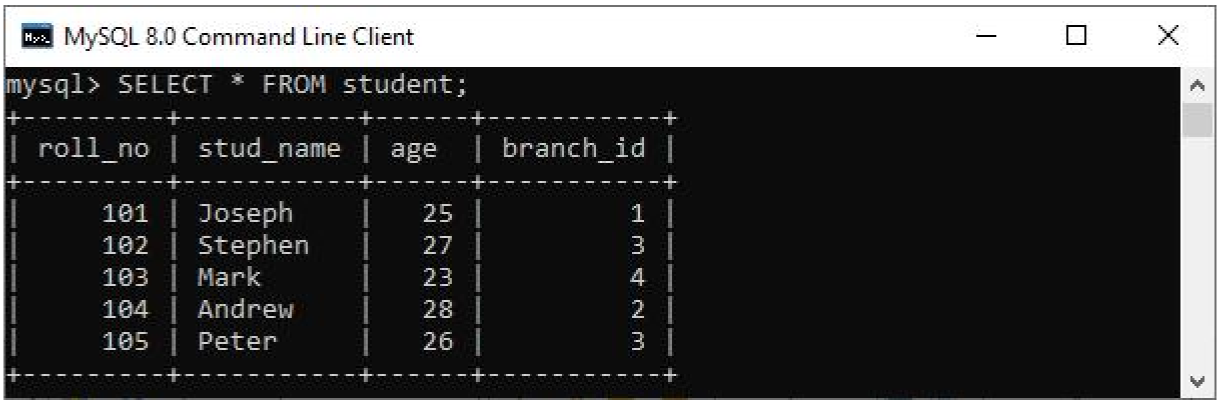
Denormalization in Databases

When we normalize tables, we break them into multiple smaller tables. So when we want to retrieve data from multiple tables, we need to perform some kind of join operation on them. In that case, we use the denormalization technique that eliminates the drawback of normalization.

Denormalization is a technique used by database administrators to optimize the efficiency of their database infrastructure. This method allows us to add redundant data into a normalized database to alleviate issues with database queries that merge data from several tables into a single table. The denormalization concept is based on the definition of normalization that is defined as arranging a database into tables correctly for a particular purpose.

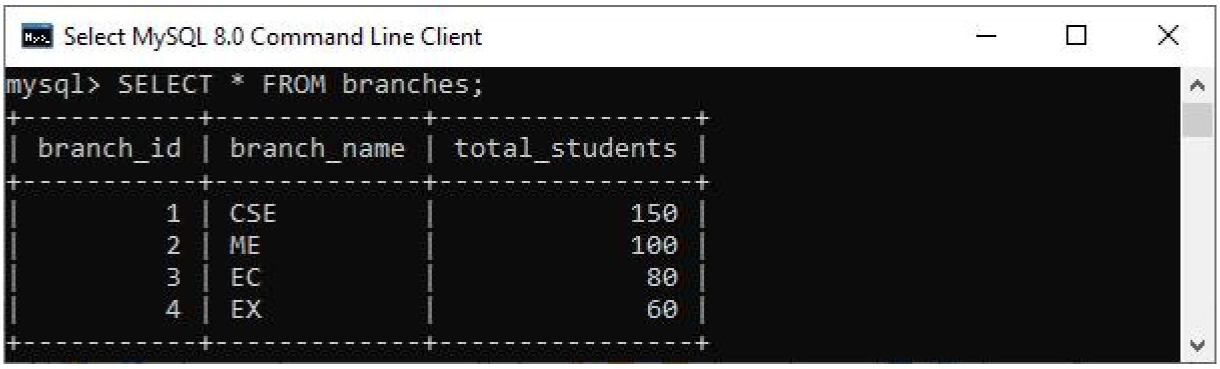
NOTE: Denormalization does not indicate not doing normalization. It is an optimization strategy that is used after normalization has been achieved.

For Example, We have two table students and branch after performing normalization. The student table has the attributes roll_no, stud-name, age, and branch_id.



```
mysql> SELECT * FROM student;
+---------+-----------+------+-----------+
| roll_no | stud_name | age  | branch_id |
+---------+-----------+------+-----------+
|     101 | Joseph    |   25 |         1 |
|     102 | Stephen   |   27 |         3 |
|     103 | Mark      |   23 |         4 |
|     104 | Andrew    |   28 |         2 |
|     105 | Peter     |   26 |         3 |
+---------+-----------+------+-----------+
```

Additionally, the branch table is related to the student table with branch_id as the student table's foreign key.



```
mysql> SELECT * FROM branches;
+-----------+-------------+----------------+
| branch_id | branch_name | total_students |
+-----------+-------------+----------------+
|         1 | CSE         |            150 |
|         2 | ME          |            100 |
|         3 | EC          |             80 |
|         4 | EX          |             60 |
+-----------+-------------+----------------+
```

A JOIN operation between these two tables is needed when we need to retrieve all student names as well as the branch name. Suppose we want to change the student name only, then it is great if the table is small. The issue here is that if the tables are big, joins on tables can take an excessively long time.

In this case, we'll update the database with denormalization, redundancy, and extra effort to maximize the efficiency benefits of fewer joins. Therefore, we can add the branch name's data from the Branch table to the student table and optimizing the database.

Pros of Denormalization

The following are the advantages of denormalization:

1. Enhance Query Performance

Fetching queries in a normalized database generally requires joining a large number of tables, but we already know that the more joins, the slower the query. To overcome this, we can add redundancy to a database by copying values between parent and child tables, minimizing the number of joins needed for a query.

2. Make database more convenient to manage

A normalized database is not required calculated values for applications. Calculating these values on-the-fly will take a longer time, slowing down the execution of the query. Thus, in denormalization, fetching queries can be simpler because we need to look at fewer tables.

3. Facilitate and accelerate reporting

Suppose you need certain statistics very frequently. It requires a long time to create them from live data and slows down the entire system. Suppose you want to monitor client revenues over a certain year for any or all clients. Generating such reports from live data will require "searching" throughout the entire database, significantly slowing it down.

Cons of Denormalization

The following are the disadvantages of denormalization:

- It takes large storage due to data redundancy.
- It makes it expensive to updates and inserts data in a table.
- It makes update and inserts code harder to write.
- Since data can be modified in several ways, it makes data inconsistent. Hence, we'll need to update every piece of duplicate data. It's also used to measure values and produce reports. We can do this by using triggers, transactions, and/or procedures for all operations that must be performed together.

How is denormalization different from normalization?

The denormalization is different from normalization in the following manner:

- Denormalization is a technique used to merge data from multiple tables into a single table that can be queried quickly. Normalization, on the other hand, is used to delete redundant data from a database and replace it with non-redundant and reliable data.
- Denormalization is used when joins are costly, and queries are run regularly on the tables. Normalization, on the other hand, is typically used when a large number of insert/update/delete operations are performed, and joins between those tables are not expensive