عنوان: **طراحی پایگاه داده چند زبانه**
نویسنده: محمد پهلوان
تاریخ: ۱۳۹۲/۰۸/۰۹ ۱۲:۰
آدرس: www.dotnettips.info
گروه‌ها: MySQL, database design, SQL

در این مقاله قصد داریم اطلاعات مفیدی را در مورد طراحی دیتابیس‌های چند زبانه، در اختیار شما بگذاریم. مدتی قبل به طراحی دیتابیسی که چند زبانه بودن توضیحات کالا را برای مشتریانی از کشورهای مختلف پشتیبانی می‌کرد، نیاز داشتم . وقتی شروع به پیاده سازی طرح دیتابیس کردم، جواب سرراست نبود. زمانیکه در وب برای بهترین راه جستجو می‌کردم، با نظرات و روش‌های زیادی مواجه شدم. در ادامه بعضی از روش‌های محبوب را بیان می‌کنم.

ستون اضافی : این ساده‌ترین راه است و به ازای هر ستونی که نیاز به ترجمه داشته باشد، ستون اضافی در نظر می‌گیریم.

```
CREATE TABLE app_product (
     Id Int IDENTITY NOT NULL,
     Description_en Text,
     Description_pl Text,
     PRIMARY KEY (Id)
);
```

مزایا :

سادگی

کوئری‌های آسان (بدون نیاز به join)

معایب :

اضافه کردن زبان جدید نیاز به تغییر جداولی که چند زبانه هستند دارد

اگر وارد کردن داده برای همه زبان‌ها الزامی نباشد (بعضی جاها فقط زبان پیش فرض الزامی است) ممکن است داده‌های زیاد و یا فیلدهای خالی در دیتابیس ایجاد شود

نگهداری آن مشکل است

یک جدول ترجمه : این روش تمیزترین راه از دیدگاه ساختار دیتابیس به نظر می‌رسد. شما همه متن هایی را که نیاز به ترجمه دارد، در یک جدول ذخیره می‌کنید.

```
CREATE TABLE ref_language (
         Code Char(2)NOT NULL,
         Name Varchar(20) NOT NULL,
         PRIMARY KEY (Code)
);

CREATE TABLE app_translation (
        Id Int IDENTITY NOT NULL,
        PRIMARY KEY (Id)
);

CREATE TABLE app_translation_entry (
       TranslationId Int NOT NULL,
       LanguageCode Char(2) NOT NULL,
       Text Text NOT NULL,
       FOREIGN KEY (TranslationId) REFERENCES app_translation(Id),
       FOREIGN KEY (LanguageCode) REFERENCES ref_language(Code)
);

CREATE TABLE app_product (
      Id Int IDENTITY NOT NULL,
      Description Int NOT NULL,
      PRIMARY KEY (Id),
      FOREIGN KEY (Description) REFERENCES app_translation(Id)
);
```

مزایا :
اضافه کردن زبان جدید به تغییر طرح دیتابیس نیاز ندارد

به نظر تمیز است و رویکرد رابطه‌ای دارد

همه ترجمه‌ها در یک مکان است (بعضی‌ها می‌گویند این جز معایب است چون امکان خوانایی و نگهداری کمتر است)

معایب :
کوئری‌های پیچیده (به join های چندگانه نیاز دارد تا شرح کالا را به درستی نمایش دهد)

پیچیدگی زیاد
جدول ترجمه اضافی به ازای هر جدول چند زبانه : برای هر جدولی که نیاز به ترجمه دارد یک جدول اضافی ساخته می‌شود. جدول اصلی داده‌های غیر مرتبط به زبان و جدول دوم همه اطلاعات ترجمه شده را ذخیره می‌کند.

```
CREATE TABLE ref_language (
      Code Char(2)NOT NULL,
      Name Varchar(20) NOT NULL,
      PRIMARY KEY (Code)
);

CREATE TABLE app_product (
      Id Int IDENTITY NOT NULL,
      PRIMARY KEY (Id)
);

CREATE TABLE app_product_translation (
      ProductId Int NOT NULL,
      LanguageCode Char(2) NOT NULL,
      Description Text NOT NULL,
      FOREIGN KEY (ProductId) REFERENCES app_product(Id),
      FOREIGN KEY (LanguageCode) REFERENCES ref_language(Code)
);
```

