Multiple companies in same database

Asked 11 years, 7 months ago Modified 4 years ago Viewed 5k times

4

I'm working on a system which for every "company" has their own "users" and their own "bills". That scenario is better in performance and management? Handle all companies in the same database and link everything to an idempresa, or database for each client?

database architecture multi-tenant

Share

Improve this question

Follow

edited Dec 19, 2020 at 0:47

user4157124

2,904 ● 14 ● 30 ● 44

asked May 22, 2013 at 22:34

Gustavo Salgado

417 ● 5 ● 17

I think a lot of that depends on which RDMS you're using..
– Mike Christensen May 22, 2013 at 22:36

4 Answers

Sorted by: Highest score (default)

10

This is called multi tenancy architecture and each customer is a tenant. There are various strategies to deal with it and each one might bring potential problems.

Having a separate database for each tenant is an option that provides data separation and do not require you to add a column to identify each tenant in your tables and queries, but also has the downside to keep multiple databases up to date.

Having a column in each table of a single database to identify your tenants is also a good strategy, but then it brings problems when scaling and managing different features for different customer for example.

You need to study all available strategies and decides which one is best based on your requirements and pain points.

Share Improve this answer

Follow

answered May 22, 2013 at 22:42

tucaz

6,684 ● 6 ● 40 ● 62

> Hi, if I have users that belong to a Company, every operation that they do in the system, based on there Login can map the tenant, no? Or should I have to put the company id in every table? – Patrick Feb 2, 2017 at 12:37

Putting a tenant data in a separate Database is a straight forward approach and less painful option but then in a

3

long run, when your product gets wildly successful, maintaining this database will become a nightmare.

On the Other hand keeping all the Tenants data in a single database could also make your application non scalable and less performable. The better approach would be the combination of both, the decision of making the choice between these two is completely based on the type, usage and size of the customer.

In certain cases, you may need to provision a separate database for a particular module or feature of your application may be for security or to isolate the specific data alone. I have written an article on these lines; kindly have a look at http://blog.techcello.com/2012/07/database-sharding-scaling-data-in-a-multi-tenant-environment/

Share Improve this answer

Follow

answered May 24, 2013 at 6:37

Ilyas F

533 ● 4 ● 12

Hi, if I have users that belong to a Company, every operation that they do in the system, based on there Login can map the tenant, no? Or should I have to put the company id in every table? – Patrick Feb 2, 2017 at 12:37

1 Adding Company/Tenant ID in every table is the right and straight forward approach. – Ilyas F Mar 1, 2017 at 8:41

Hi, yes I agree with you, I have been developping a SaaS solution and the best way to list an entity for example is to get the logged CompanyId, insert in the query and list everything. Thanks. – Patrick Mar 1, 2017 at 14:23

1

This is an old thread, but it's worth mentioning this for others with this question who may come across this post in the future.

I've had great success on projects in the past by using PostgreSQL and putting the global tables in the "public" schema (like users, groups, etc.) and the same set of tables for each tenant in their own separate schemas.

For example:

For every tenant that's added to the system, a new schema is created with a standard set of tables for the application:

```
CREATE SCHEMA tenant1;
CREATE TABLE tenant1.products (...);
CREATE TABLE tenant1.orders (...);
etc.
```

Each tenant's schema would have its own isolated section within the database with the same set of tables that every other tenant has but filled with their own data.

In the default "public" schema you'd have global "users" and "tenants" tables (along with tables for things like groups and access control lists). Every user belongs only to a single tenant. Upon login, the tenant for that user is looked up and from that point forward any time you connect to the database you set it to use that tenant's schema:

```
SET search_path TO tenant1, public;
```

Once the schema search_path is set, all your SQL queries can be written as if you're working with a single database with tables named "products", "orders", and so forth (along with the tables in the "public" schema). So you can just use something like "SELECT * FROM products" and it would get the products belonging to this user's tenant.

Share Improve this answer

Follow

answered Dec 16, 2020 at 19:52

Dan Delaney

11 ● 1

That's a good solution! I wonder how it would look like if I use object-relational mapping in a backend application such as Express.js or ASP.Net. It would be appreciated if you write a blog post on how to approach this. – Amjad Abujamous May 8, 2023 at 19:11

0

I think the scaling problem of multi-tenant in a single database can be overcome by proper planning up-front. Plan to make it easy to migrate a tenant and their data to another database anything they become big enough to justify it.

If you can automate this migration, based on tenant ID, in each table then it should be easy and safe. I'd just make sure I tested it often as development of new features are going on.

You can mitigate the risks of multi-tenant on one database. You can't really do much when there are multiple databases. You can only be diligent and disciplined to make sure all the databases stay in sync.

Good luck!!!

Share Improve this answer

Follow

answered Oct 17, 2019 at 15:15

Bob

41 ● 2