عنوان: **معماری پایگاه داده چند مستاجری (Multi-Tenant Data Architecture)**
نویسنده: محمد پهلوان
تاریخ: ۱۳۹۲/۰۹/۱۱ ۹:۴۰
آدرس: www.dotnettips.info
برچسب‌ها: database design, Database, Cloud computing, SaaS

اعتماد و یا فقدان آن، عامل شماره یک مسدود کردن استفاده از نرم افزار به عنوان خدمات است. معماری پایگاه داده چند مستاجری برای رسیدگی به مشکل نرم افزار به عنوان سرویس (SaaS) که می‌تواند خدمات به تعدادی کلاینت ارائه کند استفاده می‌شود . معماری دیتابیس چند مستاجری وقتی مفید است که یک نمونه از دیتابیس به تعدادی کلاینت خدمات دهد. وقتی که نرم افزارهای محلی نصب می‌کنید نرم افزارهای به عنوان یک سرویس با مشتریان متمرکز، دسترسی به داده‌ها مبتنی بر شبکه با سربار کمتر را فراهم می‌کنند. اما به منظور برخورداری بیشتر از مزیت‌های یک نرم افزار سرویس، یک سازمان باید از سطحی از کنترل روی داده صرفنظر کند و به فروشنده نرم افزار جهت نگهداری و امنیت به دور از چشم آنها اعتماد کند.

برای به درست آوردن این اعتماد، یکی از بالاترین الویت ها، آینده نگری معماری نرم افزار و ساخت یک معماری داده است که باید هر دو قوی و به اندازه کافی ایمن باشد، این دو برای راضی کردن مستاجران و کلاینت هایی که علاقمند هستند کنترل داده‌های حیاتی تجارت خود را به شخص سومی واگذار نمایند، موثر است در حالی که برای اداره کردن و نگهداری مقرون به صرفه است.

سه روش مدیریت چند مستاجری داده

دیتابیس‌های جداگانه برای هر مستاجر

دیتابیس مشترک و schema جداگانه برای هر مستاجر

دیتابیس مشترک و schema مشترک

انتخاب روش مناسب برای برنامه شما به عوامل زیر بستگی دارد :
سایز دیتابیس هر مستاجر

تعداد مستاجران

تعداد کاربران هر مستاجر

نرخ رشد مستاجر

نرخ رشد دیتابیس مستاجر

امنیت

هزینه

1) دیتابیس‌های جداگانه برای هر مستاجر :
ذخیره سازی داده‌های مستاجران در دیتابیس‌های جداگانه ساده‌ترین روش است. در این روش هر مستاجر یک دیتابیس دارد. منابع و کدهای برنامه معمولا در سرور بین همه مستاجران مشترک است اما هر مستاجر مجموعه ای از داده دارد که بطور منطقی از سایر مستاجران جدا شده است.

مزایا :
امنیت بیشتر

سهولت سفارشی سازی برای هر مستاجر

سهولت نگهداری (Backup و Restore) برای هر مستاجر

معایب:
برای نگهداری سخت افزار قوی مورد نیاز است

این روش هزینه بیشتری برای تجهیزات (Backup و Restore) برای هر مستاجر دارد

2) دیتابیس مشترک و schema جداگانه برای هر مستاجر :
خدمات دهی به چندین مستاجر در یک دیتابیس مشترک اما هر مستاجر یک مجموعه از جداول گروهبندی شده دارد که با Schema جدا شده است که برای هر مستاجر الزامی است.

مزایا :
برای دیتابیس برنامه‌های کوچک مناسب است. وقتی تعداد جداول برای هر مشتری کم است

هزینه کمتری نسبت به روش اول دارد

برای مشتریانی که نگران امنیت هستند، سطح منطقی مناسبی برای جداسازی داده ه وجود دارد

معایب:
اطلاعات مستاجران در صورت بروز خطا به سختی restore می‌شود

مدیریت آن برای دیتابیس‌های بزرگ مشکل است

3) دیتابیس مشترک و schema مشترک :

این روش شمامل یک دیتابیس و یک مجموعه از جداول برای چندین مستاجر است. داده‌های جدول می‌تواند شامل رکوردهای هر مستاجر باشد

مزایا :
در مقایسه با روش قبلی، کمترین هزینه سخت افزاری را دارد

می‌تواند مستاجران بیشتری رادر هر سرور پشتیبانی کند

قابلیت بروز رسانی آسان در یک جا برای همه مستاجران

مدیریت آسان دیتابیس و خطا و Backup و Restore

معایب:
امنیت بیشتری مورد نیاز است تا مطمئن شوید هیچکس به اطلاعات سایر مستاجران دسترسی ندارد.

می‌تواند روی کارایی کوئری‌ها تاثیر بگذارد چون تعداد رکوردها زیاد است.

بروزرسانی و سفارشی کردن فقط برای یک مستاجر سخت است

منابع :

http://msdn.microsoft.com/en-us/library/aa479086.aspx

http://www.codeproject.com/Articles/51334/Multi-Tenants-Database-Architecture

نظرات خوانندگان

نویسنده: XPlan
تاریخ: ۱۳۹۲/۰۹/۱۲ ۱۵:۲

با تشکر از شما
در صورت امکان برای پیاده سازی با روش 2 و 3 یک مثال ساده بیان کنید.

نویسنده: محمد پهلوان
تاریخ: ۱۳۹۲/۰۹/۱۲ ۱۷:۵۷

در لینک MSDN که در قسمت منابع آمده روش 2 و 3 به صورت اجمالی و دستورات SQL نحوه کار با schema ذکر شده است.