عنوان: **نرمال سازی (قسمت اول: First Normal Form)**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۳ ۱۴:۵۰
آدرس: www.dotnettips.info
گروه‌ها: database design, logical design, normalization, first normal form

مقدمه

نرمالسازی یا normalization باعث جلوگیری از تکرار و افزونگی اطلاعات می‌شود. و همچنین مانع از یکسری ناهنجاری‌ها در عملیات درج، بروز رسانی، حذف و انتخاب خواهد شد.
شکل‌های نرمال متعددی تعریف شده اند که به شرح زیر است:
شکل نرمال اول (1NF)

شکل نرمال دوم (2ND)

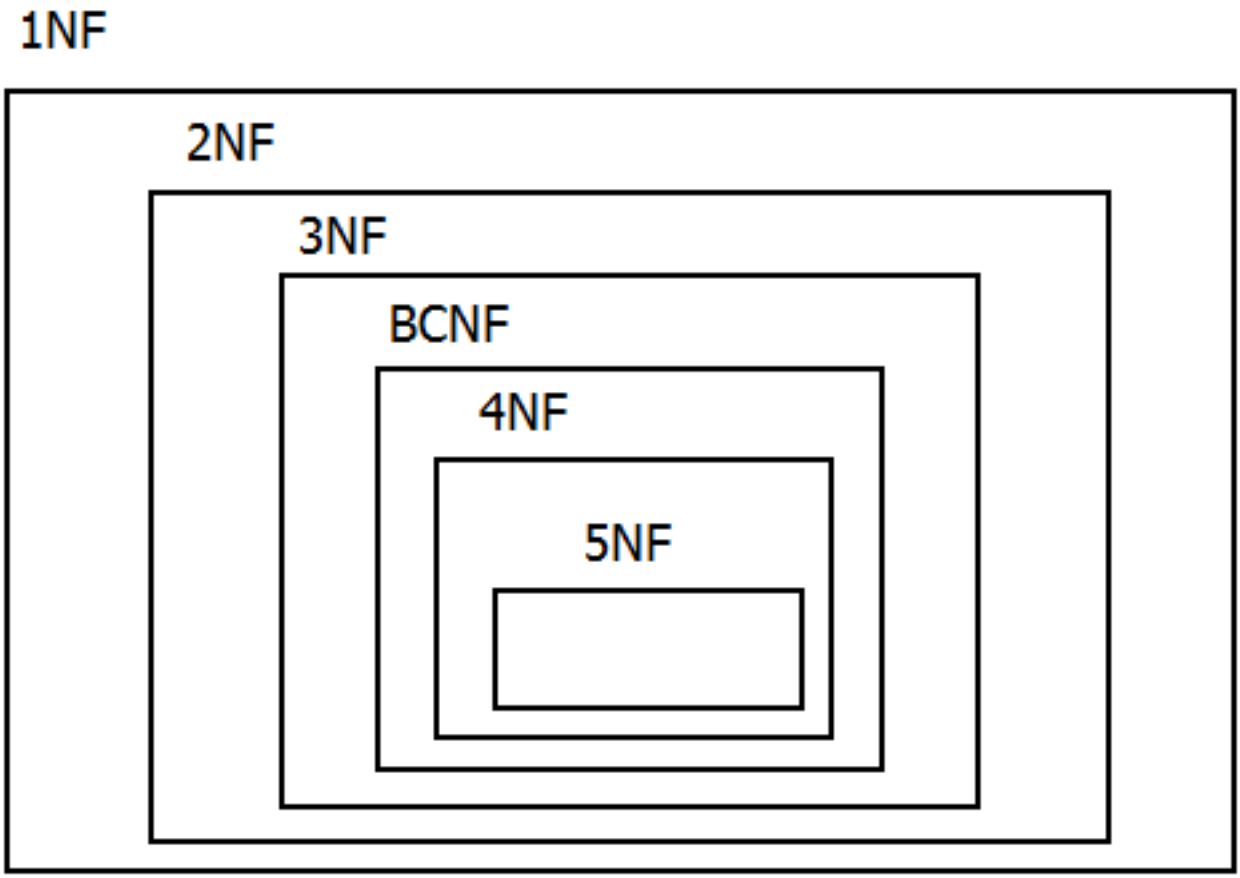
شکل نرمال سوم (3NF)

شکل نرمال بویس کاد (BCNF)

شکل نرمال چهارم (4NF)

شکل نرمال پنجم (5NF)

سه شکل اول نرمال یعنی 1NF ، 2NF و 3NF توسط دکتر Codd تعریف شده اند. شکل نرمال بویس کاد نیز که یک تعریف اصلاح شده و قوی‌تر از 3NF به Boyce و Codd منسوب است . بعد از آن Fagin شکل چهارم نرمال(4NF) را تعریف کرد (چرا که در آن زمان BCNF شکل سوم نرمال خوانده می‌شد).



