# Denormalization

1. **What is denormalization?**

Denormalization is a database optimization technique. As we all know, Normalization is that we store our data in separate logical tables and attempt to minimize redundant data. We are striving to have only one copy of each piece of data in our database. This is great because if we need to change data we only need to update the data in one place. However, is this always a good way to achieve a good performance? The answer is not necessarily! Imaging that, if you need to retrieve some data from a lot of separate tables, you have to use some joins, and sometimes group bys in your SQL. If the tables are large, there's gonna be a long time for doing joins. So, that's why sometimes we need to add redundant data to one or more tables to help us avoid expensive joins so that we can achieve better performance. This is called Denormalization. Denormalization does NOT mean no normalization in our database, it actually means we can do some denormalization after doing normalization. Under denormalization, it is okay to have some redundancy and to take extra effort to update the database to get the efficiency advantages of fewer joins.

Pros and Cons:

| Pros | Cons |
|---|---|
| Retrieving data is faster (fewer joins) | Updates and inserts are expensive |
| SQL can be simpler (and therefore less likely to have bugs) | Data may be inconsistent if without good maintenance |
| | Need more storage |

2. **Example**

Table: Instructor

| instructorId | firstName | lastName | age | sex | …etc |
|---|---|---|---|---|---|
| 1 | Judy | Wang | 29 | f | … |

| | | | | | |
|---|---|---|---|---|---|
| 2 | Ellen | Yang | 32 | f | … |
| 3 | Kevin | Lee | 40 | m | … |
| … | … | … | … | … | … |

Table: Course

| courseId | name | description | category | instructorId | …etc |
|---|---|---|---|---|---|
| 1 | Database | descip1 | CS | 2 | … |
| 2 | Algorithm | descrip2 | CS | 3 | … |
| 3 | Machine learning | descrip3 | CS | 1 | … |
| … | … | … | … | … | … |

If we need to retrieve all the courses info with instructors' names. We can do something like this:

```
select c.*, t.firstName, t.lastName from Course c left join Teacher t on
c.instructorId == t.instructorId;
```

If we do Denormalization, we can add a column called "instructorName" to table Course. It is gonna be like this:

| courseId | name | description | category | instructorId | instructorName | …etc |
|---|---|---|---|---|---|---|
| 1 | Database | descip1 | CS | 2 | Ellen Yang | … |
| 2 | Algorithm | descrip2 | CS | 3 | Kevin Lee | … |
| 3 | Machine learning | descrip3 | CS | 1 | Judy Wang | … |
| … | … | … | … | … | … | … |

Now, we can do sql like this:

```
select * from Course.
```

It is much simpler, right?

3. **Reference**

- https://www.geeksforgeeks.org/denormalization-in-databases/