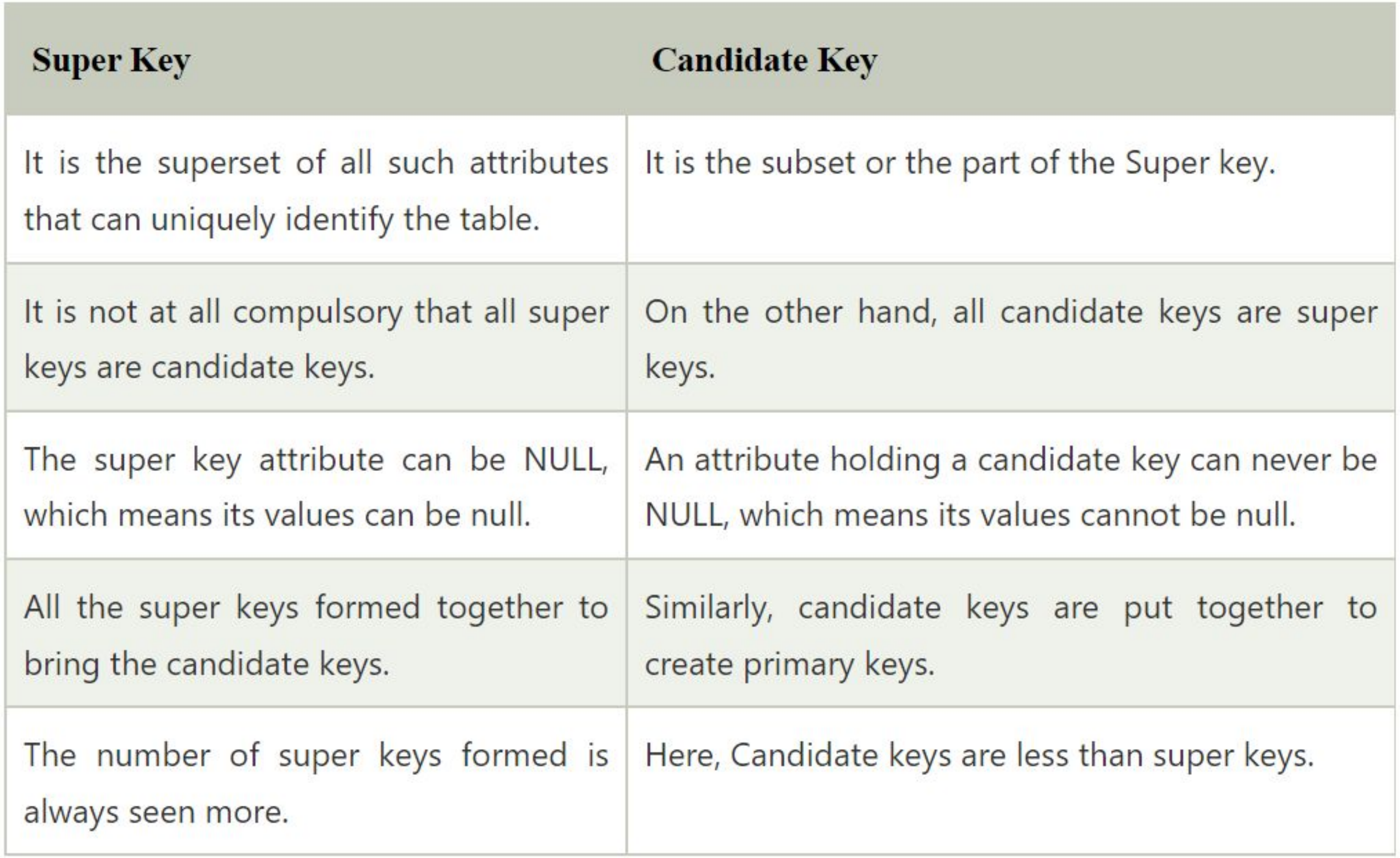
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

Names are common. Hence you need name as well Address to uniquely identify a record.

- **Key in above table: {cust_id, product_code}**
- This is a composite key as it is made up of more than one attributes.