تصویر فوق می‌گوید اگر جدولی در شکل سوم نرمال باشد حتما دارای شکل دوم نرمال و شکل اول نرمال هم خواهد بود.

شکل اول نرمال (First Normal Form)

تعریف رسمی:

یک متغیر رابطه ای به شکل اول نرمال است اگر و فقط اگر در هر مقدار مجاز آن متغیر رابطه ای، هر چندتایی فقط یک مقدار برای هر خصیصه داشته باشد.

منظور از اصطلاحات متغیر رابطه ای، چندتایی و خصیصه به طور غیر رسمی به ترتیب برابر است با جدول، سطر و ستون.

قسمت کلیدی تعریف، این جمله است: "فقط یک مقدار برای هر خصیصه داشته باشد"

به دو جدول زیر توجه کنید، این جداول به شکل اول نرمال نمی‌باشد چرا که به ازای هر مشتری برای خصیصه شماره تلفن چند مقدار خواهیم داشت:

Primary Key

کد مشتری	نام مشتری	شماره تلفن	شماره تلفن	شماره تلفن
کد۱	مشتری۱	مقدار۱	مقدار۲	
کد۲	مشتری۲			
کد۳	مشتری۳	مقدار۱		
کد۴	مشتری۴	مقدار۱	مقدار۲	مقدار۳

Primary Key

کد مشتری	نام مشتری	شماره تلفن ها
کد۱	مشتری۱	مقدار۱ - مقدار۲
کد۲	مشتری۲	
کد۳	مشتری۳	مقدار۱
کد۴	مشتری۴	مقدار۱ -مقدار۲-مقدار۳

در جدول اول ستون شماره تلفن چند بار تکرار شده است. یعنی برای یک مشتری چند مقدار برای خصیصه شماره تلفن خواهیم داشت که این مغایر با تعریف شکل اول نرمال است. همین اتفاق نیز در جدول دوم افتاده است با این فرق که مقادیر خصیصه شماره تلفن در یک ستون درج شده اند.

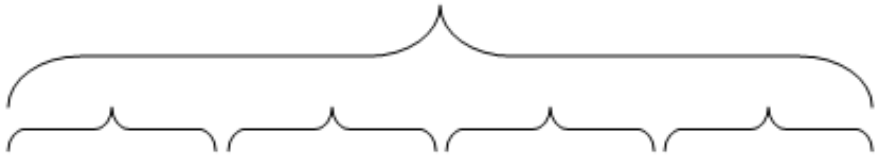
برای تبدیل جدول غیر نرمال فوق به یک جدول نرمال اول، بایستی کاری کنیم که خصیصه شماره تلفن فقط یک مقدار را بگیرد. یعنی:در جدول فوق می‌بینید که برای خصیصه شماره تلفن به ازای هر سطر فقط یک مقدار داریم.

Primary Key Primary Key

کد مشتری	نام مشتری	شماره تلفن
کد۱	مشتری۱	مقدار۱
کد۱	مشتری۱	مقدار۲
کد۲	مشتری۲	
کد۳	مشتری۳	مقدار۱
کد۴	مشتری۴	مقدار۱
کد۴	مشتری۴	مقدار۲
کد۴	مشتری۴	مقدار۳

در جدول غیر نرمال مثال پیشین چند مقدار برای یک خصیصه داشتیم. حال به مثالی می‌پردازیم که یک مجموعه از خصیصه‌ها چند

بار تکرار می‌شوند.
به جدول غیر نرمال زیر توجه کنید. دو خصیصه ترم و معدل چند بار در جدول تکرار می‌شوند. اصللاحا به این‌ها گروه‌های تکرار شونده می‌گویند.



کد دانشجو	نام دانشجو	ترم	معدل	ترم	معدل	ترم	معدل	ترم	معدل
کد۱	نام۱	۱	۱۴	۲	۱۵				
کد۲	نام۲	۱	۱۸	۲	۱۳	۳	۱۵	۴	۱۷
کد۳	نام۳	۱	۱۴	۲	۱۷	۳	۱۲		
کد۴	نام۴								

گروه‌های تکرار شونده را با آکولاد ({}) مشکل کرده ام. این گونه جداول (که حتی در شکل نرمال اول هم قرار ندارند) مشکلات فراوانی دارند که در زیر به مواردی اشاره خواهیم داشت:
چگونه معدل ترم 5 را در جدول درج کنیم؟ پس برای اینکه بتوانیم تمام معدل‌ها را در جدول داشته باشیم باید به تعداد حداکثر ترم تحصیلی گروه‌های تکرار شونده در جدول داشته باشیم.
برای دانشجویی که فقط یک ترم تحصیل کرده است تمام گروه‌های تکرار شونده به غیر از یکی خالی خواهد ماند. فضای بسیاری به هدر خواهد رفت.
گزارش گیری بسیار سخت خواهد شد. بطور نمونه، چطور می‌خواهید بالاتری معدل دانشجویان را بدست بیاورین؟

