



DATABASE NORMALIZATION

WHAT IS 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 the smaller table and links them using a relationship.
- The normal form is used to reduce redundancy from the database table.

CONTINUE...

- Normalization organizes the columns and tables of a database to ensure that database integrity constraints properly execute their dependencies.
- It is a systematic technique of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update, and Deletion anomalies.
- In this process, we normalize the table where the data in columns can be fetched with a key.

What is a KEY in SQL?

- Key is a value used to identify the data in records differently.
- SQL key is beneficial when there are various columns in the table, and we need to identify a single or group of columns.
- SQL KEY helps identify the column that requires to be extracted from the database. We can also identify duplicate data in the table by using SQL KEY.
- The relationships among different tables or columns are established by using the SQL key. These relationships create a difference between the tables or columns.
- As we know, SQL keys are used to identify columns uniquely, but some columns don't have a SQL key and can't be identified with a key. So, these columns are called non-key columns.

Database – Primary Key

- The primary key is very useful when we need to identify only one value from the entity. However, the entity may contain various keys, but the most suitable key is called the Primary Key.
- For example, the key that can be used to identify an employee in an Employee table can be Employee ID, as it is different for every entry in the table. So, we can make Employee ID the Primary Key in this case.
- Now the selection of an employee can be made by using the primary key.