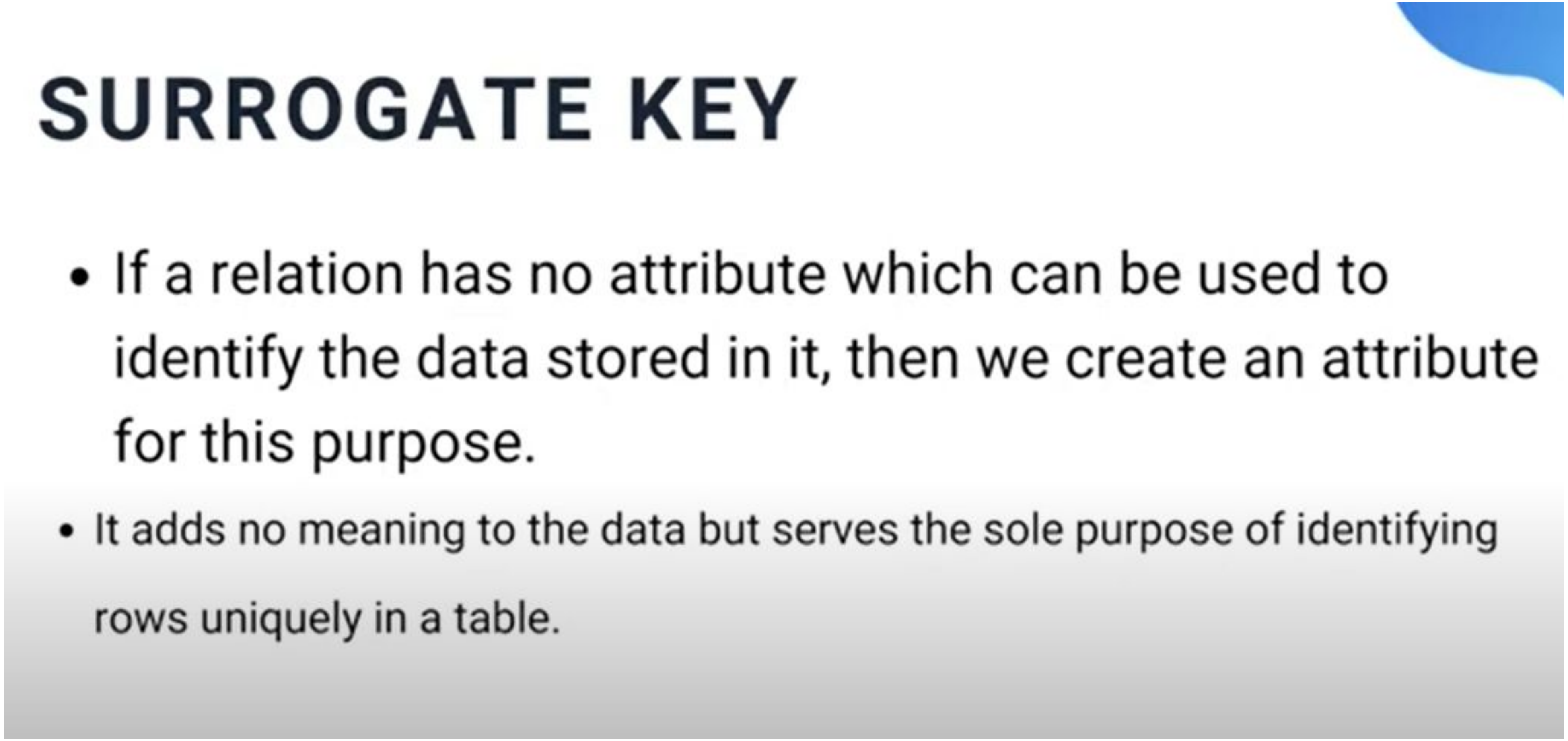
```
cust_Id    order_Id    product_code    product_count
--------   --------    ------------    -------------
C01        O001         P007           23
C02        O123         P007           19
C02        O123         P230           82
C01        O001         P890           42
```



Super Key	Candidate Key
It is the superset of all such attributes that can uniquely identify the table.	It is the subset or the part of the Super key.
It is not at all compulsory that all super keys are candidate keys.	On the other hand, all candidate keys are super keys.
The super key attribute can be NULL, which means its values can be null.	An attribute holding a candidate key can never be NULL, which means its values cannot be null.
All the super keys formed together to bring the candidate keys.	Similarly, candidate keys are put together to create primary keys.
The number of super keys formed is always seen more.	Here, Candidate keys are less than super keys.

Compound Key

- A compound key is **similar to a composite key in that two or more fields are needed to create a unique value**.
- However, a compound key is created when two or more primary keys from different tables are present as foreign keys within an entity.
- The foreign keys are used together to uniquely identify each record.



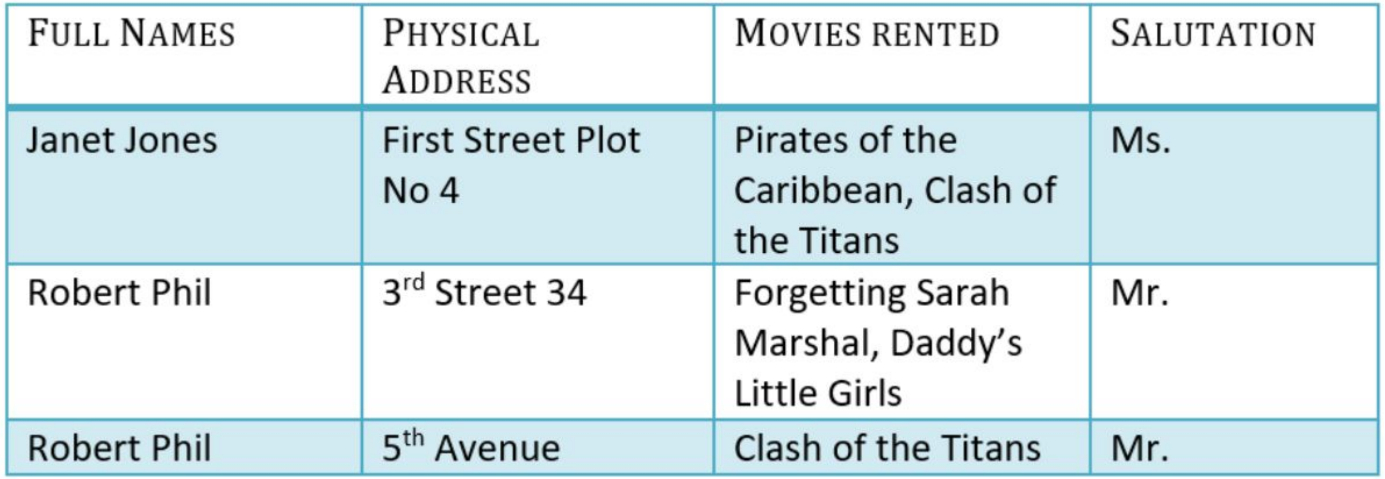
SURROGATE KEY

- If a relation has no attribute which can be used to identify the data stored in it, then we create an attribute for this purpose.
- It adds no meaning to the data but serves the sole purpose of identifying rows uniquely in a table.