پس با تبدیل جدول غیر نرمال به شکل نرمال اول، به مشکلات فوق غلبه خواهیم کرد:

PK		PK	
کد دانشجو	نام دانشجو	ترم	معدل
کد۱	نام۱	۱	۱۴
کد۱	نام۱	۲	۱۵
کد۲	نام۲	۱	۱۸
کد۲	نام۲	۲	۱۳
کد۲	نام۲	۳	۱۵
کد۲	نام۲	۴	۱۷
کد۳	نام۳	۱	۱۴
کد۳	نام۳	۲	۱۷
کد۳	نام۳	۳	۱۲
کد۴	نام۴		

اما یک متغیر رابطه ای که فقط به صورت شکل اول نرمال است ساختاری دارد که به دلایل متعدد، نامطلوب است.

در جدول فوق اطلاعاتی وجود دارد که به دفعات تکرار شده است. مثلا نام دانشجو به تعداد ترم‌ها تکرار شده است. در صورتی که باید نام دانشجو یکبار ذخیره شده باشد. پس یک جدولی که به فرم نرمال اول هست می‌تواند افزونگی اطلاعات داشته باشد.

در بخش بعدی ابتدا وابستگی تابعی مورد بررسی قرار خواهد گرفت سپس به فرم دوم نرمال پرداخته خواهد شد.

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۳ ۱۶:۳۲

با تشکر از مطلب خوبتان.

این نوع مباحث رو با ormها و کلاس‌های دات نتی بهتر میشه توضیح داد. مثلا یک مشتری داریم با چندتا تلفن. یک دانشجو داریم با چندتا ترم و درس و نمره. این چندتا رو میشه به صورت یک icollection تعریف کرد در یک کلاس بجای اینکه پشت سر هم خاصیت اضافه کنیم. یا حتی زمانیکه مشتری سه تا تلفن داره مشکلی نداره تمامش در همان جدول اصلی قرار بگیره. در ef به این نوع‌ها، complextype گفته میشه (یک خاصیت تو در تو در کلاس، حالتیکه خاصیت کلاس خودش از نوع یک کلاس هست اما این کلاس تبدیل به یک جدول جدا نمیشه) یا مثلا در nhibernate به اون component mapping هم می‌گن.

نویسنده: سمیرا
تاریخ: ۱۳۹۲/۱۰/۱۰ ۲۱:۲۰

با سلام
میدونید چرا رابطه‌های دو صفته BCNFهستن؟

عنوان: **نرمال سازی (قسمت دوم: Second Normal Form)**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۳ ۱۷:۳۵
آدرس: www.dotnettips.info
گروه‌ها: second normal form ,نرمالسازی ,database design, normalization

وابستگی تابعی

برای وارد شدن به بحث نظری نرمالسازی نیاز هست با مفهوم وابستگی تابعی آشنا شویم.
وابستگی تابعی یک مبحث نسبتا مفصل و تئوری هست که زمان زیادی برای شرح جزئیات آن نیاز هست در نتیجه در حد آشنایی و نیازمان به آن توجه خواهیم داشت.

به جدول زیر نگاه کنید:

primary key		primary key	
S#	City	P#	Qty
S1	London	P1	100
S1	London	P2	100
S2	Paris	P1	200
S2	Paris	P2	200
S3	Paris	P2	300
S4	London	P2	400
S4	London	P4	400
S4	London	P5	400

این جدول نشان می‌دهد هر عرضه کننده(S#) چه قطعه (P#) را به چه تعداد (Qty) تولید کرده است. City هم شهریست که عرضه کننده در آن سکونت دارد.

از داده‌های فعلی جدول می‌شود برداشت‌های مختلفی داشت که چندتای آن به قرار زیر:
عرضه کنندگان یکسان دارای شهرهای یکسان هستند
هر عرضه کننده و قطعه تنها با یک مقدار از qty در تنظار است.

تعریف وابستگی تابعی یا functional dependency

تعریف رسمی:
اگر r یک رابطه و X و Y زیر مجموعه‌های دلخواهی از مجموعه خصیصه‌های r باشند آنگاه می‌گوییم Y به صورت تابعی وابسته به X است و آن را به صورت زیر می‌نویسیم:

X-->Y

اگر و تنها اگر در هر مقدار مجاز و ممکن از r، هر مقدار X متناظر با دقیقا یک مقدار از Y باشد. یعنی به ازای هر X تنها یک Y داشته باشیم. به بیان دیگر هرگاه دو چندتایی از r مقدار مقدار X یکسانی داشته باشند آنگاه مقدار Y آنها یکسان باشد .