مزایا :
اضافه کردن زبان جدید به تغییر طرح دیتابیس نیاز ندارد

کوئری‌های نسبتا ساده است

معایب :
ممکن است تعداد جداول دو برابر شود

سه مثال نشان داده شده در بالا به ما ایده می‌دهند که چگونه روش‌های مختلف ممکن است استفاده شوند. البته اینها همه گزینه‌های ممکن نیستند، فقط محبوبترین روش‌ها هستند و شما می‌توانید آنها را ویرایش کنید؛ به عنوان مثال با تعریف View های اضافی که join های پیچیده شما را در کدها، ذخیره می‌کنند. راه حلی که شما انتخاب می‌کنید به نیازمندی‌های پروژه وابسته است. اگر شما به سادگی نیاز دارید و مطمئن هستید تعداد زبان‌های پیشتیبانی کم و ثابت است، می‌توانید راه حل اول را انتخاب کنید. اگر به انعطاف پذیری بیشتری نیاز دارید، می‌توانید join های ساده در کوئری‌های چند زبانه را انتخاب کنید. راه حل سوم انتخاب مناسبی است.

منبع :

http://fczaja.blogspot.com/2010/08/multilanguage-database-design.html

نظرات خوانندگان

نویسنده: ناصر فرجی
تاریخ: ۱۳۹۲/۰۸/۰۹ ۱۶:۱۴

روشی که من خودم مدتی استفاده میکردم اضافه کردن یک فیلد language به هر تیبل بود. موقع ثبت دیتا هر زبانی بود همون زبان رو تو این فیلد نگه میداشتم. مثلا برای فارسی fa و انگلیسی en و ... , موقع نمایش هم بر اساس زبان سایت یک کوئری ساده گرفته می‌شد و اطلاعات زبان مورد نظر لود می‌شد.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۲/۰۸/۰۹ ۱۹:۲۹

با سلام و تشکر از مطلب خوبتون
طراحی با یک جدول زبان و نگه داشتن کلید خارجی در جداول مربوطه بهتر میشه
چندید ساله که از این طراحی استفاده میکنیم و جواب داده.
یکی از مزایایی که داره میتونی مدیریت سامانه را نسبت به هر زبان بطور مستقل انجام بدی
و هرجا که نیاز داشتی همزمان چند رکورد را درج کنید.
ساده و روان.
البته استراتژی سیستم استفاده از الگوی مناسب رو توجیه میکنه.

نویسنده: محمد پهلوان
تاریخ: ۱۳۹۲/۰۸/۱۰ ۱۰:۵۰

استفاده از یک فیلد language در هر تیبل باعث می‌شود شما برای یک موجودیت کالا مثلا در سه زبان مختلف 3 رکورد درج کنید که در ارتباط این کالا با دیگر جداول دچار مشکل خواهید شد

نویسنده: محمد پهلوان
تاریخ: ۱۳۹۲/۰۸/۱۰ ۱۱:۳۶

روش دوم واقعا روش تمیز و جمع و جوریه اما کوئری هاش واقعا پیچیده اس و انعطاف نداره. به نظر من مخصوصا برای استفاده از EF روش سوم روش مناسب‌تری باشه. این کوئری‌ها رو با توجه به حجم داده زیاد و سایت پرترافیک بررسی کنید.
خودم بین روش دو و سه مرددم. از دوستان میخوام نظراتشون را با دلائل بیان کنن تا به نتیجه خوبی برسیم

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۱۰ ۱۱:۵۱

مطلب Globalization در ASP.NET MVC - قسمت ششم در همین راستا مفید است.

نویسنده: بختیاری
تاریخ: ۱۳۹۲/۰۸/۱۰ ۱۵:۱۶

سلام
من روش آقای موسوی را منطقی می‌دانم خودم هم با این روش یک سایت طراحی کردم که الان درست و بدون مشکل کار می‌کند