1NF (First Normal Form)

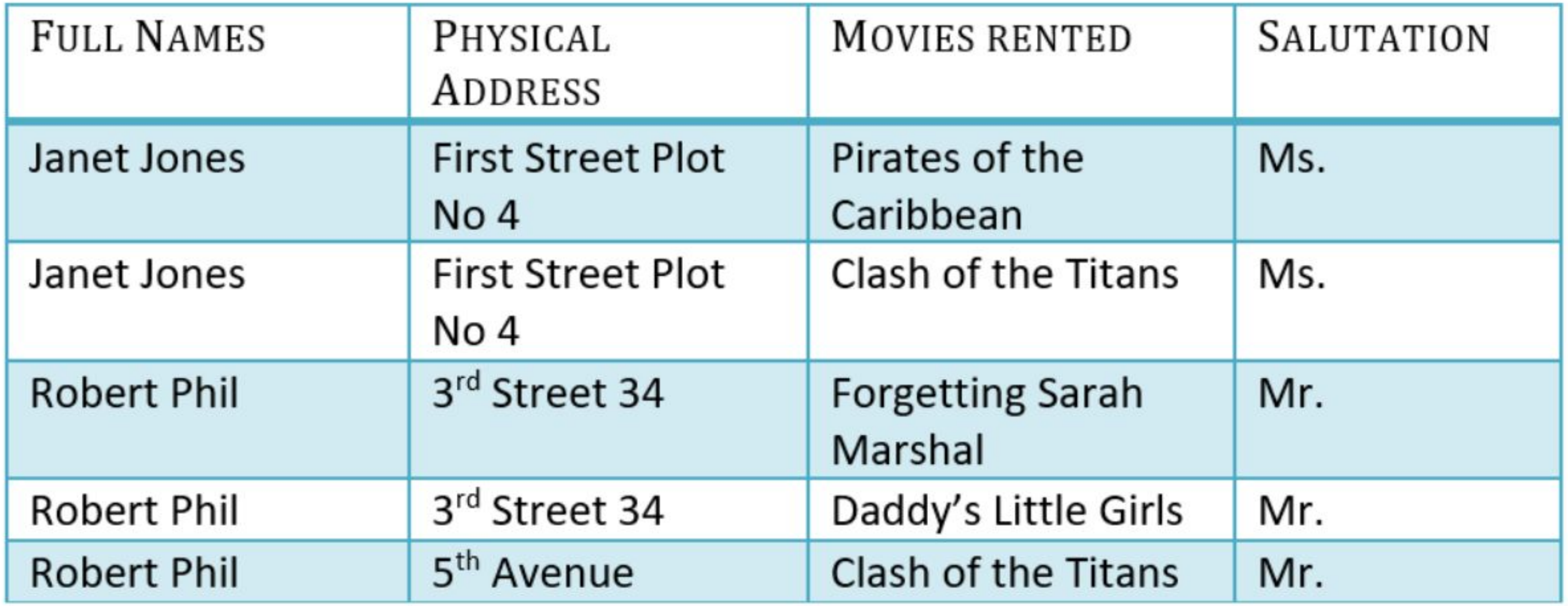
Rules:

- Each table cell should contain a single value.
- Each record needs to be unique.



FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

1NF Example

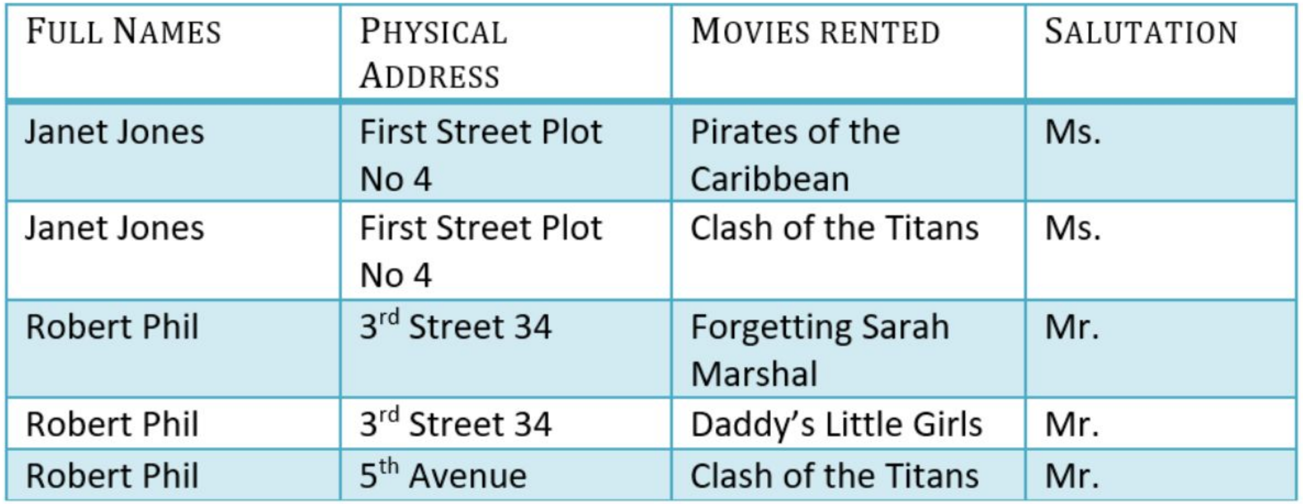


FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

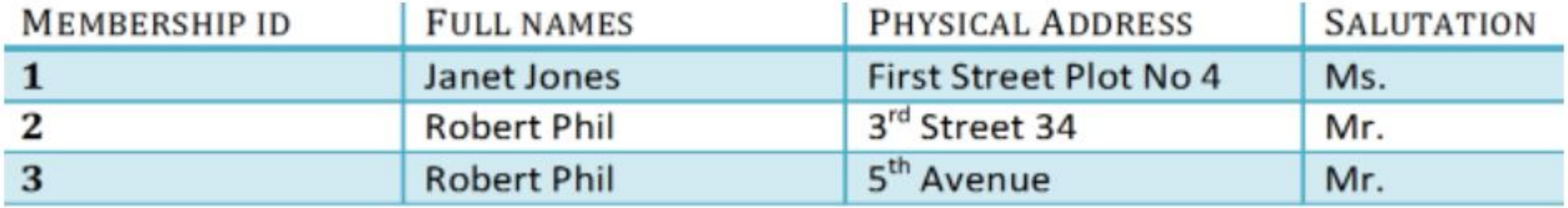
2NF (Second Normal Form)

- **Rules**
- Rule 1- Be in 1NF
- Rule 2- it should not have Partial Dependency.