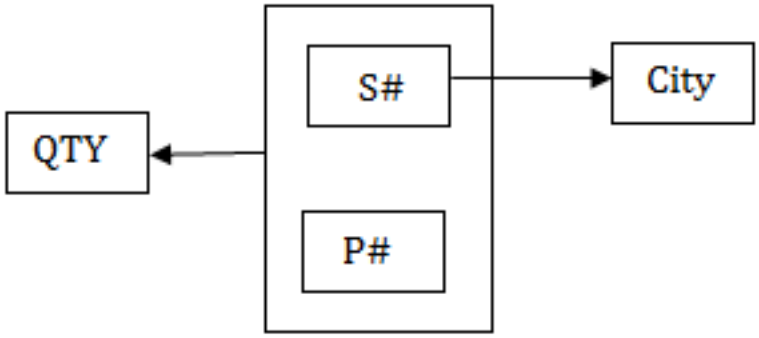
گفته شد که هر عرضه کنند تنها با یک شهر تناظر دارد. مثلا عرضه کنده ای با مقدار S1 تنها با شهر London در تناظر است. و به ازای هر عرضه کننده قطعه تنها یک QTY خواهیم داشت مثلا به ازای عرضه کننده با مقدار S4 و قطعه با مقدار P2 تنها یک سطر (در نتیجه یک Qty) وجود دارد (این دو خصیصه کلید هستند)

اما P# به S# وابستگی تابعی ندارد. مثلا به ازای S4 ما چند عرضه کننده خواهیم داشت.

وابستگی تابعی را می‌توان بشکل نمودار در آورد. در زیر نمودار وابستگی همراه با وابستگی‌های تابعی جدول مورد نظر آمده است:

$S\# \rightarrow City$

$\{S\#, P\#\} \rightarrow Qty$



تعریف شکل نرمال دوم

یک متغیر رابطه ای به شکل دوم نرمال است اگر و فقط اگر به شکل اول نرمال بوده و هر خصیصه غیر کلیدی وابسته به کلید اولیه باشد.

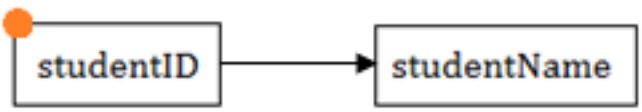
بر می‌گردیم به آخرین جدول مطلب گذشته یعنی:

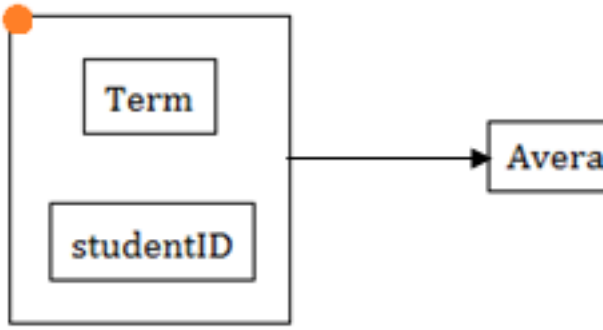
PK			PK
معدل	ترم	نام دانشجو	کد دانشجو
۱۴	۱	نام۱	کد۱
۱۵	۲	نام۱	کد۱
۱۸	۱	نام۲	کد۲
۱۳	۲	نام۲	کد۲
۱۵	۳	نام۲	کد۲
۱۷	۴	نام۲	کد۲
۱۴	۱	نام۳	کد۳
۱۷	۲	نام۳	کد۳
۱۲	۳	نام۳	کد۳
		نام۴	کد۴

کلید اولیه این جدول از ترکیب دو ستون کد دانشجو و ترم تشکیل شده است.
معدل را کلید اولیه تعیین می‌کند یعنی معدل وابسته به مقدار کلید اولیه است، اما نام دانشجو وابستگی به کلید اولیه ندارد و به جای آن وابسته به ستون کد دانشجو است. در نتیجه طبق تعریفی که داشتیم این جدول به شکل دوم نرمال نیست.
این جدول دقیقا مشابه به جدول عرضه کننده - قطعات است (که در ابتدا مطلب آمده است) پس نمودار FD آن نیز با FD این جدول برابر است.

برای تبدیل از فرم 1 به فرم 2 نرمال باید جدول را تجزیه کنیم به دو جدول:
جدول دانشجو (کد دانشجو - نام دانشجو)
جدول معدل (کد دانشجو - ترم - معدل)

به نمودار FD جدول فوق بعد از تجزیه شدن دقت بفرمایید:





همانطور که مشاهده می‌شود فلش‌ها تنها از خصیصه‌های کلید اولیه خارج شده اند در حالی که قبل از تجزیه شدن فلش ای وجو داشت که از کلید اولیه خارج نشده بود. کلیدهای اولیه توسط نقطه نارنجی رنگ علامت گذاری شده اند.

