Unnatural Keys

Nature doesn’t come with identifiers.

Matt Schellhas

DALL-E — “A photograph of meat in the shape of a key on a solid background”

At time of writing, I am working in the music industry. And as part of that work, we want a database of all of the songs in the world so that we can properly identify unknown songs and provide attribution so that folks can get paid appropriately. It is a noble goal with some interesting engineering challenges.

There's also some... *less* interesting engineering challenges.

One is a bit self-inflicted. The first instinct for every DB person when faced with the "database of all the songs in the world" problem is to go with a natural key. They think: "there's a bunch of IDs we have to store anyways that the business cares about. That's the definition of a natural key! Let's just use them". After all, there are a lot of songs in the world — slightly more than 100 million, depending on who you ask and what they consider to be a song. Adding our own surrogate means a few hundred megabytes of overhead, excluding indexes on the *other* IDs that the business cares about.

There's even industry standards that *should* take care of this for us. ISRC is literally the ISO standard (ISO 3901) for "uniquely identifying sound recordings". And if you've worked in software for any length of time, you know that it does not.

For example:

- **Not all sound recordings are songs.** Is that recording of rain hitting off of a window a song? No. Does it have an ISRC? Oh yeah. Millions of them.
- **Not all songs have recordings.** People have been making songs for millennia. People have only been recording things for about 150 years. ISRC has existed for about 30 years. There are gaps there. There are race conditions between writing a song and playing a song and recording that song and getting an ISRC allocated for that song. There

are a whole lot of people making unrecorded music every day and a bunch of people recording music that they aren't bothering to register with an ISRC.

- **ISRC only cares about the *recording*.** I talk about the music industry like one homogenous thing, when in reality it is a conglomeration of sub-industries all fighting with one another to extract the most money possible from people listening to music. For songwriters, there is a *separate* ID for a song's composition (ISWC). There is yet another ID for the *release* of the recording (GRid). If it's a product, then it gets a UPC. And then there's all of the various broadcast stuff (radio, TV, internet streaming, social media content, etc.) with their own separate IDs. All of them (and more) are licensed separately, each with their own definition of "song".
- **Most songs have more than one recording.** Generally speaking, each separate recording of a song is supposed to have its own ISRC. For some cases that's pretty clear. A live recording of a song? Sure, new ISRC. A cover? Yup, new ISRC. Remixes? Sure, new ISRC. But what about a *Greatest Hits* album? What about a song used in a movie soundtrack? Since ISRC has a country code and registrant code, does a release in a new country get a new ISRC? What if the recording ownership changes? What if the song is remastered in a way that is inaudible to most humans? What about the karaoke version of the song where the vocals are stripped out? The answer to many of these questions is a solid ***maybe***. And since there are a few hundred organizations making millions of these decisions over the course of decades, there is very little consistency in them.

I'd like to say that this sort of thing is terrible or uncommon, but it is neither. This sort of stuff happens in every industry I've worked in. Hell, I've worked on ACH and electronic health records. They make the music industry look

like paradise by comparison. Across them all, I've found one constant:

There are no real world natural keys — only someone else's surrogate key.

ISRC isn't some intrinsic trait of songs, it's a surrogate key designated by recording companies. SSN, VIN, email address, UPC, tax numbers, country codes, language codes, URIs... they're all just someone else's made up identifier. Fingerprints are unique, but make for poor primary keys and you'll run into problems with whole "people without hands" outlier group. Days kind of work even though they are made up, as long as you assume a time-zone and a calendar system and only care about stuff within a specific few millennia of existence *and* your rows are unique per day.

When you use a natural key, what you're really doing is making a foreign key to another system with absolutely no constraints or consistency guarantees. For something like ISRC it's even worse. Since multiple companies manage them, it's more like a foreign key to an unsynchronized distributed system. What are the odds that the data stays consistent and unique? Not good. Not good at all.

Some times that is an acceptable tradeoff. If you're working with Facebook, then using their user and page IDs makes sense. You can probably trust them to keep those IDs unique, and if your product is tightly coupled to theirs then there's not much more risk in coupling the database as well. If you're working with countries, then ISO country codes are usually good enough. Folks in international waters or on the International Space Station are used to dealing with it, and they're probably a small fraction of your userbase.

Most times, it is not an acceptable tradeoff. The integrity of your data is worth a few bytes per row. Keeping your business from depending on the

competency of some non-profits or some government is worth a few bytes per row. Keeping your business from depending on another company that *probably has a vested interest in your failure* is worth a lot more than a few bytes per row.

Nature doesn't come with identifiers. Humans give labels to things. We're the ones who give things names and codes and symbols. They're all arbitrary, unnatural, human inventions. When doing schema design, the question isn't "should I use a natural key or a surrogate key?". There's only **your** surrogate keys or **someone else's** surrogate keys. The question you should be asking yourself is "do I *really* want to build this table around an unenforced foreign key?".