Employee
Employee_ID
Employee_NAME
Employee_Salary
Employee_AGE

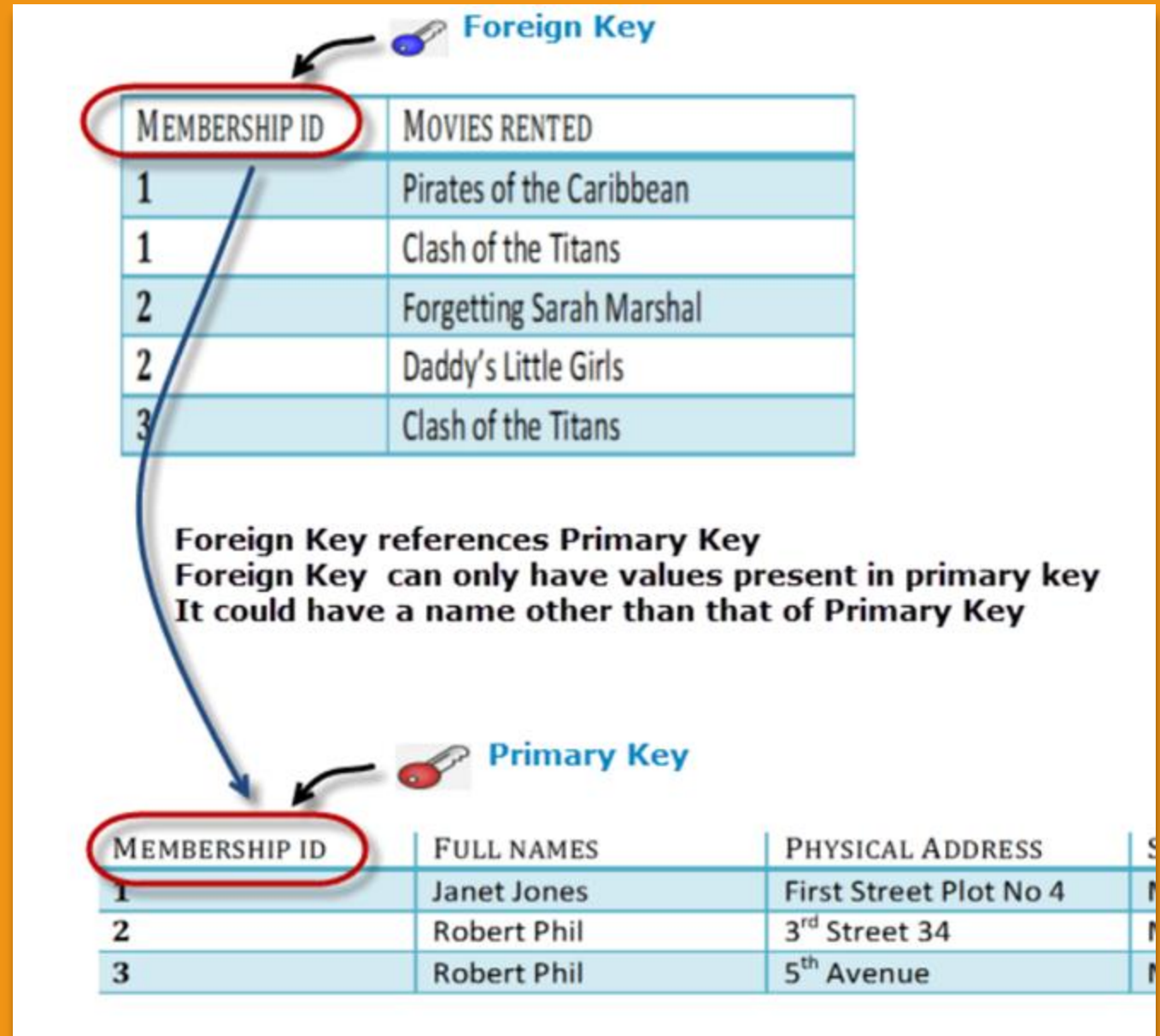
Database – Composite Key

- The composite Key becomes useful when there are more attributes in the Primary Key.
- For example, let us talk about the same table we discussed above the Employee table. Suppose the employees are given different Project Ids and roles that can help them uniquely identify from the table.
- Now, the employee can be identified with the help of any of the keys such as Employee_ID, Employee_ProjectID, or Employee_ROLE. So, there are multiple primary keys.
- When a primary key has more attributes to be considered, it is called a composite key. Therefore, combining all these keys is called Composite Key or Concatenated Key.

Database - 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.
- Foreign keys can be null even though primary keys can not.

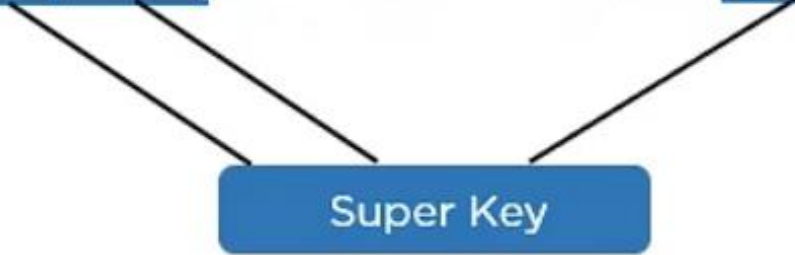


Super Key

- Super key is a set of over one key that can identify a record uniquely in a table, and the Primary Key is a subset of the Super Key.

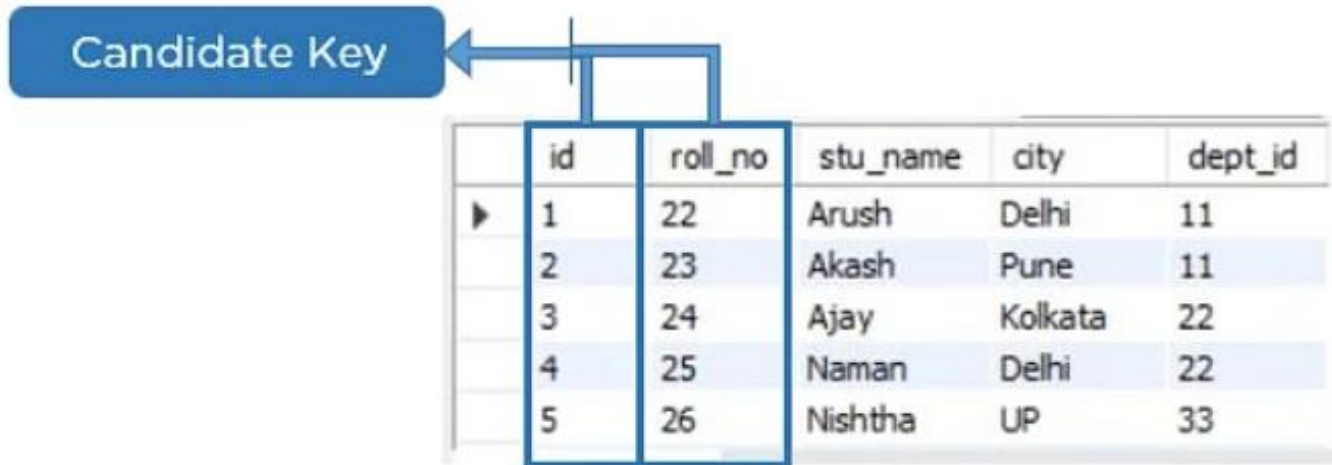
	id	roll_no	stu_name	enroll_no	city	dept_id
▶	1	22	Arush	223	Delhi	11
	2	23	Akash	224	Pune	11
	3	24	Ajay	225	Kolkata	22
	4	25	Naman	226	Delhi	22
	5	26	Nishtha	227	UP	33
	6	27	Lamba	228	Haryana	44