و بالاخره فرم دوم نرمال جدول سابق:

کد دانشجو	ترم	معدل
کد۱	۱	۱۴
کد۱	۲	۱۵
کد۲	۱	۱۸
کد۲	۲	۱۳
کد۲	۳	۱۵
کد۲	۴	۱۷
کد۳	۱	۱۴
کد۳	۲	۱۷
کد۳	۳	۱۲

کد دانشجو	نام دانشجو
کد۱	نام۱
کد۲	نام۲
کد۳	نام۳
کد۴	نام۴

کلیدهای اولیه با نقطه بنفش علامت گذاری شده است.

در اینجا با تجزیه جدول، به شکل سوم نرمال رسیدیم. در پست بعدی مثالی از یک جدول نرمال دوم خواهم آورد و همزمان با بررسی معایب آن شکل سوم نرمال را نیز معرفی خواهم نمود.

مرجع

کتاب پایگاه داده‌ی C.J. Date

نظرات خوانندگان

نویسنده: senaps
تاریخ: ۱۳۹۱/۱۱/۱۳ ۱۸:۴۷

خوب من خیلی خوشحالم....
من همیشه دیتابیسم رو به همین شکل طراحی میکنم! (یعنی حداقل جداولم حد نرمال دوم رو دارن! :))....
حالا تا ببینم در اینده چی‌میشه ماجرا که ببینم بر این اساس، ایا من کلا جداولم رو نرمال طراحی میکنم یا چی؟!
اخه من هیچوقت نرمال سازی رو یاد نگرفتم(البته تو دانشگاه هم درس نداد این مسئله رو استاد مربوطه...!) ولی خوب طراحی دیتابیس رو دوتایی با هم اینجوری کار کردیم که من معمولا مثل جدول‌های اخر این پست کار میکنم....

نویسنده: محمد سلم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۳ ۲۰:۵۸

این دو جدول آخر به شکل سوم نرمال هستند. یعنی شرط نرمال سوم را نیز محقق کرده اند. در مطلب بعدی یک مثال از جدولی خواهم آورد که به شکل دوم نرمال بوده ولی به شکل سوم نرمال نباشد.

نویسنده: حسینی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۱۷:۴

با سلام؛ شما ترکیب کد دانشجو و ترم رو کلید اصلی در نظر گرفتید؛ به نظرتون بهتره که کلید اصلی رو یه ستون جدا در نظر بگیریم یا همین کاری که شما انجام دادید؟ لطفا مزایا و معایب هر کدام را بفرمائید.
با سپاس فراوان

نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۲۲:۳۲

این امکان هم وجود داره که یک ستون دیگه به جدول اضافه کنید و آن را به عنوان PK در نظر بگیرید. اما باید به این نکته بسیار مهم نیز توجه داشته باشید که نمیشه آن دو ستون (کددانشجو و ترم) را همینطور به حال خود رها کرد. با این فرض که با اضافه شدن این ستون دیگه هیچ دو سطر تکراری به خاطر uniqueness بودن PK نخواهیم داشت.
شما لازمه که یک قید منحصربفرد تکریبی در کنار PK برای آن دو ستون در جدول ایجاد کنید. تا به ازای یک ترم معین و یک دانشجو معین تنها یک معدل ثبت بشه.
پس با لحاظ توضیحات فوق جدول به این شکل در می‌آید:

```
create table Avgs
(
   identifier int not null identity(1,1) primary key,
   student_id varchar(10) not null
        references Students
   term_id tinyint not null
        references Terms
   average tinyiny,
   check (averge between 0 and 20),
   unique (student_id, term_id)
)
```

ستونی به نام identity وظیفه PK را به عهده می‌گیره. و از نوع identity هم هست.
دو ستون کد دانشجو و کد معدل کلیدهای خارجی هستند. و ترکیب این دو ستون برای حفظ یکپارچگی و جامعیت داده‌ها منحصربفرد نظر گرفته شدن.
یک قید هم برای معدل گذاشته شده که معدل غیر متعارف در آن درج نشه.

به سناریوی زیر توجه کنید:

فرض کنید میخواهید بر اساس کد دانشجو و یک ترم معین در جدول برای بدست آوردم معدل جستجو داشته باشید. خب لازمه که بر اساس آن دو ستون جستجو داشته باشید نه آن ستونی که به عنوان PK در نظر گرفته شده. پس محتویات ستون identity کاملا مصنوعی و غیر طبیعی بوده و بطور مستقیم قابل استفاده نیست.