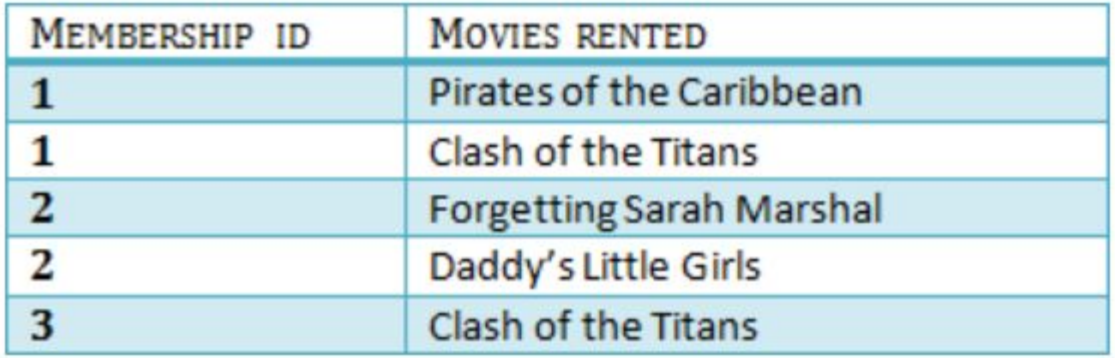
1NF Example



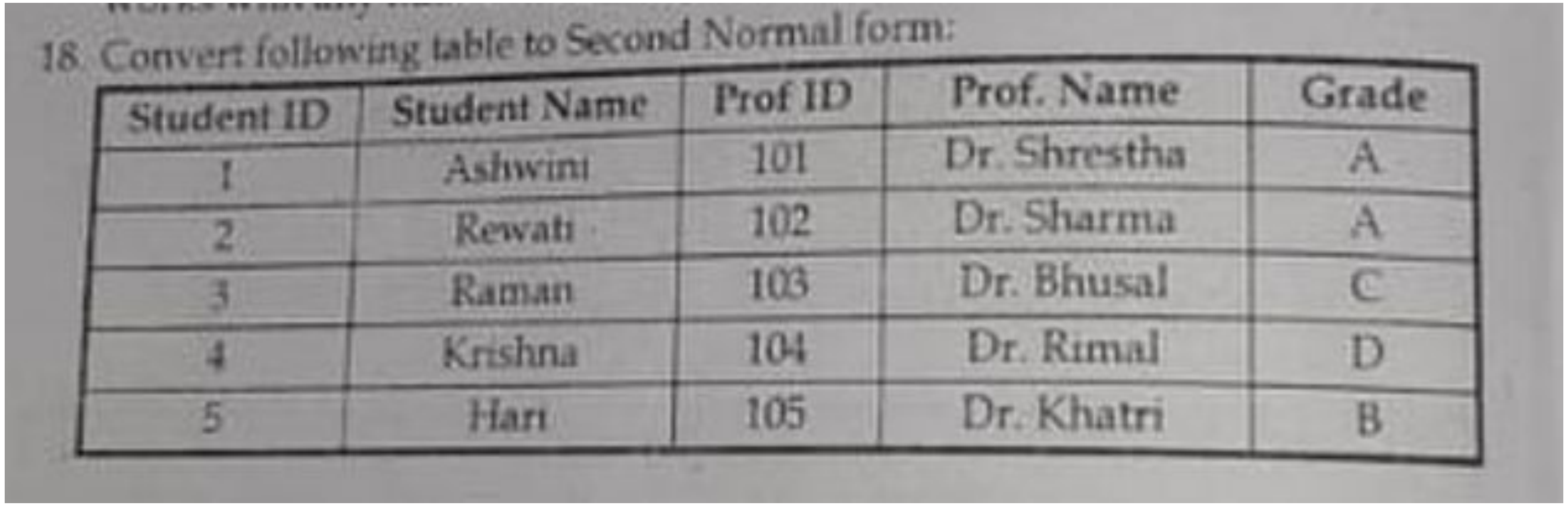
Full Names	Physical Address	Movies rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.



Membership ID	Full Names	Physical Address	Salutation
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

