عنوان: **NoSQL ؟**
نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۸:۲۳
آدرس: www.dotnettips.info
برچسب‌ها: JSON, NoSQL, RavenDB

به شما خواننده گرامی پیشنهاد می‌کنم مطلب قبلی " آشنایی با JSON؛ ساده - خوانا - کم حجم " که پیش درآمدی بر این موضوع است را مطالعه کنید.

NoSQL یک مفهوم عام است و تعریف ساده آن "پایگاه داده بدون SQL است". به این معنی که در آن خبری از جدول ها، روابط بین آن‌ها و ... نیست!
اما چرا باید با وجود اینکه SQL به اغلب نیازهای ما پاسخ داده است، باید سراغ تکنولوژی‌های دیگر رفت؟

وقتی نگاهی به لیست شرکت‌های بزرگی می‌اندازیم که جز مشتریان پر و پا قرص NoSQL هستند(+ و +)، تعجب می‌کنیم! آیا آن‌ها از قدرت و قابلیت‌های SQL بی خبراند؟

پاسخ این گونه از سوال‌ها به تحلیل سیستم مربوط می‌شود. به عهده تحلیل گر است تا با توجه به اجزاء سیستم و ارتباط آن‌ها بهترین روش را برای ذخیره سازی اطلاعات انتخاب کند.
NoSQL بر اساس نحوه پیاده سازی اش دسته بندی شده است؛ که مهم‌ترین آن‌ها در زیر آمده است :
Wide Column Store

Document Store

Key Value / Tuple Store

Graph Databases

Multimodel Databases

Object Databases

برای آشنایی بهتر با هر کدام به nosql-database.org مراجعه کنید.

انتخاب روش؛ یک مثال ساده :
فرض کنید روال استخدام نیروی کار جدید در یک سازمان، از قرار زیر باشد:

ثبت مشخصات فردی

ارائه مدارک تحصیلی

شرکت در آزمون استخدامی
شرکت در مصاحبه (درصورت قبول شدن در آزمون)
شرکت در دوره آموزشی (در صورت قبول شدن در مصاحبه)

روش‌های ممکن برای نگهداری اطلاعات :
روش اول، تهیه پوشه هایی برای نگهداری اطلاعات مربوط به هر مرحله به صورت مجزا است.

روش دوم، تهیه یک پرونده برای هر شخص و نگهداری اسناد مربوط به شخص (در هر مرحله) است.

انتخاب روش اول امکان پذیر است، اما باعث پیچیده‌تر شدن سیستم و اتلاف زمان می‌شود که مطلوب نیست. برای پیاده سازی روش دوم، SQL پاسخ گوی نیاز پروژه نیست و با توجه به نیاز پروژه بهترین روش نگهداری اطلاعات، Document Store (نگهداری اطلاعات بر اساس ساختار اسناد) است.
خوش بختانه تعداد پایگاه‌های داده ای که بر اساس تکنولوژی Document Store پیاده سازی شده اند، زیاد است و از قدرتمندترین آن‌ها می‌توان به CouchDB ، MongoDB و RavenDB اشاره کرد. هرکدام از این انتخاب‌ها مزایا و معایبی دارند که باید با توجه به نیاز خود، مقایسه ای انجام داده و بهترین را انتخاب کنید.
انتخاب من RavenDB بوده است و دلایل آن :
بر اساس زبان سی شارپ نوشته شده است و همچنین با LINQ خیلی خوب کار می‌کند.

Transaction را پشتیبانی می‌کند.

اساس ذخیره سازی آن JSON است.

محیط Management Studio کاربر پسندی دارد.

نقطه آغازین بحث بعد RavenDB خواهد بود که *Bryan Wheeler* (مدیر توسعه بسترهای نرم افزاری در msn) در باره آن گفته :

RavenDB just rocked my world. It’s extremely approachable, even for non-database guys – it took me less than 30 minutes to get up and running

خوشحال می‌شوم، نظرات و تجربیات شما را در رابطه با NoSQL بدانم.

نظرات خوانندگان

نویسنده: RaminMjj
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۸:۵۴

سلام.
ابتدا تشکر میکنم از مطلبی که دارید ارائه میدهید.
شیوه نگارشتون هم بسیار خوبه.
فقط یک پیشنهاد دارم. اونهم اینه که یک مطلب اختصاص بدهید به؛ در چه پروژه‌هایی باید از NoSQL استفاده کرد و چه پروژه‌هایی نباید.

