DEFINITION

database normalization

Posted by: Margaret Rouse WhatIs.com

Follow:

Contributor(s): Margaret Rouse & Jack Vaughan

Database normalization is the process of organizing data into tables in such a way that the results of using the database are always unambiguous and as intended. Such normalization is intrinsic to relational database theory. It may have the effect of duplicating data within the database and often results in the creation of additional tables.

The concept of database normalization is generally traced back to E.F. Codd, an IBM researcher who, in 1970, published a paper describing the relational database model. What Codd described as "a normal form for database relations" was an essential element of the relational technique. Such data normalization found a ready audience in the 1970s and 1980s -- a time when disk drives were quite expensive and a highly efficient means for data storage was very necessary. Since that time, other techniques, including denormalization, have also found favor.

While data normalization rules tend to increase the duplication of data, it does not introduce data redundancy, which is unnecessary duplication. Database normalization is typically a refinement process after the initial exercise of identifying the data objects that should be in the relational database, identifying their relationships and defining the tables required and the columns within each table.

Simple data normalization example

Customer	Item purchased	Purchase price
Thomas	Shirt	$40
Maria	Tennis shoes	$35
Evelyn	Shirt	$40

Pajaro	Trousers	$25

If this table is used for the purpose of keeping track of the price of items and you want to delete one of the customers, you will also delete the price. Normalizing the data would mean understanding this and solving the problem by dividing this table into two tables, one with information about each customer and the product they bought and the second with each product and its price. Making additions or deletions to either table would not affect the other.

Normalization degrees of relational database tables have been defined and include:

First normal form (1NF). This is the "basic" level of database normalization, and it generally corresponds to the definition of any database, namely:

- It contains two-dimensional tables with rows and columns.
- Each column corresponds to a subobject or an attribute of the object represented by the entire table.
- Each row represents a unique instance of that subobject or attribute and must be different in some way from any other row (that is, no duplicate rows are possible).
- All entries in any column must be of the same kind. For example, in the column labeled "Customer," only customer names or numbers are permitted.

Second normal form (2NF). At this level of normalization, each column in a table that is not a determiner of the contents of another column must itself be a function of the other columns in the table. For example, in a table with three columns containing the customer ID, the product sold and the price of the product when sold, the price would be a function of the customer ID (entitled to a discount) and the specific product.

Third normal form (3NF). At the second normal form, modifications are still possible because a change to one row in a table may affect data that refers to this information from another table. For example, using the customer table just cited, removing a row describing a customer purchase (because of a return, perhaps) will also remove the fact that the product has a certain price. In the third normal form, these tables would be divided into two tables so that product pricing would be tracked separately.

Extensions of basic normal forms include the domain/key normal form, in which a key uniquely identifies each row in a table, and the Boyce-Codd normal form, which refines and enhances the techniques used in the 3NF to handle some types of anomalies.

Database normalization's ability to avoid or reduce data anomalies, data redundancies and data duplications, while improving data integrity, have made it an important part of the data developer's toolkit for many years. It has been one of the hallmarks of the relational data model.

The relational model arose in an era when business records were, first and foremost, on paper. Its use of tables was, in some part, an effort to mirror the type of tables used on paper that acted as

the original representation of the (mostly accounting) data. The need to support that type of representation has waned as digital-first representations of data have replaced paper-first records.

But other factors have also contributed to challenging the dominance of database normalization.

Over time, continued reductions in the cost of disk storage, as well as new analytical architectures, have cut into normalization's supremacy. The rise of denormalization as an alternative began in earnest with the advent of data warehouses, beginning in the 1990s. More recently, document-oriented NoSQL databases have arisen; these and other nonrelational systems often tap into nondisk-oriented storage types. Now, more than in the past, data architects and developers balance data normalization and denormalization as they design their systems.

Margaret Rouse asks:

How do you think data normalization is best applied toda as per-bit disk drive prices decline and new storage technologies come online?

Join the Discussion

This was last updated in September 2016

Continue Reading About database normalization

- Learn IT: Unleashing the Power of the Database has more information.
- Important rules for SQL Server normalization
- Say hello to the normal forms
- When not to normalize your SQL database
- E.F. Codd's groundbreaking paper on relational methods and data normalization (PDF)

- A look at where normalization fails the designer

Related Terms

database replication

Database replication is the frequent electronic copying of data from a database in one computer or server to a database in ... See complete definition

flat file

A flat file contains records that have no structured interrelationship. A flat file typically consists of a text file, from which... See complete definition

foreign key

A foreign key is a column or columns of data in one table that connects to the primary key data in the original table. See complete definition

Dig Deeper on SQL Server Database Modeling and Design

ALL NEWS GET STARTED EVALUATE MANAGE PROBLEM SOLVE

Azure Cosmos DB features, pricing morph with new provisioning model

Why running SQL Server on Docker is no longer frowned upon

Microsoft Cosmos DB takes Azure databases to a higher level

SQL Server for Linux gets closer -- Windows database adds HA traits

Load More

Join the conversation

4 comments

Share your comment

Send me notifications when other members comment.

Add My Comment

Oldest ▼

[-] **Margaret Rouse**

- 30 Sep 2016 11:48 AM

How do you think data normalization is best applied today, as per-bit disk drive prices decline and new storage technologies come online?

Reply

[-] **Steveo250k**

- 3 Oct 2016 7:52 AM

In your article I'm not sure I believe your logic. In the first paragraph you state normalization "...may have the effect of duplicating data within the database and often results in the creation of additional tables."

But my understanding is the goal of normalization is to reduced, to the point of eliminating, duplication of data. (https://en.wikipedia.org/wiki/Database_normalization).

But then in the 4th paragraph from the end you say, "Database normalization's ability to avoid or reduce data anomalies, data redundancies and data duplications, ..."

Which is it: does normalization reduce or increase duplication? It's the former.

You also state normalization increases the number of tables. Sure it does. Is that a problem? It's like saying Object Oriented programming increases the number of objects. Of course it does. That's the point. A database solution is a model of some reality. Reality has many parts. Therefore the model will have many parts. The closer the model to reality the more parts the model will have.

Reply

[-] **Munqath**

- 20 Sep 2017 6:40 AM

Exactly Steveo250k, I got the same doubts and I find your comment just clarified the things.

Reply

[-] **ahmedseid**

- 22 May 2017 11:00 AM

is the best idea that ensures efficiency and effectiveness.

Reply

Self-service BI software opens the door to wider analytics uses in organizations. This handbook compares top self-service tools ...

Tableau vs. Power BI vs. Qlik: How the BI rivals stack up

Tableau, Power BI and Qlik Sense offer similar self-service BI functionality, consultant Rick Sherman says. But he points to some...