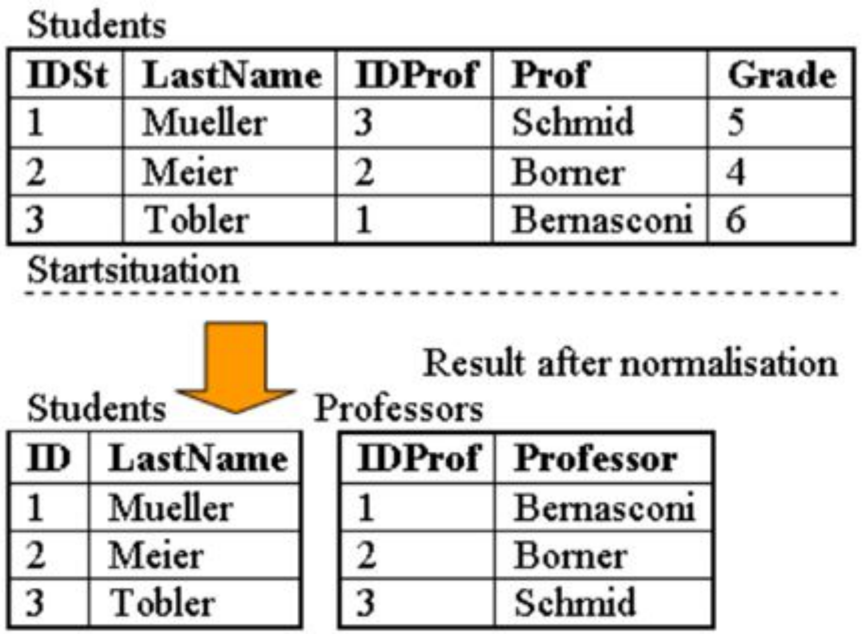


Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans



18. Convert following table to Second Normal form:

Student ID	Student Name	Prof ID	Prof. Name	Grade
1	Ashwini	101	Dr. Shrestha	A
2	Rewati	102	Dr. Sharma	A
3	Raman	103	Dr. Bhusal	C
4	Krishna	104	Dr. Rimal	D
5	Hari	105	Dr. Khatri	B



Students

IDSt	LastName	IDProf	Prof	Grade
1	Mueller	3	Schmid	5
2	Meier	2	Borner	4
3	Tobler	1	Bernasconi	6

Startsituation

Result after normalisation

Students

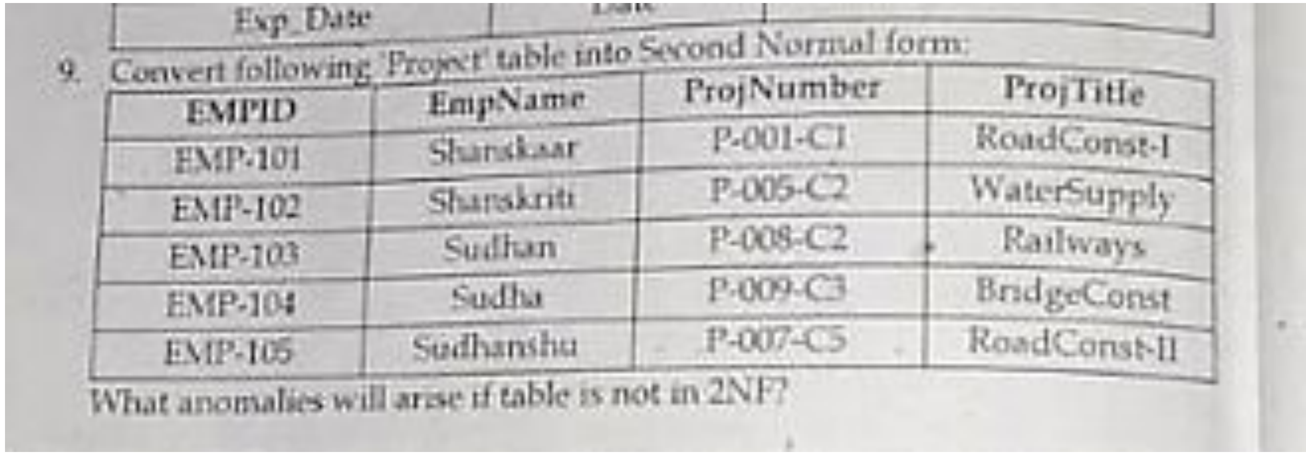
ID	LastName
1	Mueller
2	Meier
3	Tobler

Professors

IDProf	Professor
1	Bernasconi
2	Borner
3	Schmid

IDSt	IDProf	Grade
1	3	5
2	2	4
3	1	6

The table in this example is in first normal form (1NF) since all attributes are single valued. But it is not yet in 2NF. If student 1 leaves university and the tuple is deleted, then we loose all information about professor Schmid, since this attribute is fully functional dependent on the primary key IDSt. To solve this problem, we must create a new table Professor with the attribute Professor (the name) and the key IDProf. The third table Grade is necessary for combining the two relations Student and Professor and to manage the grades. Besides the grade it contains only the two IDs of the student and the professor. If now a student is deleted, we do not loose the information about the professor.



Exp_Date	Date	

9. Convert following 'Project' table into Second Normal form:

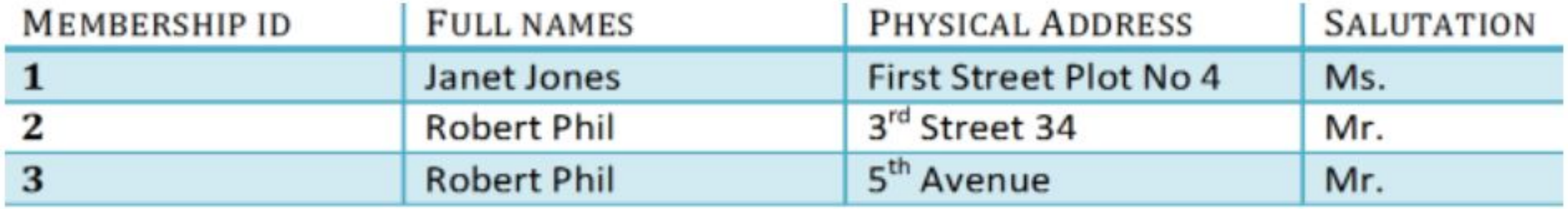
EMPID	EmpName	ProjNumber	ProjTitle
EMP-101	Sharskaar	P-001-C1	RoadConst-I
EMP-102	Sharskriti	P-005-C2	WaterSupply
EMP-103	Sudhan	P-008-C2	Railways
EMP-104	Sudha	P-009-C3	BridgeConst
EMP-105	Sudhanshu	P-007-C5	RoadConst-II

What anomalies will arise if table is not in 2NF?

