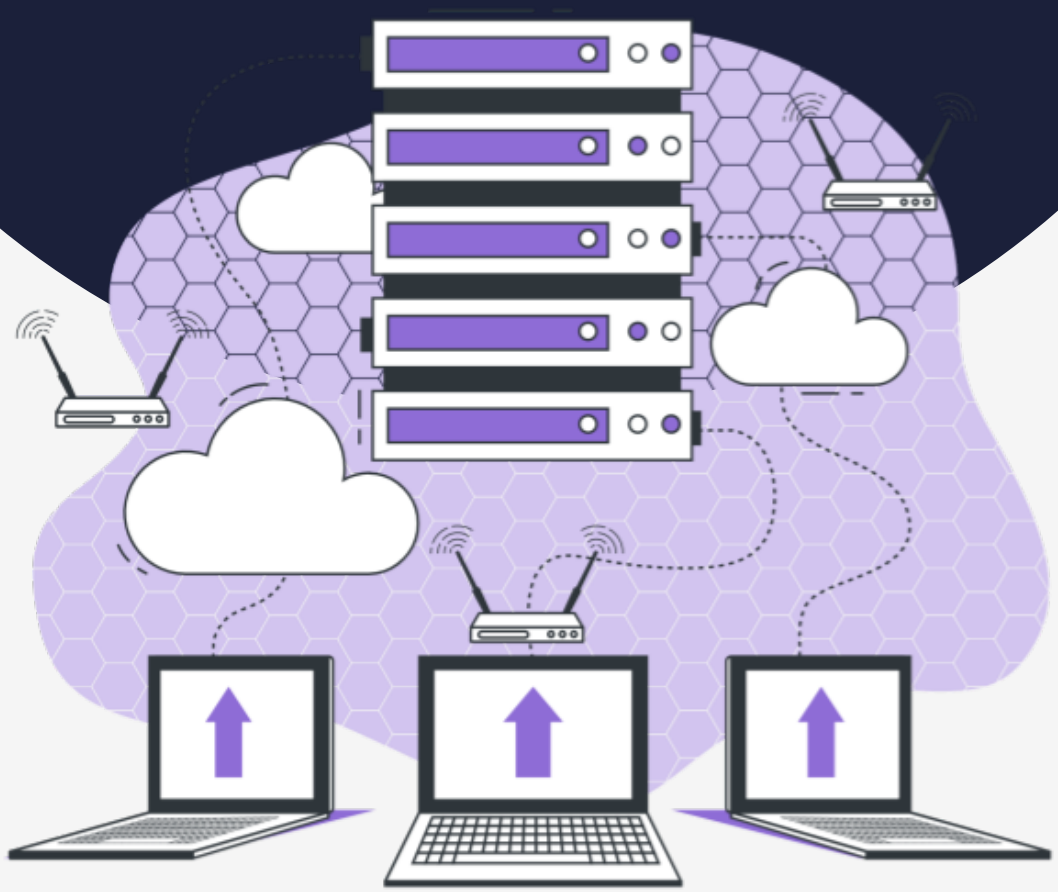
Lesson:

Introduction To Normalization



PW SKILLS

Lecture Checklist

1. Introduction to Single Responsibility Principle.
2. Updation anomaly.
3. Deletion anomaly.
4. Insertion anomaly.
5. Introduction to Normalization.

Normalization is a fundamental concept in Relational Database management systems. From the previous lectures, we have been discussing more about data redundancy. We now know that data redundancy must be avoided in the database design and our database must be ensured with no data redundancy and high maintainability and efficiency.

One of the important ways to remove data redundancy or duplicacy is by Normalization. In this lecture let's have a look at the Single responsibility principle, the problems of data redundancy like updation anomaly, insertion anomaly, and deletion anomaly and finally define what is normalization.

Introduction to Single Responsibility Principle

Before understanding what is a single responsibility principle, have a look at the below table.

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Ganguly
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S105	Rohith	B1	C105	Digital Marketing	Mr. Irfan
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

The above table is of an ed-tech company which has multiple students enrolled, multiple instructors employed and multiple courses launched. The table stores the data of students that is “Std_Id”, “Std_Name” and “Std_Batch”, the data of courses “Course_Id” and Course_Name” and also the data of instructors.

Now if we observe the table carefully, the table designed to store information is not proper. This is not a good practice for storing data. Yes, the RDBMS does not complain if the data is stored as shown in the table but it would lead to a lot of data inconsistency. We will be looking into them in the upcoming sections of the lecture.

To avoid these kinds of problems we have the “Single Responsibility Principle”. The Single Responsibility Principle (SRP) in Relational Database Management Systems (RDBMS) refers to the concept that each database table should have a single responsibility or purpose.

To put it in simple words it is always preferable to have data related to a single entity in one table. The above-shown table contains information of three entities namely student, course, and instructor which is not a good practice and doesn’t follow the single responsibility principle. The best practice is to store data of different entities in different tables and establish a relationship between them.

It is very important to keep in mind the “Single Responsibility Principle” while dealing with RDBMS.