نویسنده: mze666
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۰

سلام - خیلی ممنون بابت مطلب خوبتون. فقط اگر براتون ممکنه یه آموزش گام به گام یا یه نمونه پروژه از RavenDB که به نظرم بهترین هستش رو بذارید. ممنون

نویسنده: مجتبی چنانی
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۲۰

با سلام

من به عنوان کسی که در پروژه‌های خود از انوع ذخیره سازی‌ها بر اساس نیاز استفاده کردم(سرعت! راحتی! پلتفرم ها! و...) هم نظر میدم و هم پاسخ شما دوست عزیز را می‌دم.

قطعا انتخاب اینکه از چه روشی برای ذخیره سازی داده‌ها استفاده شود بسته به تیم پیاده سازکننده پروژه و نیز طراحان و... دارد. من با یک مثال توضیحی را خدمت شما می‌دهم.

در یک پروژه که اخیرا در حال اجرا هست(در دست من و هم تیمی‌های من) این پروژه یک پروژه بزرگ و با دیدها و اهداف وسیعی هست. ما در این برنامه هم از ادرس دهی بر اساس پوشه‌ها و دایرکتوری‌ها داده‌ها را ذخیره کردیم(اطلاعاتی مانند لینک فایل‌ها و یا تصاویر و...) و حتی در بعضی محل‌ها نیاز بود که اطلاعات یک فرد را در یک فایل xml قرار می‌دادیم و بعضی وقتها هم در پایگاه داده و هم فایل xml به این دلیل که در مورد اول تنها برنامه سمت کلاینت نیاز به این اطلاعات داشت و در آنجا پارسر قوی xml وجود داشت اما در مورد دوم ما به یک سری دیتا نیاز داشتیم که هم در سرور به آنها نیاز داریم و هم کلاینت! خب در بحث وب ما به مدیران اگر می‌خواستیم xml ارائه کنیم قطعا راه حل خوبی نبود و از سرعت و کارایی ما کم می‌کرد لذا از پایگاه داده استفاده کردم ولی برای زمانی که کاربر کلاینتی ما نیاز به اطلاعات داشت به این دلیل که بار سرور زیاد نشود از xmlها استفاده می‌شد که با یک لینک مستقیم می‌توانست به دست اورد(البته خود لینک همین فایل xml هم ساخته می‌شد! هیچ جا ذخیره نمی‌شد!)

عذر می‌خوام اگر بجای نویسنده پاسخ دادم البته این پاسخ من خیلی سربسته بود و انشا.. مفید بوده.

از نویسنده مطلب بابت مطلب خوبشون که کم دیدم در تارنماهای فارسی به اون بپردازن(متاسفانه بسیاری از اساتید دانشگاهی با این مفهوم حتی اشنایی ندارند با اینکه دانستن کلیت ان یک تعریف ساده است!) موفق باشید.

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۲۷

از شما ممنونم به خاطر پیشنهاد خوبتون.

به نظر خودم موضوعی که شما مطرح کردید جای بحث بیشتری دارد و حتما این مورد رو برای نوشته‌های آینده مد نظر قرار خواهم داد.

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۳۰

ممنون از شما که این مثال رو مطرح کردید.
در نوشته‌های من جای یک مثال برای واضح شدن بیشتر موضوع خالی بود!

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۳۶

موافقم، هیچ چیز مثل یک مثال کاربردی یادگرفتنی نیست...

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۴۰

یه مشکلی داره RavenDb ! ... اونم اینه که مجوزش از هموناس که اگه پروژه تجاری باشه باس پولشو بدی! (اگه متن باز که باشه هیچ!)

نویسنده: mze666
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۴۷

بله درسته ولی اگه بخوایم حساب کنیم ما از خیلی چیزا توی برنامه‌های تجاریمون استفاده میکنیم (Telerik, Stimulsoft, ...) ولی پولشو نمیدیم. اینم روش. (البته نمیگم کار خوبی مکینیم!)