البته لازم به ذکر که عموما کلید اولیه همزمان unique clustered index نیز در نظر گرفته میشه. اگر داده‌های این ستون بطور متوالی و پشت سر هم در جدول درج نشن باعث ایجاد fragmentation میشه. و لازمه که ایندکس rebuild بشه.
و اگر کلید اولیه ترکیبی باشه کار در ارتباطات کمی دشوار میشه چون نیاز به کلیدهای خارجی ترکیبی نیز هست.
در joinها نیز چون پیوند بر اساس کلید اولیه و کلیدخارجی هست، هر چه کلید اولیه سبک‌تر باشه (حجم کمتری داشته باشه و از نوعی باشه که سریع‌تر توسط پردازنده پردازش بشه) سرعت پردازش نیز طبیعتا افزایش پیدا میکنه.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۲۷ ۰:۴۳

به این نوع کلیدها surrogate key هم می‌گن.

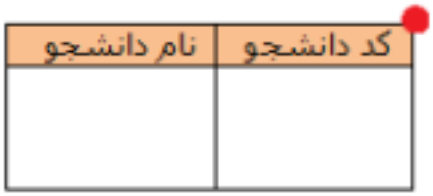
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۲/۰۵/۲۷ ۲:۲۱

بله همینطوره.
سایت ویکی توضیحات خوبی راجب معایب و مزایای استفاده از surrogate key داده.
یه مرور بسیار جزئی که به معایب و مزایا داشتم متوجه شدم که در پست قبلیم از معایبش به normalization و از مزایاش به performance اش اشاره ای داشتم.

عنوان: نرمال سازی (قسمت سوم: Third Normal Form)
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۳ ۲۱:۴۰
آدرس: www.dotnettips.info
برچسب‌ها: database design, normalization, third normal form

معایب شکل دوم نرمال

ابتدا اجازه دهید که مثالی از یک جدول بیاورم که به شکل دوم نرمال بوده ولی به شکل سوم نرمال نباشد. برای این منظور دو جدول زیر که هر دو در شکل سوم نرمال به سر می‌برند را با هم ترکیب می‌کنیم. ستون هایی از جدول که با نقاط قرمز رنگ علامت گذاری شده اند کلیدهای اولیه جدول می‌باشند.



اگر این دو جدول را با هم ترکیب کنیم، جدولی حاصل می‌شود که به فرم دوم نرمال است یعنی تمام خصیصه‌های غیر کلیدی وابسته به کلید اولیه (کد دانشجو) می‌باشند. اما همانطور که در بخش بعدی گفته خواهد شد، به شکل سوم نرمال نمی‌باشد.

کد دانشجو	نام دانشجو	نام رشته	نوع رشته
دانشجو۱	نام۱	رشته۱	نوع۱
دانشجو۲	نام۲	رشته۱	نوع۱
دانشجو۳	نام۳	رشته۱	نوع۱
دانشجو۴	نام۴	رشته۲	نوع۳
دانشجو۵	نام۵	رشته۲	نوع۳
دانشجو۶	نام۶	رشته۳	نوع۲

خصیصه "نوع رشته" به کلیداولیه جدول وابستگی تابعی دارد ولی از نوع متعدی (یعنی وابستگی از طریق خصیصه نام دانشجو می‌تواند بدست باید، چرا که نوع رشته به نام رشته و نام رشته به نام دانشجو وابستگی تابعی دارد)، این موضوع علاوه بر افزونگی اطلاعات باعث بی نظمی در به هنگام سازی خواهد شد. بطور نمونه

ایراد در عمل insert: این واقعیت که یک رشته خاص دارای یک نوع رشته خاص است را نمی‌توان اضافه کنیم، مثلا نمی‌توانیم بیان کنیم که رشته ریاضی از نوع علوم پایه است مگر آن که دانشجویی باشد در رشته ریاضی مشغول به تحصیل است.

