How to store array or multiple values in one column

28

4

Running Postgres 7.4 (Yeah we are in the midst of upgrading)

I need to store from 1 to 100 selected items into one field in a database. 98% of the time it's just going to be 1 item entered, and 2% of the time (if that) there will be multiple items.

The items are nothing more than a text description, (as of now) nothing more than 30 characters long. They are static values the user selects.

Wanted to know the optimal column data type used to store the desired data. I was thinking BLOB but didn't know if this is a overkill. Maybe JSON?

Also I did think of ENUM but as of now I can't really do this since we are running Postgres 7.4

I also wanted to be able to easily identify the item(s) entered so no mappings or referencing tables.

sql arrays postgresql types

edited Jun 15 '11 at 16:22

Daniel DiPaolo
44.9k 12 104 110

asked Jun 15 '11 at 16:01

Phill Pafford
48.9k 79 243 366

8 Your last requirement is one that should not exist. If you are looking directly into the database for data and wanting to violate normalization rules for the sake of "I can read it more easily", you are doing it wrong. – Daniel DiPaolo Jun 15 '11 at 16:23

2 Answers

You have a couple of questions here, so I'll address them separately:

I need to store a number of selected items in one field in a database

My general rule is: don't. This is something which all but *requires* a second table (or third) with a foreign key. Sure, it may seem easier now, but what if the use case comes along where you need to actually query for those items individually? It also means that you have more options for lazy instantiation and you have a more consistent experience across multiple frameworks/languages. Further, you are less likely to have connection timeout issues (30,000 characters is a lot).

You mentioned that you were thinking about using ENUM. Are these values fixed? Do you know them ahead of time? If so this would be my structure:

Base table (what you have now):

```
| id primary_key sequence
| -- other columns here.
```

Items table:

```
| id primary_key sequence
| descript VARCHAR(30) UNIQUE
```

Map table:

```
| base_id  bigint
| items_id bigint
```

Map table would have foreign keys so base_id maps to Base table, and items_id would map to the items table.

And if you'd like an easy way to retrieve this from a DB, then create a view which does the joins. You can even create insert and update rules so that you're practically only dealing with one table.

What format should I use store the data?

If you have to do something like this, why not just use a character delineated string? It will take less processing power than a CSV, XML, or JSON, and it will be shorter.

What column type should I use store the data?

Personally, I would use `TEXT`. It does not sound like you'd gain much by making this a `BLOB`, and `TEXT`, in my experience, is easier to

answered Jun 15 '11 at 17:03

cwallenpoole
61.5k 18 106 146

OP may find `array_agg` useful in a view to see one line per base row with multiple `description_field` s – Andrew Lazarus Jun 15 '11 at 20:25

1 If I could, what is the benefit of having the `Map table` over saving the `Base table` 's primary key as a column directly on the rows in `Items table` ? – DelightedD0D Jul 16 '16 at 10:17

6

Well, there is an array type in recent Postgres versions (not 100% about PG 7.4). You can even index them, using a GIN or GIST index. The syntaxes are:

```
create table foo (
  bar  int[] default '{}'
);

select * from foo where bar && array[1] -- equivalent to bar && '{1}'::int[]

create index on foo using gin (bar); -- allows to use an index in the above query
```

But as the prior answer suggests, it will be better to normalize properly.

answered Jun 15 '11 at 20:27

Denis de Bernardy
59.2k 8 93 124