Super Key



Candidate Key

- A candidate key is a set of one or more columns that can identify a record uniquely in a table, and YOU can use each candidate key as a Primary Key.



	id	roll_no	stu_name	city	dept_id
▶	1	22	Arush	Delhi	11
	2	23	Akash	Pune	11
	3	24	Ajay	Kolkata	22
	4	25	Naman	Delhi	22
	5	26	Nishtha	UP	33

What is Anomaly?

- In Database Management System (DBMS), anomaly means the inconsistency occurred in the relational table during the operations performed on the relational table.
- There can be various reasons for anomalies to occur in the database.
- For example, if there is a lot of redundant data present in our database then DBMS anomalies can occur. If a table is constructed in a very poor manner, then there is a chance of database anomaly. Due to database anomalies, the integrity of the database suffers.
- The other reason for the database anomalies is that all the data is stored in a single table. So, to remove the anomalies of the database, normalization is the process that is done where the splitting of the table and joining of the table (different types of join) occurs.

Insertion anomaly

- There are circumstances in which certain facts cannot be recorded at all.
- For example, each record in a "Faculty and Their Courses" relation might contain a Faculty ID, Faculty Name, Faculty Hire Date, and Course Code. Therefore, the details of any faculty member who teaches at least one course can be recorded, but a newly hired faculty member who has not yet been assigned to teach any courses cannot be recorded, except by setting the Course Code to null.

Faculty and Their Courses

Faculty ID	Faculty Name	Faculty Hire Date	Course Code
389	Dr. Giddens	10-Feb-1985	ENG-206
407	Dr. Saperstein	19-Apr-1999	CMP-101
407	Dr. Saperstein	19-Apr-1999	CMP-201

424

Dr. Newsome

29-Mar-2007

?

Update anomaly

- The same information can be expressed on multiple rows; therefore, updates to the relationship may result in logical inconsistencies.
- For example, each record in an "Employees' Skills" relation might contain an Employee ID, Employee Address, and Skill; thus, a change of address for a particular employee may need to be applied to multiple records (one for each skill). If the update is only partially successful - the employee's address is updated on some records but not others - then the relation is left in an inconsistent state. Specifically, the relation provides conflicting answers to the question of what this particular employee's address is.

Employees' Skills

Employee ID	Employee Address	Skill
426	87 Sycamore Grove	Typing
426	87 Sycamore Grove	Shorthand
519	94 Chestnut Street	Public Speaking
519	96 Walnut Avenue	Carpentry

Deletion anomaly

- Under certain circumstances, the deletion of data representing certain facts necessitates the deletion of data representing completely different facts. The "Faculty and Their Courses" relation described in the previous example suffers from this type of anomaly, for if a faculty member temporarily ceases to be assigned to any courses, the last of the records on which that faculty member appears must be deleted, effectively also deleting the faculty member, unless the Course Code field is set to null.

Faculty and Their Courses

Faculty ID	Faculty Name	Faculty Hire Date	Course Code
389	Dr. Giddens	10-Feb-1985	ENG-206
407	Dr. Saperstein	19-Apr-1999	CMP-101
407	Dr. Saperstein	19-Apr-1999	CMP-201