نویسنده: RaminMjj
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۲:۱۱

با تشکر از جوابی که دادید.
ولی من میخواستم مطلبی مشابه این مقاله ارائه بشه تا بیشتر با NoSQL آشنا بشیم
http://www.dbta.com/Articles/Editorial/Trends-and-Applications/SQL-or-NoSQL-How-to-Choose-the-Right-Database-for-Your-Application-71240.aspx

نویسنده: peyman
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۲:۲۶

فکر میکنم Neo4j هم عالی باشه ! گر چه مایکروسافت هم داره روی Trinity کار میکنه که هر دو از نوع گراف دیتابیس‌ها هستن و به نظر من کار با گراف دیتابیس‌ها خیلی زیباتر و لذتبخش‌تر هست

نویسنده: محمد
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۲:۵۷

اتلاف زمان صحیح است و نه «اطلاف زمان». اتلاف از تلف کردن میاد. ممنون به هر حال.

نویسنده: ghafoori
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۳:۱۵

اگر برنامه داخل ایران باشه اینکارو می‌کنیم اما اگر بخواهیم اون را داخل سرور امریکا یا هر دیتاسنتری که به مجوزها گیر میده برنامه را داشته باشیم باید چکار کنیم اینجا مانگو خودش را بهتر نشون میده

نویسنده: نیما
تاریخ: ۱۳۹۱/۰۴/۱۲ ۱:۴

سلام دوست عزیز

ممنون از مطلبتون. در نگاه اول مطلب شما اینجوری به من القا کرد که اطلاعات مثلا سریالایز بشن حالا چه بصورت json و یا xml . من رو پروژه ای کار کردم که اطلاعات بصورت xml بود و فرض کنید حجم این فایلهای xml به حدود 10 کیلو بطور میانگین میرسید . گزارشگیری از این xmlها بسیار وقت گیر بود مخصوصا که اگر قرار بود group by یا اعمال دیگری رو انجام بدیم و خیلی اوقات به timeout میخورد که با عوض کردن این شیوه و قرار دادن اطلاعات در جداول مختلف مشکلات بکلی حل شد. ممنون میشم بیشتر توضیح بدین. موفق باشید

نویسنده: رضا.ب
تاریخ: ۱۳۹۱/۰۴/۱۲ ۲:۱۴

دوست عزیز، لطفا بیشتر توضیح دهید. من چند بار کامنت‌تون رو خوندم متوجه نشدم چی میگین. ممنون.

نویسنده: رضا.ب
تاریخ: ۱۳۹۱/۰۴/۱۲ ۳:۱۱

دو سوال داشتم:
- امکان انتقال (Migrate) بین یه دیتابیس relational و nosql در عمل ممکن هست؟ (منظورم تبدیل رابطه‌ای‌ها به nosqlهاست. چون برعکسش محاله ظاهرا؟!)
- نقش ORMها در برقراری ارتباط Objectی و منطق برنامه‌های شی‌گراء با این نوع دیتابیس‌های براساس سند(بدون ساختار) کجاست؟ اصلا ORM معنی میده هنگام کار با NoSQL؟
با تشکر. ممنونم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۲ ۸:۴۰

- در ravendb امکان replication به sql server وجود دارد.
- یکی از اهداف مهم ORMها در دات نت، نوشتن کوئری‌های strongly typed است. در ravendb شما از روز اول با کوئری‌های strongly typed سروکار دارید. همچنین از همان ابتدای کار هم با کلاس‌های دات نتی و نگاشت خودکار آن‌ها کار می‌کنید. کلا ravendb برمبنای معماری و همچنین توانمندی و پیشرفت‌های زبان‌های دات نتی تهیه شده.

نویسنده: mze666
تاریخ: ۱۳۹۱/۰۴/۱۲ ۹:۴۴

سلام آقای نصیری - میخواستم از شما یا آقای ترک زاده خواهش کنم که یه مورد از اینها (RavenDB, CouchDB, ...) که به نظر خودتون خوبه رو آموزش بدید.
یه آموزش اساسی مثل MVC یا Code First که تو چند جلسه تمام مباحثش رو گفتید.
ممنون.

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۲ ۱۰:۳۹

ما توی مکتب این جوری گفته بودن بهمون...
ممنون که تزکر دادین. اسلاح شد :-)

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۲ ۱۰:۵۶

ممنون از جناب آقای نصیری که پاسخشان در رابطه با ORM کامل و کافی بود.
اما در مورد سوال اول شما :
در بعضی موارد تبدیل پایگاه داده Table-Relational به بعضی موارد مثل Document Store کاملا امکان پذیر است؛ اما تبدیل آن به نوع KeyValue اساسا معنی ندارد، زیرا کاربرد این دو روش کاملا متفاوت است.
اما این نکته قابل توجه است که اگر تحلیل سیستم شما بر اساس Table-Relational انجام گرفته باشد؛ بعد از تبدیل به Document-Store، با کاهش سرعت مواجه می‌شوید.
و به نظر من زمانی باید سراغ روش‌های NoSQL رفت که ساختار Table-Relational پاسخ مناسبی برای نیاز ما نباشد.