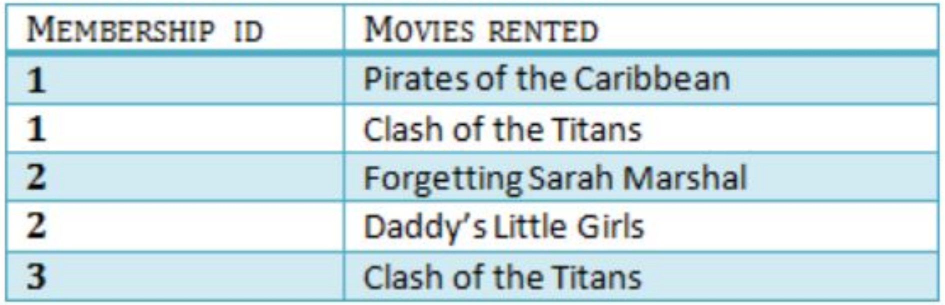
If the table is not in second normal form, redundant data can **cause wasted space and update problems**. Changing multiple rows can make an update cumbersome. Inconsistent data can be another problem if the table is not in second form. Others problems that can occur are when you try to add or delete data to the database.

3NF (Third Normal Form)

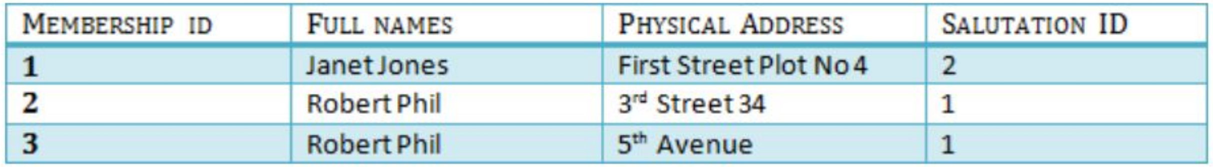
- **Rules**
- Rule 1- Be in 2NF
- Rule 2- Has no transitive functional dependencies, must have
- To move our 2NF table into 3NF, we again need to again divide our table.



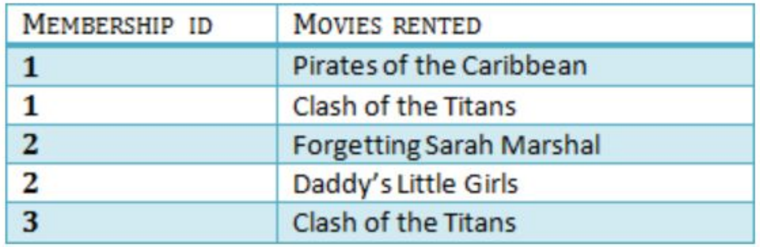
MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.



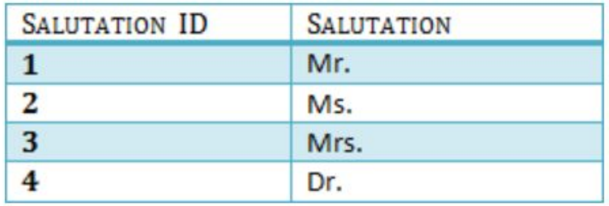
MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans



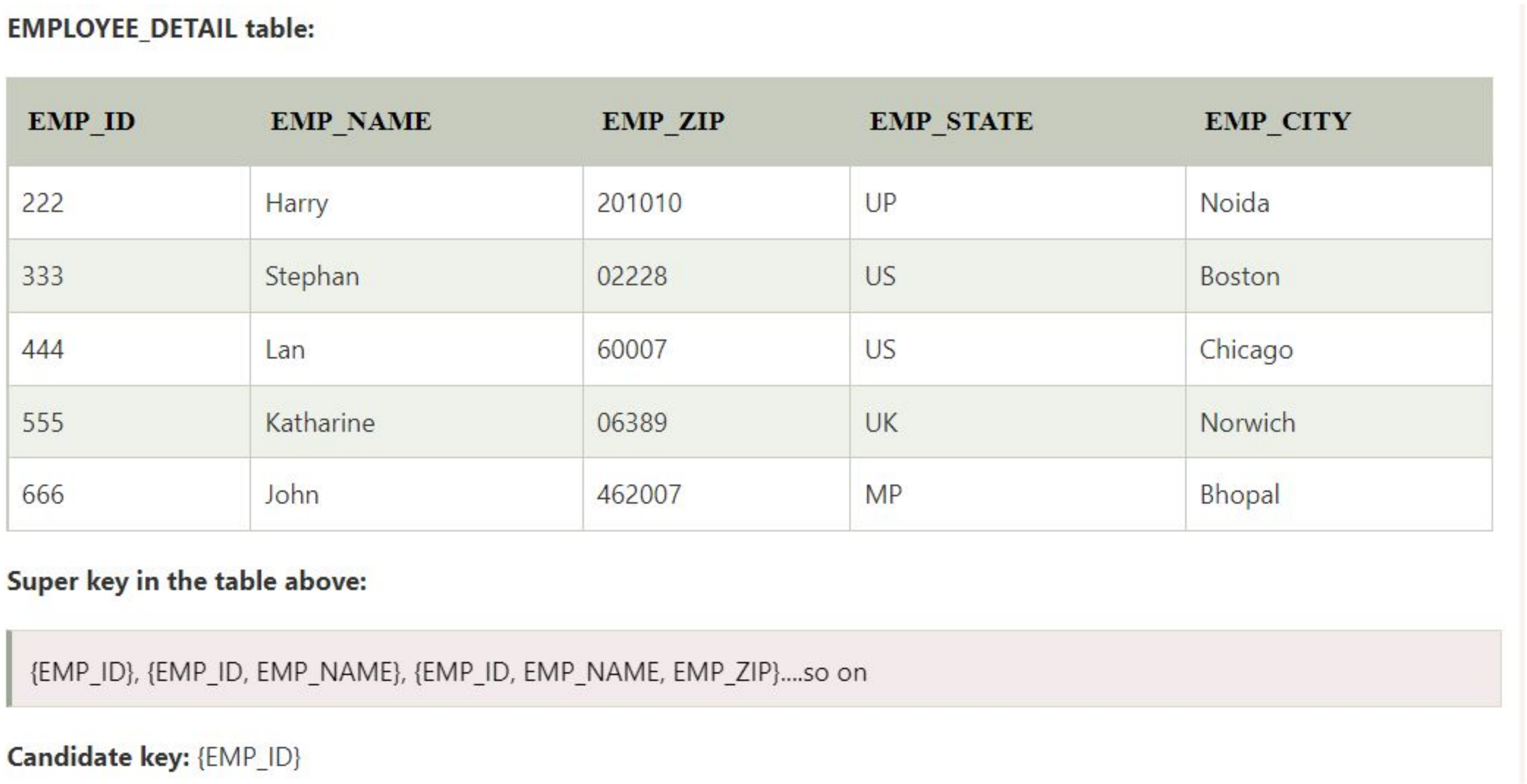
MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1



MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans



SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.



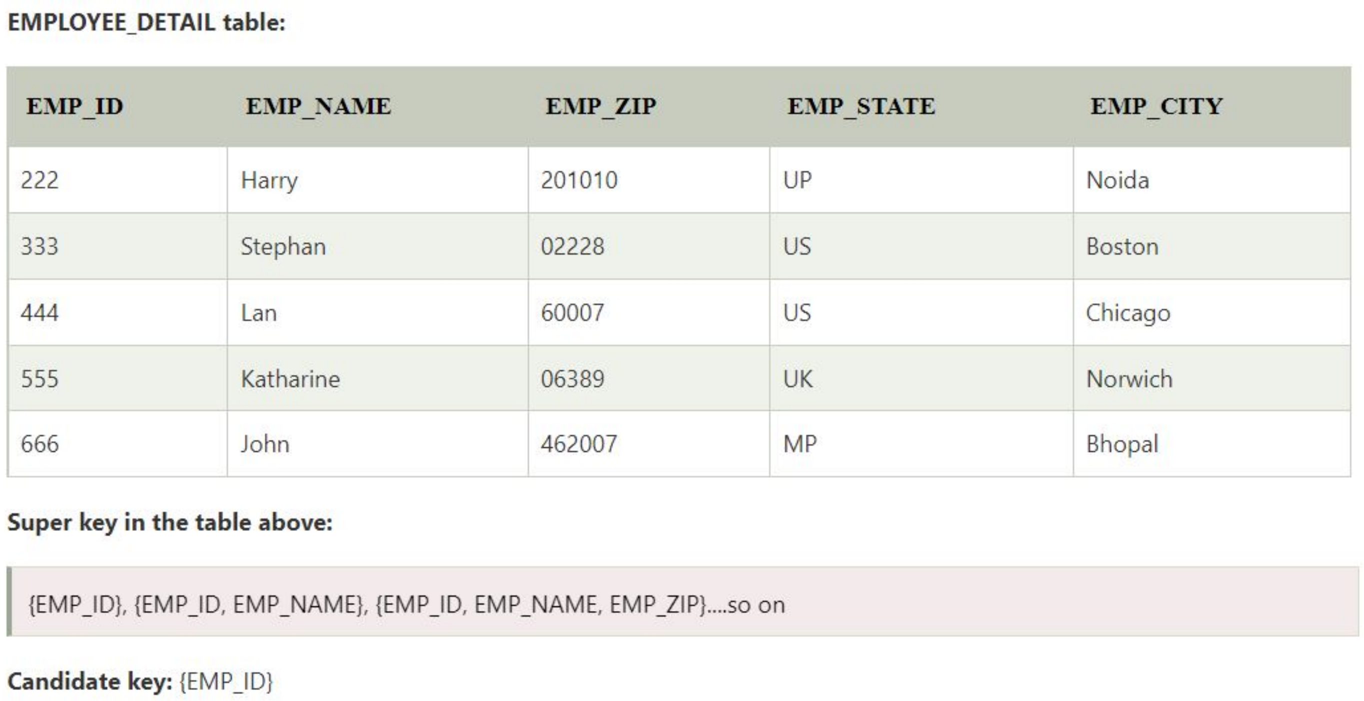
EMPLOYEE_DETAIL table:

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

Super key in the table above:

> {EMP_ID}, {EMP_ID, EMP_NAME}, {EMP_ID, EMP_NAME, EMP_ZIP}....so on

Candidate key: {EMP_ID}



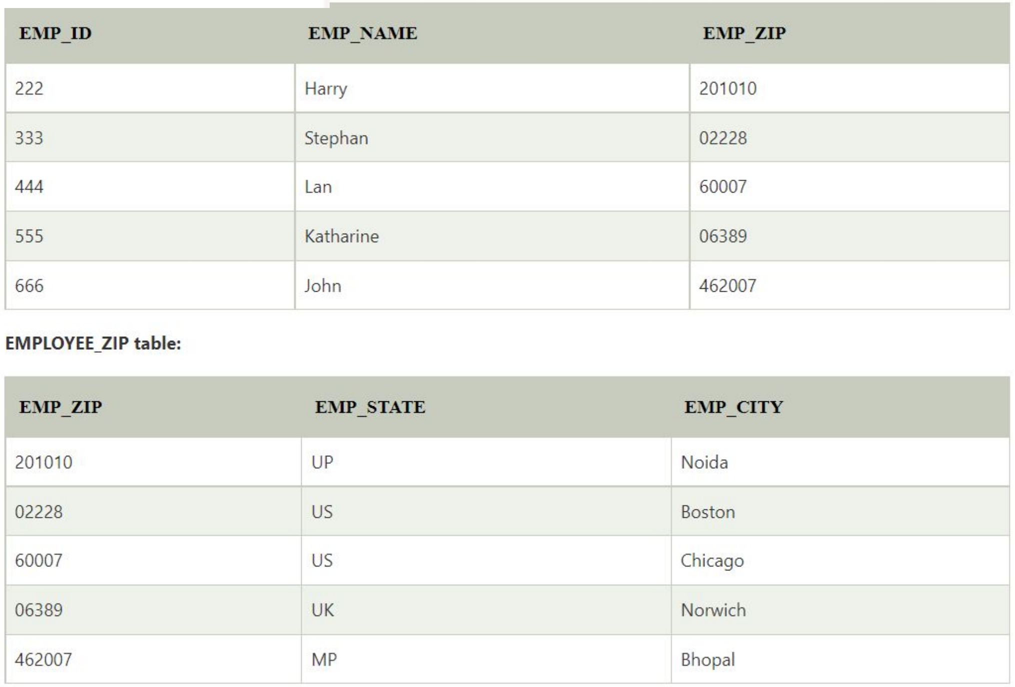
EMPLOYEE_DETAIL table:

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

Super key in the table above:

> {EMP_ID}, {EMP_ID, EMP_NAME}, {EMP_ID, EMP_NAME, EMP_ZIP}....so on

Candidate key: {EMP_ID}

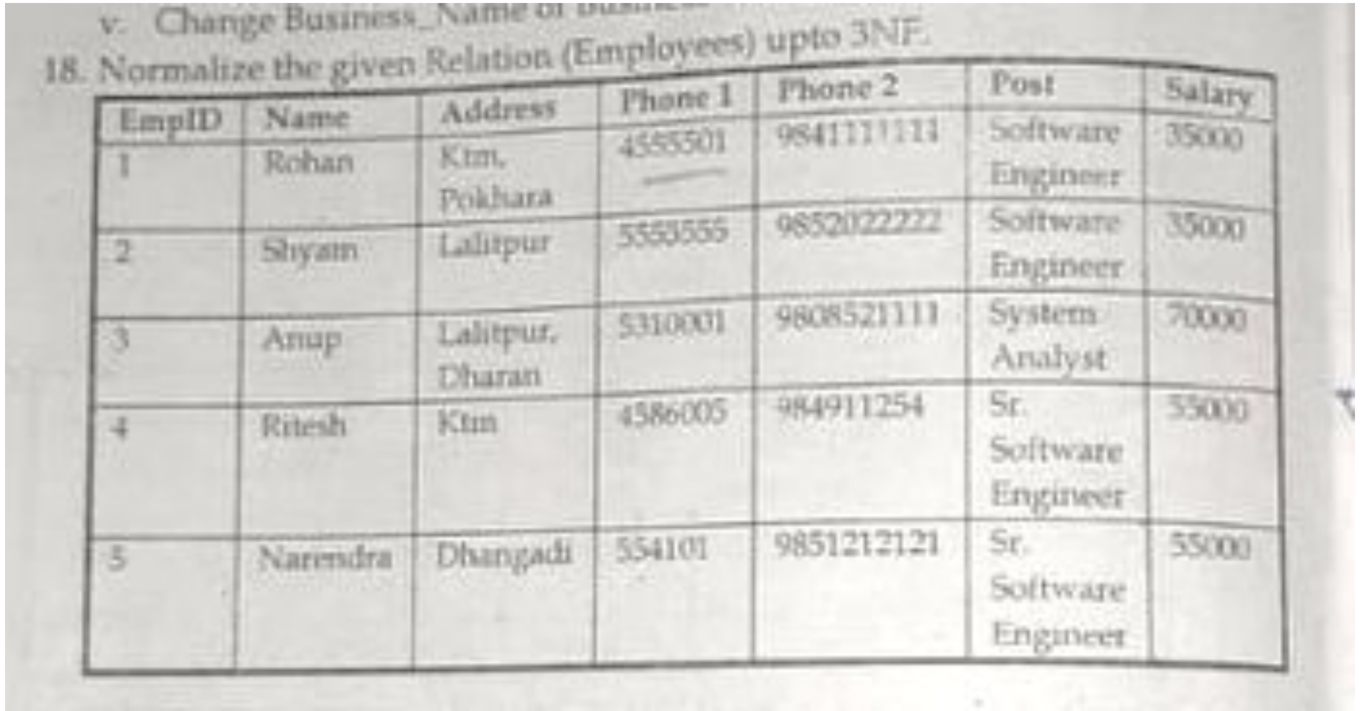


EMP_ID	EMP_NAME	EMP_ZIP
222	Harry	201010
333	Stephan	02228
444	Lan	60007
555	Katharine	06389
666	John	462007

EMPLOYEE_ZIP table:

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

- **BCNF (Boyce-Codd Normal Form)**
- Even when a database is in 3rd Normal Form, still there would be anomalies resulted if it has more than one **Candidate** Key.
- Sometimes is BCNF is also referred as **3.5 Normal Form.**
- **4NF (Fourth Normal Form) Rules**
- If no database table instance contains two or more, independent and multivalued data describing the relevant entity, then it is in 4th Normal Form.
- **5NF (Fifth Normal Form) Rules**
- A table is in 5th Normal Form only if it is in 4NF and it cannot be decomposed into any number of smaller tables without loss of data.



v. Change Business_Name of busi

18. Normalize the given Relation (Employees) upto 3NF.

EmpID	Name	Address	Phone 1	Phone 2	Post	Salary
1	Rohan	Ktm, Pokhara	4555501	9841111111	Software Engineer	35000
2	Shyam	Lalitpur	5555555	9852022222	Software Engineer	35000
3	Anup	Lalitpur, Dharan	5310001	9808521111	System Analyst	70000
4	Ritesh	Ktm	4586005	984911254	Sr. Software Engineer	55000
5	Narendra	Dhangadi	554101	9851212121	Sr. Software Engineer	55000