DELETE

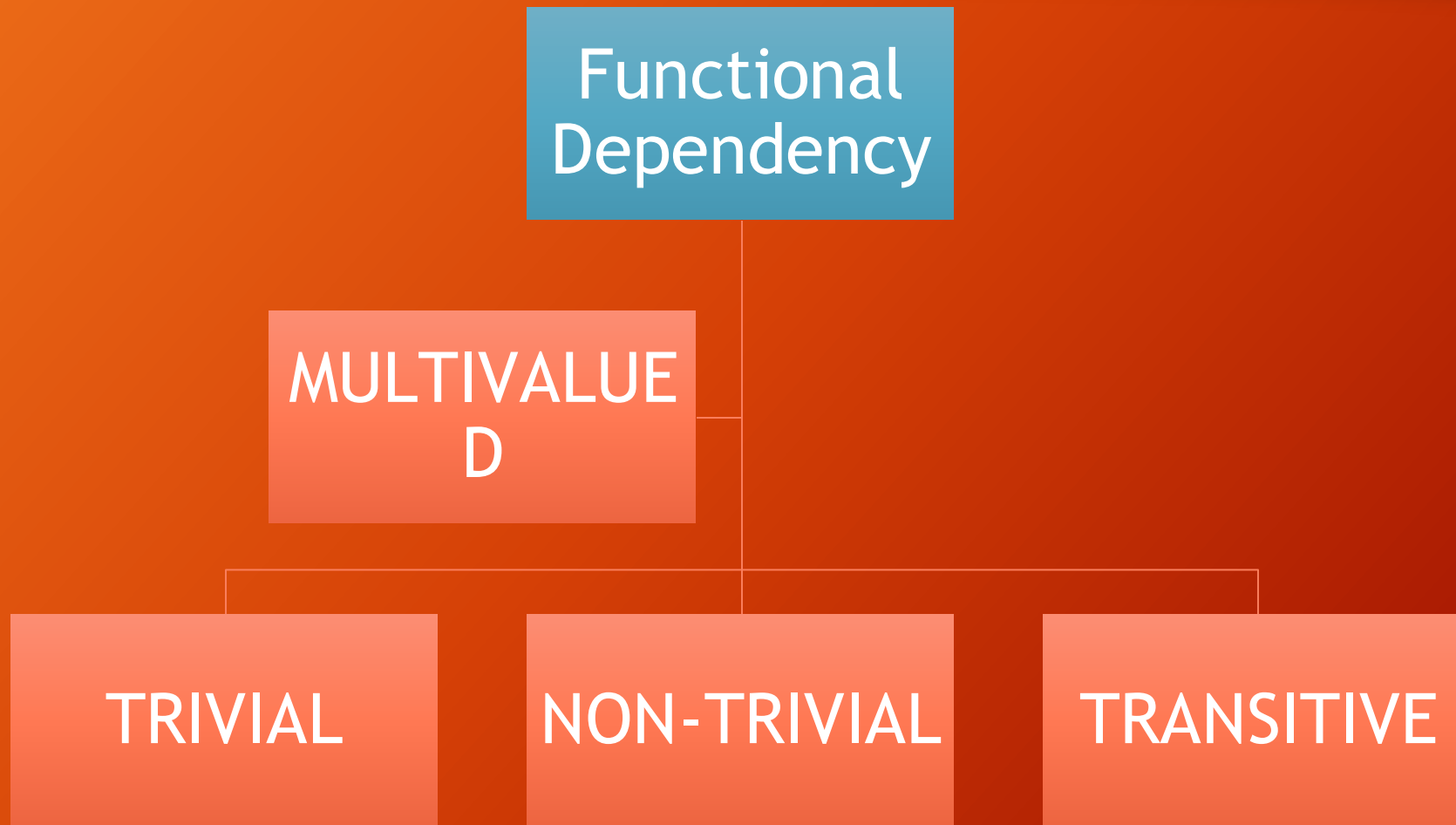
Functional Dependency

- Functional dependency is a relationship that exists between two attributes.
- It typically exists between the primary key and non-key attribute within a table.
- $X \rightarrow Y$
- The left side of FD is known as a determinant, and the right side of the production is known as a dependent.

For example:

- Assume we have an employee table with attributes: Emp_Id, Emp_Name, Emp_Address.
- Here Emp_Id attribute can uniquely identify the Emp_Name attribute of the employee table because if we know the Emp_Id, we can tell that the employee's name is associated with it.
- Functional dependency can be written as:
 - i. $\text{Emp_Id} \rightarrow \text{Emp_Name}$
 - ii. We can say that Emp_Name is functionally dependent on Emp_Id.