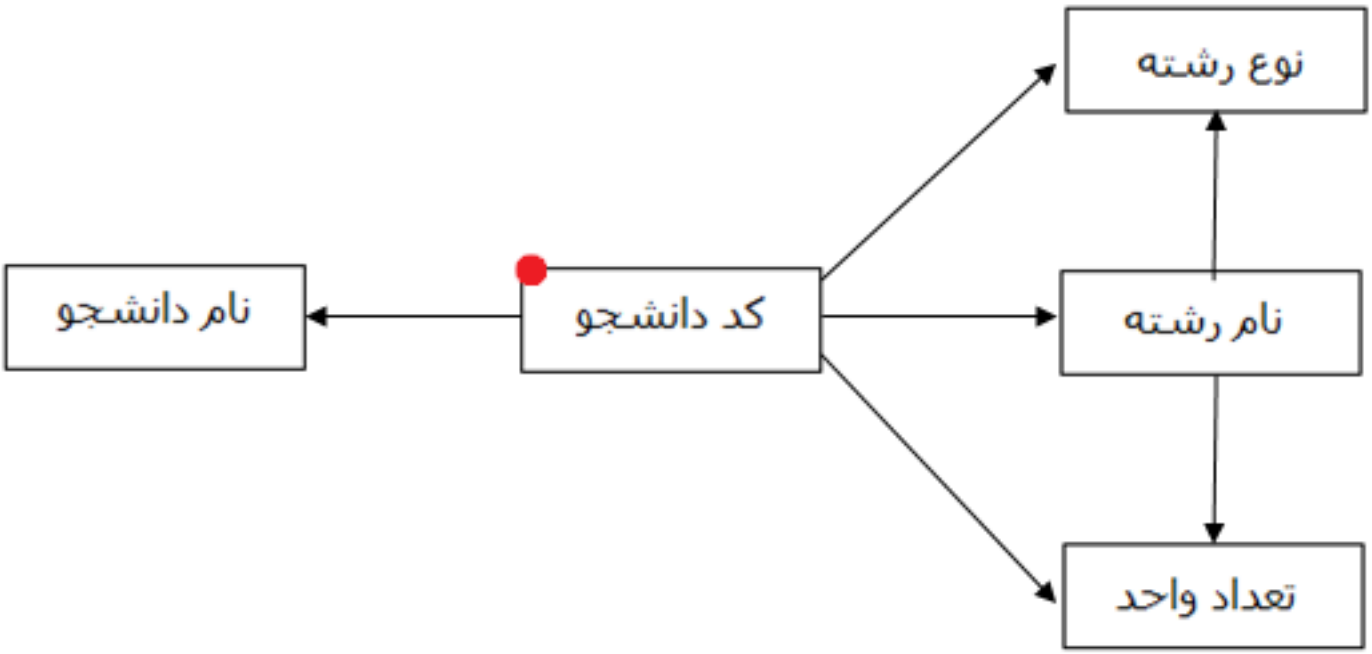
ایراد در عمل delete: با حذف یک دانشجو نه تنها اطلاعات مربوط به دانشجو بلکه اطلاعات مربوط به رشته تحصیلی نیز ممکن است حذف شود. مثلا با حذف سطر مربوط به دانشجوی شماره 6 تمام اطلاعات مربوط به رشته شماره 3 نیز حذف خواهد شد.

ایراد در عمل update: اگر فرضا بخواهیم نوع رشته ای به نام رشته1 را تغییر دهیم به جای یک سطر باید چندین سطر (سه سطر در داده‌های نمونه) را بروز رسانی کنیم.

تعریف شکل نرمال سوم

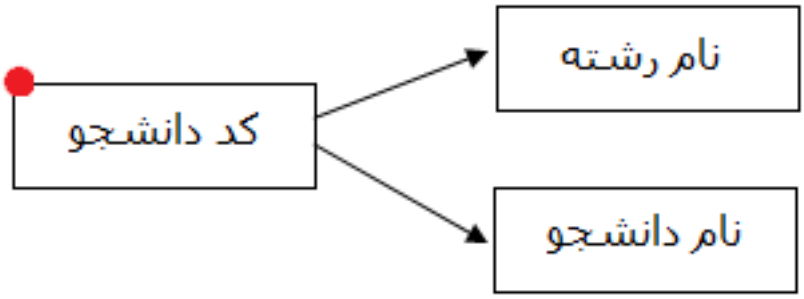
یک متغیر رابطه ای به شکل سوم نرمال است اگر به شکل دوم نرمال بوده و وابستگی‌های با واسطه (متعدی) نداشته باشد.

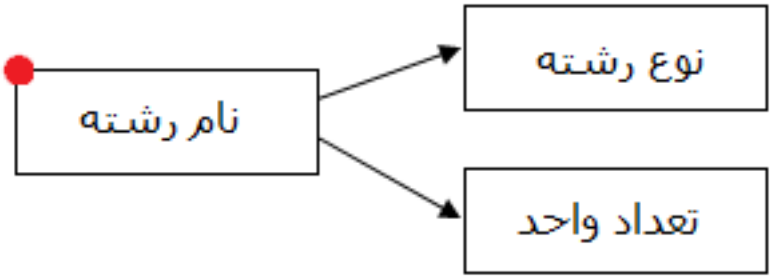
بر می‌گردیم به جدول ترکیبی قبل، نمودار FD جدول مورد نظر به صورت زیر است:



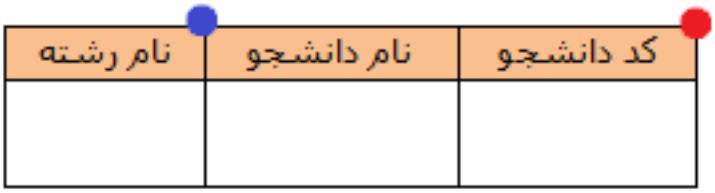
در این نمودار واضح است که وابستگی خصیصه نوع رشته به کد دانشجو از طریق خصیصه نام رشته بدست می‌آید. همینطور برای خصیصه "تعداد واحد". پس دو خصیصه‌ی نوع رشته و تعداد واحد با واسطه به کد دانشجو مرتبط هستند.

