

## **Denormalization in Database**

### **Introduction**

Denormalization is a database optimization technique in which we add redundant data to one or more tables. This can help us avoid costly joins in a relational database. Note that denormalization does not mean not doing normalization. It is an optimization technique that is applied after doing normalization.

In a traditional normalized database, we store data in separate logical tables and attempt to minimize redundant data. We may strive to have only one copy of each piece of data in database. For example, in a normalized database, we might have a Courses table and a Teachers table. Each entry in Courses would store the teacherID for a Course but not the teacherName. When we need to retrieve a list of all Courses with the Teacher name, we would do a join between these two tables.

In some ways, this is great; if a teacher changes his or her name, we only have to update the name in one place.

The drawback is that if tables are large, we may spend an unnecessarily long time doing joins on tables.

Denormalization, then, strikes a different compromise. Under denormalization, we decide that we're okay with some redundancy and some extra effort to update the database in order to get the efficiency advantages of fewer joins.

#### **Pros of Denormalization:-**

1. Retrieving data is faster since we do fewer joins
2. Queries to retrieve can be simpler (and therefore less likely to have bugs), since we need to look at fewer tables.

#### **Cons of Denormalization:-**

1. Updates and inserts are more expensive.
2. Denormalization can make update and insert code harder to write.
3. Data may be inconsistent. Which is the "correct" value for a piece of data?
4. Data redundancy necessitates more storage.

In a system that demands scalability, like that of any major tech companies, we almost always use elements of both normalized and denormalized databases.

## Uses of Denormalization

Databases intended for Online Transaction Processing (OLTP) are typically more normalized than databases intended for Online Analytical Processing (OLAP). OLTP Applications are characterized by a high volume of small transactions such as updating a sales record at a super market checkout counter. The expectation is that each transaction will leave the database in a consistent state. By contrast, databases intended for OLAP operations are primarily "read mostly" databases. OLAP applications tend to extract historical data that has accumulated over a long period of time. For such databases, redundant or "denormalized" data may facilitate Business Intelligence applications. Specifically, dimensional tables in a star schema often contain denormalized data.

## Denormalization in Database

Do you know how many different ways are there to design a database? What will do if you want to optimize the database on your website, so its performance gets a boost? If not, then this article is specially made for you. Here, we will talk about the denormalization in databases. You will get complete information about this technique for restoring the database and improve the performance of the website.

### ***What exactly denormalization is?***

Denormalization can define as the technique used for normalizing the existing database for giving a boost to the performance of the site. The approach is to make an addition of redundant data where it needed the most. There are many extra attributes used in a present table along with adding new tables.

Furthermore, the instances of existing tables also created in this technique. The main motive for this strategy is to minimize the running time of chosen queries so that the queries start getting better access to the data.

### ***What are the major database denormalization techniques?***

There are plenty of techniques through which the database can be denormalized. We have explained about each of them in brief below:

- ***Storing derivable information***

This technique is perfect to follow if you want to execute calculation repeatedly while queries. The derived calculation needs to be stored in the master table if there are detail records in it. They are perfect to use if you require derivable values more frequently or you don't prefer to alter source values regularly.

- ***Moving data fields into a supertype***

In this type of technique, the data fields of all the subtypes are moved into supertypes. The new model looks completely different from the normalized

model. The subtypes are completely removed from the data fields using this technique.

- ***Using hardcore values***

This technique is perfect to use if there are reference table along with consistent records. There will be no need to join the tables for getting the reference values. You need to check out the constraint for validating values with this technique. You have to rewrite a new value each time in the table.

- ***Copying data field from an entity***

In this technique, there is no removal of the entity performed by the user. Instead, the data fields copied from one entity to another. This technique consumes very less time and boosts the retrieval speed.

### ***Why should denormalization need to be used in the database?***

It is important that you should have proper knowledge about why this technique needed to be used and when. Let's talk about the situations where the denormalization technique should be adopted.

- ***Maintaining History***

There is no doubt that the data changes over the time making the requirement to store valid values during the creation of the record. Some of you might be thinking what these changes can be. For instance, the first & last name of an individual can change, or modification even occurs in the business address of a client. The past data can't be recreated correctly if the latest details don't get added.

This problem can eliminate by making an addition of a table having the history of these modifications. In that case, denormalization can turn out to be very handy and make the process less complicated.

- ***Making query performance better***

Many queries use the multiple tables for accessing the data that used on a regular basis. For example, you have to join five tables for returning the name of clients and items delivered to them. Lots of tables along the path may have a large quantity of data. In that scenario, the **customer\_id** attribute should be added directly to the **items\_sold** table.

- ***Facilitating reporting***

Plenty of applications demand analytical and statistical details. However, there is lots of time consumed in creating reports using live data that affect the overall performance negatively. At that time, it will be wise to use the denormalization process for saving analytical details instead of calculating this data many times.

### ***What are the major advantages of denormalization?***

There are plenty of advantages offered by the denormalization in the database. You might have already understood it by checking the things mentioned above. Let's allow to tell you about them in detail below:

- ***Retrieving data***

The users have to join very fewer tables that boost up the process and make it less time-consuming. It can be very handy while storing the data.

- ***Tables reduction***

There are many instances where the number of tables gets reduced by using denormalization.

### ***What are the drawbacks of denormalization in the database?***

There is a great chance that you may be thinking on why one should go for denormalizing or not. After reading this article till now, you should have assumed that it is the ideal method for boosting the performance of the database.

However, there is some drawbacks associated with this technique about which you need to stay aware. We have written about each of them in detail below:

- ***Possible data anomalies***

The amount of data needed to get modified increased during the process of denormalization. That's why you have to check out about all the cases of duplicate data carefully. There is a requirement to utilize stored procedures, triggers, and transactions for avoiding data anomalies.

- ***Affecting other operation's speed***

The denormalization affects the speed of many other operations like data insertion, deletion, and modification. It is not common, but a complex query can slow down the complete system. It may be a reason to worry for some programmers and organizations.

- ***Additional storage space***

A large amount of data needs to be duplicated during the denormalization process in the database. It results in demanding more storage space than normal.

- ***Extra coding***

The selected queries modified in this process that no doubt brings plenty of advantages. However, you have to pay its price in the form of extra coding. Additionally, you have to make an update in the values of new attributes that added to the present records. You need a little bit of extra coding for that purpose.

- ***Extra documentation***

It is very important to document all the steps in the denormalization process. Furthermore, the rules have to be revised if any modification in the database design happened later. It will result in increasing the documentation process.

***Things to keep in mind during the denormalization process***

Denormalization is a very complicated process that demands pretty serious effort and expertise. That's why it is important that you should follow the below mentioned tips for performing this task adequately.

- Always target those components in the database that you want to boost. You should avoid trying denormalizing the complete database.
- You should have adequate learning about the logical design of the application to have a complete idea about the components affected greatly by the denormalization.
- Knowing the data storage techniques will be a plus point. It will help in picking you the most relevant one.
- The modification of data in the application needs to be analyzed carefully. The reason behind it is that the maintenance of the database's integrity becomes difficult if data changes too frequently.

## Conclusion

We suggest you don't rely on the denormalization process too much and only going for it when the database requires performing better at some tasks. Creating a clean and highly defined normalized database can be greatly beneficial for a longer run. There is a need to stay pretty careful during the denormalization process as it is important to document all the modifications in the database.

## Reference

- [www.google.com](http://www.google.com)
- [www.wikipedia.com](http://www.wikipedia.com)
- [www.studymafia.org](http://www.studymafia.org)