SaaS database design - Multiple Databases? Split? [closed]

Asked 16 years, 3 months ago Modified 4 years ago Viewed 22k times

21

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Update the question so it can be answered with facts and citations by editing this post.

Closed 3 years ago.

Improve this question

I've seen SaaS applications hosted in many different ways. Is it a good idea to split features and modules across multiple databases? For example, putting things like the User table on one DB and feature/app specific tables on another DB and perhaps other commonly shared tables in another DB?

database-design architecture database-schema multi-tenant saas

Share

edited Feb 6, 2014 at 15:20

Improve this question

Follow

givanse
14.9k ● 8 ● 54 ● 77

asked Sep 16, 2008 at 3:18

Vyrotek
5,439 ● 5 ● 47 ● 72

possible duplicate of What are the advantages of using a single database for EACH client? – givanse Feb 6, 2014 at 8:42

10 Answers

Sorted by: Highest score (default)

35

Start with one database. Split data/functionality when project requires it.

Here is what we can learn from LinkedIn:

- A single database does not work
- Referential integrity will not be possible
- Any data loss is a problem
- Caching is good even when it's modestly effective
- Never underestimate growth trajectory

Source:

LinkedIn architecture

LinkedIn communication architecture

Share Improve this answer

Follow

answered Sep 16, 2008 at 4:22

Sergey Kornilov

1,792 ● 2 ● 13 ● 22

13

[High Scalability](#) is a good blog for scaling SaaS applications. As mentioned, splitting tables across databases as you suggested is generally a bad idea. But a similar concept is sharding, where you keep the same (or similar) schema, but split the data on multiple servers. For example, users 1-5000 are on server1, and users 5000-10000 on server2. Depending on the queries your application uses, it can be an efficient way to scale.

Share Improve this answer

Follow

answered Sep 16, 2008 at 4:15

Tom Ritter

101k ● 31 ● 142 ● 174

9

For SaaS applications, you use multiple databases for multiple tenants, but usually don't split it module-wise.

This is the most common model I have seen in SaaS application design. Your base schema is replicated for each tenant that you add to your application.

Share Improve this answer

Follow

answered Sep 16, 2008 at 4:18

Vaibhav

11.4k ● 11 ● 53 ● 71

most saas apps are 1 db scoped by user_id or account_id – Brian Dillingham Aug 10, 2019 at 0:13

4

Having a single database is best for data integrity because then you can use foreign keys. You can't have this built-in data integrity if you split the data into multiple databases. This isn't an issue if your data isn't related, but if it is related, it would be possible for your one database to contain data that is inconsistent with another database. In this case, you would need to write some code that scans your databases for inconsistent data on a regular basis so you can handle it appropriately.

However, multiple databases may be necessary if you need your site/application to be highly scalable (e.g. internet scale). For example, you could host each database on a different physical server.

Share Improve this answer

Follow

answered Sep 16, 2008 at 3:24

Chris Gillum

15k ● 5 ● 50 ● 66

What about the database size then if we keep all data at one place? I believe we've some sort of limitation.
– Basheer Kharoti Oct 31, 2020 at 15:51

3

Splitting the database by features might not be a good idea unless you see strong evidence suggesting the need. Often you might need to update two databases as part of a single transactions - and distributed transactions are much more harder to work with. Furthermore, if the database needs to be split, you might be able to employ sharding.

Share Improve this answer

Follow

answered Sep 16, 2008 at 3:31

Binil Thomas

13.8k ● 10 ● 58 ● 70

2

Have a look at Azure SQL's Multi-tenant SaaS database tenancy patterns that details a list of solutions and decision criteria.

https://learn.microsoft.com/en-us/azure/azure-sql/database/saas-tenancy-app-design-patterns

This next discussion includes lots of feedback from devs who've been there done that. The general concensus is avoid multiple databases if you can and enforce tenant

only queries automatically. SQL Azure offers row level security to assist in this. It can also be done at the appplication level.

https://www.indiehackers.com/post/should-i-keep-only-one-database-for-each-customer-in-a-saas-product-2af0af42f4

One final thought.. choosing single database at the start, does not exclude you from going database per tenant later on. You can even later support many smaller customers in one DB with larger or premium paying customers having their own DB. However starting with database per tenant means your up for a significant migration cost should you later switch back to multiple tenants per database.

Share Improve this answer Follow

edited Dec 17, 2020 at 22:20

answered Dec 17, 2020 at 21:29

Tony O'Hagan

22.6k ● 3 ● 72 ● 81

1

Ask yourself: What do you gain by moving everything into separate databases?

A lot of pain in terms of management would be my guess. I'd be more keen personally to have everything in a single database and if you hit issues that cannot be solved by a

single database later then migrate the data into multiple databases.

Share Improve this answer

Follow

answered Sep 16, 2008 at 3:20

mjallday

10.1k ● 9 ● 53 ● 73

0

Keep it a natural design (denormalize as much as needed, normalize as less as required). Split the DB Model into its modules and keep the service oriented principles in mind by fronting data with a service (that owns the data).

Share Improve this answer

Follow

answered Sep 16, 2008 at 15:24

Daniel Fisher lennybacon

4,164 ● 1 ● 34 ● 41

0

There are a variety of ways to accomplish it, but the issues of multi-tenancy go deeper than just the data model. I hate to be plugging product, but check out SaaSGrid by my the company I work at, Apprenda.We're a cloud operating system that allows you to write single tenant SOA apps (feel free to use NHibernate for data access) that automatically injects multi-tenancy into your app. When you publish your app, you can do things like choose a data model (isolated database or shared) and SaaSGrid will deploy accordingly and your app will run without any code changes - just write code as if it were for a single tenant!

Share Improve this answer

Follow

answered Dec 4, 2008 at 21:50

Jesse Kliza

0

Why to use database at all ?

I think it's good idea to use distributed storage systems like Hadoop, Voldemort (project-voldemort.com developed and used by LinkedIn).

I think db good for sensetive data like money operations , but for everything else you can use distributed storages.

Share Improve this answer

Follow

answered Apr 23, 2009 at 16:41

Denis Kurilenko