Updation anomaly

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Ganguly
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S105	Rohith	B1	C105	Digital Marketing	Mr. Irfan
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

We know that the above database consists of details of multiple entities such as student, course, and instructor. We also have seen that the database table doesn't follow the single responsibility principle. Let's now look at some of the problems which can be seen if the single responsibility principle is not followed.

Let's have a look at the first row of data which says the student named Dhoni has enrolled in the data science course and is being taught by Mr. Ganguly. Now imagine Mr. Ganguly is not feeling well and the class will be taught by Vishwa, sir. In this case, what we will be doing is we will be replacing the instructor Mr. Ganguly with Mr. Vishwa.

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Vishwa
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S105	Rohith	B1	C105	Digital Marketing	Mr. Irfan
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

Now after replacing the 1st row will contain the data of the student named dhoni who is enrolled in data science and will be taught by Mr. Vishwa.

Here if we observe carefully the two tables, one before replacing the data and one after replacing the data we can clearly see that the data of Mr. Ganguly is lost. The new table has no record of Mr. Ganguly.

Our intention was to update the data but here the changes resulted in the deletion of the data. This is an updation anomaly.

An update anomaly is a problem that can occur in a relational database when modifications (updates) are made to data that result in inconsistent or unexpected changes across the database. In this case, we intended to update the data but resulted in the deletion of the data. This is one of the reasons why we should not be storing multiple entities in a single table.

Deletion anomaly

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Ganguly
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S105	Rohith	B1	C105	Digital Marketing	Mr. Irfan
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

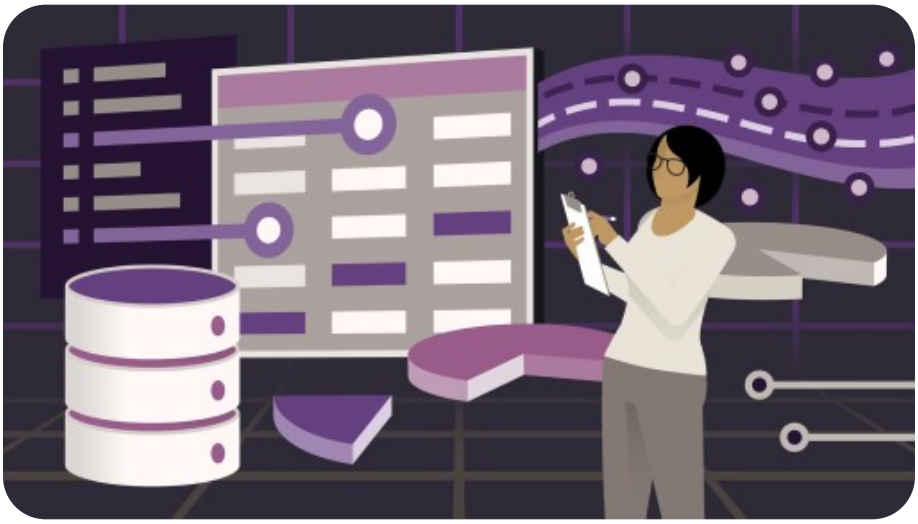
Considering the above table, let's assume the digital marketing course is not performing so well and edtech decides to suspend the course. In this condition, we need to remove row number 5. On removing row number 5 we get the below table.

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Ganguly
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S105	Rohith	B1	C105	Digital Marketing	Mr. Irfan
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

Here if we observe carefully we can find another issue. On comparing the two tables we can clearly identify that the student data of Rohith and the instructor data of Mr. Irfan is lost.

Our intention was to delete just the course but in this process, we lost the data of a student and the instructor. This is called Deletion anomaly.

A deletion anomaly occurs when deleting a row of data from table results in the unintended loss of other data that is still relevant or needed.



Insertion anomaly

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Ganguly
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

After looking at the updation and deletion anomaly now let's have a look of another issue which can be seen if the single responsibility principle is not followed.

Considering the above table, let's assume the company plans to launch a new course. A new course on "IoT", in order to do this we need to add a new entry in the table. If a new entry must be added we need all of the fields such as student name, student id, and course instructor.

If the course has no students enrolled and the instructor is not yet decided then the course cannot be added. This is an insertion anomaly.

An insertion anomaly is a problem that can occur in a relational database when attempting to insert new data into a table, but being unable to do so due to missing or incomplete information.

Normalization

Std_Id	Std_Name	Std_Batch	Course_Id	Course_Name	Course_Instructor
S101	Dhoni	B1	C101	Data Science	Mr. Ganguly
S102	Pandya	B1	C102	Big Data	Mr. Dev
S103	Kohli	B1	C103	Full Stack	Mr. Ishanth
S104	Rahul	B1	C104	Back end	Mr. John
S105	Rohith	B1	C105	Digital Marketing	Mr. Irfan
S106	Bumrah	B1	C106	Data Analyst	Mr. Sachin

Now after looking at some of the issues that can be raised at different conditions on the above-mentioned table, we can tell that the table is not properly designed.

This state where the database has some issues which result in data inconsistency is called the Denormalised table.

The process of fixing the problems in a denormalized table is called normalization.

We will be looking at why normalization and how it will be done in the upcoming lectures.


Normalization in
DBMS