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۲ ۱۱:۱۳

سلام
نظر شما تا حدودی صحیح است اما کلاس‌های دات نت مثل XMLWriter , XDocument و ... قابل مقایسه با Engine قدرتمندی که برای یک پایگاه داده نوشته می‌شود، نیستند.
همچنین یکی از نیازها که باعث می‌شود سراغ NoSQL برویم، حجم عظیم اطلاعات است.
پس هیچ نگرانی در مورد حجم اطلاعات نباید وجود داشته باشد...

نویسنده: رضا.ب
تاریخ: ۱۳۹۱/۰۴/۱۲ ۱۵:۳۷

خب یعنی برای رفتن سمت هر NoSQLی باید دلیل مرجحی داشته باشیم. و بخاطر جدید بودن و استفاده سازمان‌های عظیم از آنها و یا حتی آسان‌تر بودن، دلیل نمیشود که پایگاه‌داده‌ای رابطه‌ای رو رها کنیم.
و این زمانی اتفاق میوفته که این 6 نوعی که ذکر کردید، رو کاملا بشناسیم. مزایا معایب و موارد کاربرد اون‌رو بدونیم و با اثبات ردِ کارایی مطلوب دیتابیس‌های رابطه‌ای به انتخاب NoSQLی دست بگذاریم.
در مورد کامنتتون متوجه نشدم علت اینکه یه پایگاه داده‌ی رابطه‌ای چرا نمیتونه به جفت مقدار/کلید تبدیل بشه؟ شاید بلعکس‌اش محال باشه. مثلا وراثت یا جدول‌های با ستون‌های پویا و ... اصلا در پایگاه‌های رابطه‌ای بی‌معنی هستند.

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۵:۵

شاید من نتونستم منظور خودم رو واضح بگم؛
Table-Relational و NoSQL نقطه مقابل هم نیستند و انتخاب شما بین یکی از روش‌های ذخیره کردن اطلاعات (Graph Databases ، Object Databases ، Table Relational و ...) مشابه مثال انتخاب یکی از Type هایی مثل bit ، TimeSpam ، long و ... برای ذخیره کردن یک مقدار کوچک است. درست است که همه این کارها را با string می‌توان انجام داد و لی می‌توان با انتخاب درست در سرعت و فضایی که قرار است مصرف شود، صرفه جویی کرد.

و در باره مورد بعد که مطرح کردید، شاید یک مثال ساده قضیه رو روشن‌تر کند؛ می‌شود یک عدد کوچک رو در متغییری از جنس TimeSpam ریخت، اما اگر این عدد به معنی زمان نباشد، روش ما بهینه و حتی درست نیست، اما کار انجام شده است...
در صورتی که می‌شود این مقدار را در یک متغییر از جنس int ذخیره کرد.

امیدوارم شبهه ای که برای شما ایجاد شده است، با ارائه یک مثال کاربردی از RavenDB که در پست بعدی خواهم گفت، برطرف شود...

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۴ ۱۲:۴۰

تا آماده شدن مثال کاربردی; دیدن این پست خالی از لطف نیست.

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۰/۲۳ ۱۲:۱۰

سلام

از RavenDB راضی بودین؟ آیا واقعا از جستجوی Full-Text بهره مند است و تونسته Lucene رو خوب تعبیه کنه؟

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۱۰/۲۵ ۱۳:۳۰

سلام
تا اندازه ای که کارکردم خوب بود، البته پیش نیومد که توی پروژه Enterprise از آن استفاده کنم و در مورد Full-Text , Lucene راستش تا حالا امتحان نکردم...
شاید دوستان دیگر بتوانند راهنمایی کنند.

نویسنده: بازرگان
تاریخ: ۱۳۹۱/۱۱/۲۰ ۱۴:۴۴

گوگل پلاس و فیسبوک برای بانک اطلاعاتشون از چه شیوه هایی استفاده میکنند ؟

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۲۰ ۱۷:۲۱

گوگل از بانک اطلاعاتی ساخت خودش استفاده می‌کنه: اطلاعات بیشتر ، فیس بوک هم در اینجا