Types of Functional dependency



Trivial functional dependency

- $A \rightarrow B$ has trivial functional dependency if B is a subset of A .
- The following dependencies are also trivial like $A \rightarrow A$, $B \rightarrow B$.

Example:

1. Consider a table with two columns `Employee_Id` and `Employee_Name`.
2. $\{\text{Employee_id}, \text{Employee_Name}\} \rightarrow \text{Employee_Id}$ is a trivial functional dependency as `Employee_Id` is a subset of $\{\text{Employee_id}, \text{Employee_Name}\}$.

Emp_id	Emp_name
AS555	Harry
AS811	George
AS999	Kevin

Non-trivial functional dependency

- $A \rightarrow B$ has a non-trivial functional dependency if B is not a subset of A.
- When A intersection B is NULL, then $A \rightarrow B$ is called a complete non-trivial.

Example:

(Company} -> {CEO} (if we know the Company, we know the CEO name)

But the CEO is not a subset of the Company; hence, it's a non-trivial functional dependency.

Company	CEO	Age
Microsoft	Satya Nadella	51
Google	Sundar Pichai	46
Apple	Tim Cook	57

Transitive dependency

- A transitive dependency refers to some non-prime attribute other than the candidate key that depends on another non-prime attribute that is dependent entirely on the candidate key.
- {Company} -> {CEO} (if we know the company, we know its CEO's name)
- {CEO} -> {Age} If we know the CEO, we know the Age
- Therefore, according to the 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 relation to three or more attributes.

Company	CEO	Age
Microsoft	Satya Nadella	51
Google	Sundar Pichai	46
Apple	Tim Cook	57