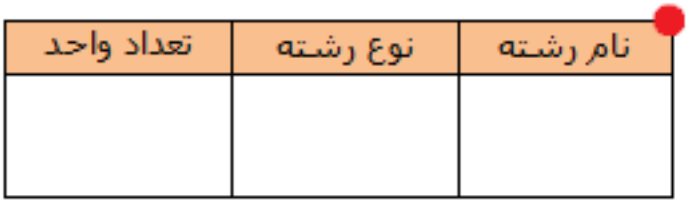
پس با تجزیه این نمودار به صورت زیر شرط شکل سوم نرمال هم محقق خواهد شد:





کافیه خصیصه کلید اولیه جدول "رشته ها" را به جدول "دانشجو" اضافه کنیم تا هر دو جدول به شکل نرمال سوم در بیایند.
نقطه قرمز به معنای کلید اولیه و نقطه آبی به معنای کلید خارجی می‌باشد:





موفق باشید

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۴ ۱:۷

خوب، اگر این سه قسمت رو بخواهیم با EF Code first مدل کنیم:

```
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

public class Student
{
    public int Id { set; get; }
    public string Name { set; get; }

    //هر دانشجو چند ترم در دانشگاه خواهد بود
    public virtual ICollection<Semester> Semesters { set; get; }
    //هر دانشجو چندین واحد دارد
    public virtual ICollection<Unit> Units { set; get; }
}

public class Semester
{
    public int Id { set; get; }
    public string Name { set; get; }
    public int Average { set; get; }

    [ForeignKey("StudentId")]
    public virtual Student Student { set; get; }
    public int StudentId { set; get; }
}

public class Unit
{
    public int Id { set; get; }
    public string Name { set; get; }
    public string UnitType { set; get; }
    public int NumberOfUnits { set; get; }

    [ForeignKey("StudentId")]
    public virtual Student Student { set; get; }
    public int StudentId { set; get; }
}
```

به نظر می‌رسه که خاصیت Average جاش در کلاس Semester نیست. حتی به Unit هم نباید به صورت مستقیم ارتباط پیدا کنه.
نیاز به یک کلاس دیگر هست که بتونه به ازای هر دانشجو، ترم و واحد، نمره ثبت کرد. میانگین، یک خاصیت آماری است که می‌تونه اصلا لحاظ نشه و در گزارشات محاسبه بشه.
و یا هر ترم یک سری واحد داره. اینطوری چطور؟ چون الان مشخص نیست در هر ترم چه واحدهایی برداشته.

نویسنده: محمد سلم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۴ ۸:۵۴

موضوعی که شما مطرح می‌کنید خارج از بحث مطرح شده است.
من تصمیم نداشتم که یک محیط عملیاتی را پیاده سازی کنم. تنها مثال هایی برای درک بهتر موضوع آوردم.
بله میانگین یک خاصیت آماری است، اما ما می‌توانیم برای سرعت بخشیدن به query هایمان برای بدست آوردن معدل، آن را بصورت فیزیکی ذخیره داشته باشیم. چون معدل بعد از ثابت و تعیین شدن دیگر تغییر نخواهد کرد.

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۴ ۹:۱۰

مشکلی که بودن میانگین در کلاس ترم ایجاد می‌کنه وابسته کردن آن به دانشجو است درحالیکه ترم باید یک موجودیت واحد و مستقل باشد. با این طراحی فعلی باید کل اطلاعات ثابت یک ترم به ازای هر دانشجو یکبار دیگر هم ثبت شود.

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۴ ۱۰:۱۲

یک نکته‌ای رو چند وقت قبل حین کار با ef بهش برخوردم که جالب بود. از دید ef ، کلیدخارجی یا کلید اصلی «فقط خواندنی» هستند. یعنی اگر در اینجا کسی بخواد نام رشته رو تغییر بده مشکل ساز خواهد شد.

به دو دلیل:

- استفاده از رشته‌ها نسبت به یک عدد چون طولانی‌تر هستند کندتر است برای حالت تعریف کلید

- اگر تغییری قرار است رخ دهد، باید به تمام جداول اعمال شود. (تعریف اطلاعات تکراری)