Constraints are the Key

Andy Warren

I’m a Data Guy

I use SQL Server, but even if I changed to a different data platform I’d still be a data guy.

For me, it’s not data science - it’s understanding that GOOD data is a great thing and that BAD data, well, it just chews up my time!

Get It Right The First Time

The odds are really good that once a table is created it’s done. No one is going to go back and fix the names or the lengths or the nullability.

If anything changes, it will probably be adding a column or an index.

If you build it well, you’ll have data as good as the users/sources provide. Constraints don’t guarantee the data is correct, just that it matches our rules!

If you don’t...then have fun explaining why report 1 doesn’t match report 2!

Constraints Don’t Take a Lot of Time

This isn’t a design class. Absolutely sweat the table design and do the best you can with what you now. Build for today, maybe for tomorrow, not for feature 17 that will probably never get built.

Once you create a table, invest 15 minutes in constraining it. Probably less. If you do it then one of two things will happen:

- You’ll catch a bunch of places where you would have had inconsistent data
- You’ll realize that one or more constraints are ‘too tight’ or just wrong

Ways to Constrain - The List

- Columns
- Data Types and Lengths and Nullability
- Primary Key (and any unique keys, ugly thought that is!)
- Defaults
- Check Constraints
- Foreign keys
- Triggers (don’t forget instead of!)
- Indexed Views

Which ones pertain to a given table and which ones go across tables? Which can do both?

Exploring Constraints

An unscripted demo!

Philosophy

Constraints should be reasonable and fairly lightweight.

Addresses are a great example. Could we make sure every change to an address is valid? Should we?

If it’s a lookup value, I want a table and a foreign key. It makes reporting so much easier!

Go column by column. Can order quantity ever be negative? Add the constraint!

Constraints add overhead. It’s hard to know what is too much. It’s not wrong to rely on the app or a proc to guarantee some things, but it’s not my first choice.

It’s Hard to Do it Perfectly

For example, recently I helped a colleague who wanted to constrain a column so that it could only have letters, numbers, or an underscore. How would you do that?

Is that good enough? Can it consist of ONLY an underscore? Can you have multiple underscores in a row?

Don’t let the perfect be the enemy of the good, or the practical.

You Didn’t Get It Right the First Time

Happens to all of us. Lack of time, vision, experience. It’s ok.

So, go back and look.

- Nullable columns. Should they be nullable? Do they really contain any nulls? May find some easy wins (and it helps the optimizer)
- Foreign Keys. My biggest target. I’ve got to be able to count on the relationships.
- Check constraints. Lowest priority for me, but still valuable.

Fixing those on existing data can be hard. Sometimes it’s just a “data fix” and add the constraint. More often it requires fixing the app or bandaiding.

Make it Cultural

We tend to focus on the relational part and get stuck in discussions about naming conventions.

Once the relationships are agreed, walk each table column by column and ask “what rule could we apply here?” and see if it can be done in the database. Remember, not all of them can or should.

Fixing It Is A Grind

Over the past year I’ve added 50+ foreign keys to a system. Every single one required identifying the key as “maybe needed”, researching the data and the calling code, sometimes changing the calling code, sometimes fixing the data, then getting the constraint into a build and having it tested.

The not-data people don’t care/aren’t interested. Product owners aren’t going to be excited about devoting resources to it. Once it’s fixed, no one notices!

But...it’s important work, at the core of what we do. If we inherit a mess, we start to fix it. If it’s new work, it’s on us to set the expectation that we do it as well as we can the first time.

Wrapping Up

It takes time to build your philosophy and understand what is enough and what is too much.

Focus on better, not perfect, one tiny step at a time.

Sometimes it requires creativity that stretches us and those are good days.

Thanks for Attending

https://www.linkedin.com/in/sqlandy