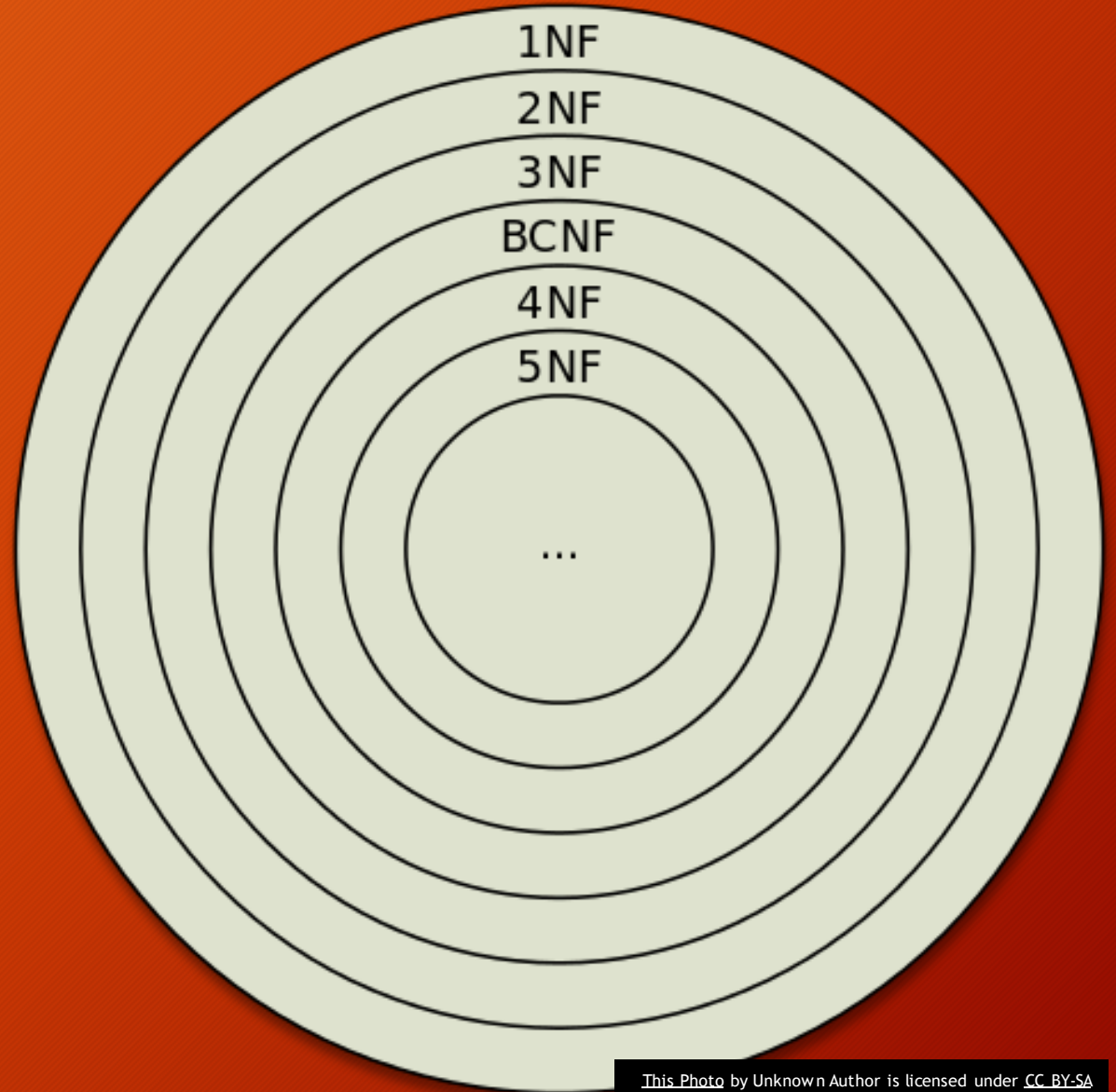
Multivalued dependency

- 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.
- 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 multivalued dependent on car_model.
- This dependence can be represented like this:
car_model -> maf_year
car_model -> colour

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

Types of Normal Forms

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



First Normal Form (1NF)

- A relation will be 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attributes.
- First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

EXAMPLE1.

- Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385, 9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389, 8589830302	Punjab

CONTINUE...

- The decomposition of the EMPLOYEE table into 1NF has been shown

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385	UP
14	John	9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389	Punjab
12	Sam	8589830302	Punjab

EXAMPLE 2.

- Students' record table that has information about student roll number, student name, student course, and age of the student.
- In the student's record table, you can see that the course column has two values. Thus, it does not follow the First Normal Form

	rollno	name	course	age
▶	1	Rahul	c/c++	22
	2	Harsh	java	18
	3	Sahil	c/c++	23
	4	Adam	c/c++	22
	5	Lisa	java	24
	6	James	c/c++	19
*	NULL	NULL	NULL	NULL

CONTINUE...

- Now, if you use the First Normal Form in the above table, you get the given table as a result.

	rollno	name	course	age
►	1	Rahul	c	22
	1	Rahul	c++	22
	2	Harsh	java	18
	3	Sahil	c	23
	3	Sahil	c++	23
	4	Adam	c	22
	4	Adam	c++	22
	5	Lisa	java	24
	6	James	c	19
	6	James	c++	19

Second Normal Form (2NF)

- In the 2NF, relational must be in 1NF.
- In the second normal form, all non-key attributes are fully functionally dependent on the primary key.
- It should be noted that the table should not contain any partial dependency, where partial dependency means a proper subset of the candidate key.

Example:

Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject

CONTINUE...

TEACHER Table

In the given table, the non-prime attribute **TEACHER_AGE** is dependent on **TEACHER_ID** which is a proper subset of a candidate key. That's why it violates the rule for 2NF.

TEACHER_ID	SUBJECT	TEACHER_AGE
25	Chemistry	30
25	Biology	30
47	English	35
83	Math	38
83	Computer	38

CONTINUE...

- To convert the given table into 2NF, we decompose it into two tables:

TEACHER_DETAIL Table:

TEACHER_ID	TEACHER_AGE
25	30
47	35
83	38

TEACHER_SUBJECT Table:

TEACHER_ID	SUBJECT
25	Chemistry
25	Biology
47	English
83	Math
83	Computer

EXAMPLE 2.

- Consider the table Location
- The Location table possesses a composite primary key cust_id, storeid. The non-key attribute is store_location. In this case, store_location only depends on storeid, which is a part of the primary key. Hence, this table does not fulfill the second normal form.

	cust_id	storeid	store_location
▶	1	D1	Toronto
	2	D3	Miami
	3	T1	California
	4	F2	Florida
	5	H3	Texas

CONTINUE...

- To bring the table to its Second Normal Form, we need to split the table into two parts.
- As we have removed the partial functional dependency from the location table, the column store_location entirely depends on the primary key of that table, storeid.

	cust_id	storeid
▶	1	D1
	2	D3
	3	T1
	4	F2
	5	H3

	storeid	store_location
▶	D1	Toronto
	D3	Miami
	T1	California
	F2	Florida
	H3	Texas

Third Normal Form (3NF)

- A relation will be in 3NF if it is in 2NF and does not contain any transitive dependency.
- 3NF is used to reduce data duplication. It is also used to achieve data integrity.
- If there is no transitive dependency for non-prime attributes (which are not a part of the candidate key), then the relationship must be in the third normal form.
- A relation is in third normal form if it holds at least one of the following conditions for every non-trivial function dependency $X \rightarrow Y$.

CONTINUE...

- The Third Normal Form ensures the reduction of data duplication.
- It is also used to achieve data integrity.

CONTINUE...

1. X is a super key.

2. Y is a prime attribute, i.e., each element of Y is part of some candidate key.

Example 1: EMPLOYEE_DETAIL Table

Super key in the table

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

Candidate key:

{EMP_ID}

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

CONTINUE...

- **Non-prime attributes:** In the given table, all attributes except EMP_ID are nonprime.
- Here, EMP_STATE & EMP_CITY dependent on EMP_ZIP and EMP_ZIP dependent on EMP_ID. The non-prime attributes (EMP_STATE, EMP_CITY) are transitively dependent on the super key(EMP_ID). It violates the rule of the third normal form.
- That's why we need to move the EMP_CITY and EMP_STATE to the new table, with EMP_ZIP as a Primary key.

CONTINUE...

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

EXAMPLE 2.

- Student table that has student id, student name, subject id, subject name, and address of the student as its columns.
- Student table, stu_id determines subid, and subid determines sub. Therefore, stu_id determines sub via subid. This implies that the table possesses a transitive functional dependency, and it does not fulfill the third normal form criteria.

	stu_id	name	subid	sub	address
►	1	Arun	11	SQL	Delhi
	2	Varun	12	Java	Bangalore
	3	Harsh	13	C++	Delhi
	4	Keshav	12	Java	Kochi

CONTINUE...

- Now to change the table to the third normal form, you need to divide the table as shown.
- As you can see in both tables, all the non-key attributes are now fully functional, dependent only on the primary key. In the first table, column names, subid, and addresses only depend on stu_id. In the second table, the sub only depends on subid.

	stu_id	name	subid	address
▶	1	Arun	11	Delhi
	2	Varun	12	Bangalore
	3	Harsh	13	Delhi
	4	Keshav	12	Kochi

	subid	subject
▶	11	SQL
	12	java
	13	C++
	12	Java

Boyce Codd Normal Form (BCNF)

- BCNF is the advanced version of 3NF.
- It is stricter than 3NF.
- A table is in BCNF if every functional dependency $X \rightarrow Y$, X is the super key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

CONTINUE...

- **Example:** Let's assume there is a company where employees work in more than one department.

EMP_ID	EMP_COUNTRY	EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

CONTINUE...

In the above table Functional dependencies are as follows:

1. $EMP_ID \rightarrow EMP_COUNTRY$
2. $EMP_DEPT \rightarrow \{DEPT_TYPE, EMP_DEPT_NO\}$

Candidate key:

$\{EMP_ID, EMP_DEPT\}$

- The table is not in BCNF because neither EMP_DEPT nor EMP_ID alone are keys
- To convert the given table into BCNF, we decompose it into three tables.

CONTINUE...

EMP COUNTRY table:	
EMP ID	EMP COUNTRY
264	India
364	UK

Emp_Dept Table:

EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

EMP_ID	EMP_DEPT
D394	283
D394	300
D283	232
D283	549

CONTINUE...

Functional dependencies:

- $EMP_ID \rightarrow EMP_COUNTRY$
- $EMP_DEPT \rightarrow \{DEPT_TYPE, EMP_DEPT_NO\}$

Candidate keys:

- For the first table: EMP_ID
- For the second table: EMP_DEPT
- For the third table: $\{EMP_ID, EMP_DEPT\}$
- Now, this is in BCNF because the left side part of both the functional dependencies is key.

EXAMPLE 2.

- Consider the subject table

The subject table follows these conditions:

1. Each student can enroll in multiple subjects.
2. Multiple professors can teach a particular subject.
3. For each subject, it assigns a professor to the student

- The table is in 1st Normal form as all the column names are unique, all values are atomic, and all the values stored in a particular column are of the same domain.
- The table also satisfies the 2nd Normal Form, as there is no Partial Dependency.
- And there is no Transitive Dependency; hence, the table also satisfies the 3rd Normal Form.

	stuid	subject	professor
▶	1	SQL	Prof. Mishra
	2	Java	Prof. Anand
	2	C++	Prof. Kanth
	3	Java	Prof. James
	4	DBMS	Prof. Lokesh

CONTINUE...

- This table follows all the Normal forms except the Boyce Codd Normal Form.
- As you can see stuid, and subject form the primary key, which means the subject attribute is a prime attribute.
- However, there exists yet another dependency - professor \rightarrow subject.
- BCNF does not follow in the table as a subject is a prime attribute, and the professor is a non-prime attribute.

CONTINUE...

- To transform the table into the BCNF, you will divide the table into two parts. One table will hold stuid which already exists, and the second table will hold a newly created column profid.
- And in the second table will have the columns profid, subject, and professor, which satisfies the BCNF.

	stuid	profid
▶	1	101
	2	102
	2	103
	3	102
	4	104

	profid	subject	professor
▶	1	SQL	Prof. Mishra
	2	Java	Prof. Anand
	2	C++	Prof. Kanth
	3	Java	Prof. James
	4	DBMS	Prof. Lokesh

Fourth Normal Form (4NF)

- A relation will be in 4NF if it is in Boyce Codd's normal form and has no multi-valued dependency.
- For a dependency $A \twoheadrightarrow B$, if for a single value of A, multiple values of B exist, then the relationship will be a multi-valued dependency.
- The given STUDENT table is in 3NF, but the COURSE and HOBBY are two independent entities. Hence, there is no relationship between COURSE and HOBBY.
- In the STUDENT relation, a student with STU_ID 21 contains two courses, Computer and Math, and two hobbies, Dancing and Singing. So, there is a Multi-valued dependency on STU_ID, which leads to unnecessary repetition of data.
- So, to make the above table into 4NF, we can decompose it into two tables

STUDENT		
STU_ID	COURSE	HOBBY
21	Computer	Dancing
21	Math	Singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey

CONTINUE...

STUDENT_COURSE

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics

STUDENT_HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

Fifth normal form (5NF)

- A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.
- 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy.
- 5NF is also known as Project-join normal form (PJ/NF).

CONTINUE...

- In the table, John takes both Computer and Math classes for Semester 1, but he doesn't take a Math class for Semester 2. In this case, a combination of all these fields is required to identify valid data.
- Suppose we add a new Semester as Semester 3 but do not know about the subject and who will be taking that subject, so we leave Lecturer and Subject as NULL. But all three columns together act as a primary key, so we can't leave the other two columns blank.
- So , to make the above table into 5NF, we can decompose it into three relations P1, P2 & P3:

SUBJECT	LECTURER	SEMESTER
Computer	Anshika	Semester 1
Computer	John	Semester 1
Math	John	Semester 1
Math	Akash	Semester 2
Chemistry	Praveen	Semester 1

CONTINUE...

P1

SEMESTER	SUBJECT
Semester 1	Computer
Semester 1	Math
Semester 1	Chemistry
Semester 2	Math

P2

SUBJECT	LECTURER
Computer	Anshika
Computer	John
Math	John
Math	Akash
Chemistry	Praveen

P3

SEMSTER	LECTURER
Semester 1	Anshika
Semester 1	John
Semester 1	John
Semester 2	Akash
Semester 1	Praveen

Advantages of Normalization

- Normalization helps to minimize data redundancy.
- Greater overall database organization.
- Data consistency within the database.
- Much more flexible database design.
- Enforces the concept of relational integrity.

Disadvantages of Normalization

- You cannot start building the database before knowing what the user needs.
- The performance degrades when normalizing the relations to higher normal forms, i.e., 4NF, 5NF.
- It is very time-consuming and difficult to normalize relations of a higher degree.
- Careless decomposition may lead to a bad database design, leading to serious problems.

Thank You