

مقدمه

نرمال سازی 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)

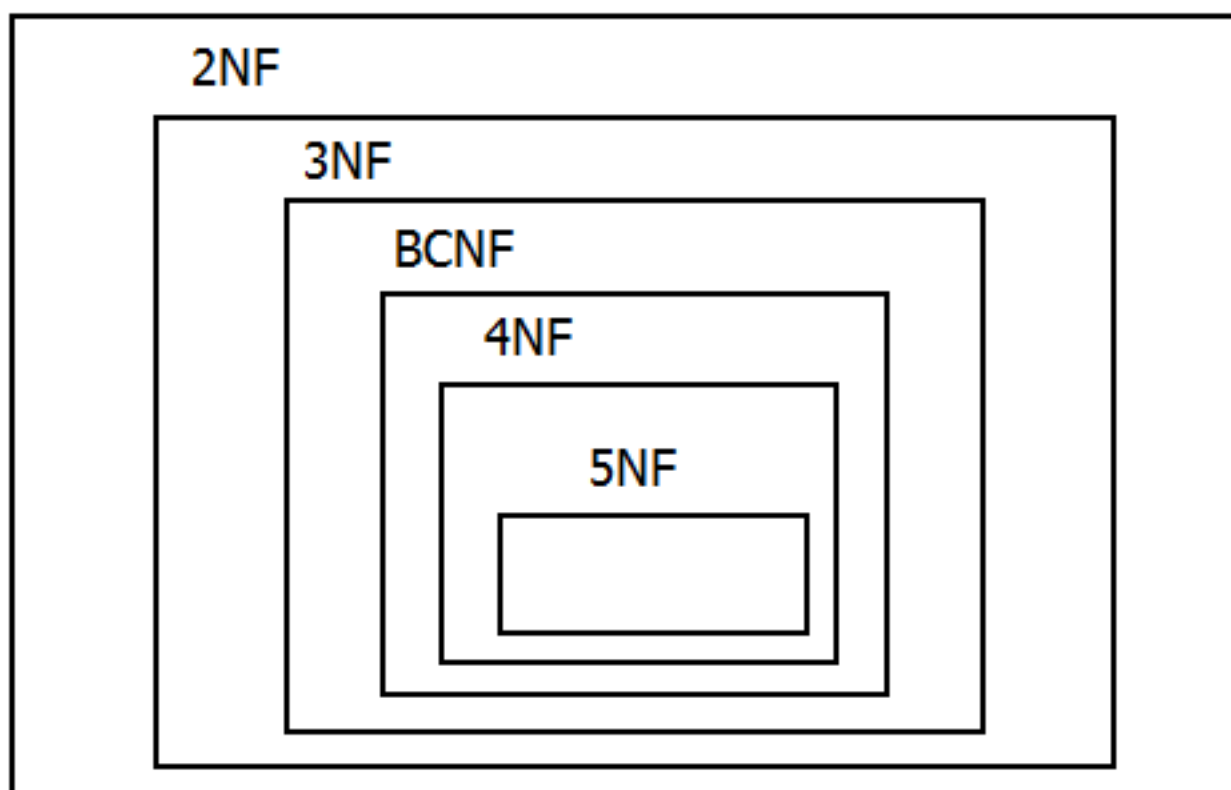
شکل نرمال بویس کاد (BCNF)

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).

1NF



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
کد ۱	مشتری ۱	مقدار ۱	مقدار ۲	
کد ۲	مشتری ۲			
کد ۳	مشتری ۳	مقدار ۱		
کد ۴	مشتری ۴	مقدار ۱	مقدار ۲	مقدار ۳

Primary Key		
کد مشتری	نام مشتری	شماره تلفن ها
کد ۱	مشتری ۱	مقدار ۱ - مقدار ۲
کد ۲	مشتری ۲	
کد ۳	مشتری ۳	مقدار ۱
کد ۴	مشتری ۴	مقدار ۱ - مقدار ۲ - مقدار ۳

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.

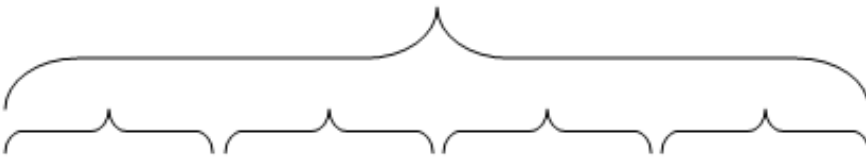
برای تبدیل جدول غیر نرمال فوق به یک جدول نرمال اول، بایستی کاری کنیم که خصیصه شماره تلفن فقط یک مقدار را بگیرد. یعنی: در جدول فوق می بینید که برای خصیصه شماره تلفن به ازای هر سطر فقط یک مقدار داریم.

Primary Key		Primary Key	
کد مشتری	نام مشتری	شماره تلفن	
کد ۱	مشتری ۱	مقدار ۱	
کد ۱	مشتری ۱	مقدار ۲	
کد ۲	مشتری ۲		
کد ۳	مشتری ۳	مقدار ۱	
کد ۴	مشتری ۴	مقدار ۱	
کد ۴	مشتری ۴	مقدار ۲	
کد ۴	مشتری ۴	مقدار ۳	

در جدول غیر نرمال مثال پیشین چند مقدار برای یک خصیصه داشتیم. حال به مثالی می پردازیم که یک مجموعه از خصیصه ها چند

بار تکرار می‌شوند.

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



کد دانشجو	نام دانشجو	ترم	معدل	ترم	معدل	ترم	معدل	ترم	معدل
کد ۱	نام ۱	۱	۱۴	۲	۱۵				
کد ۲	نام ۲	۱	۱۸	۲	۱۳	۳	۱۵	۴	۱۷
کد ۳	نام ۳	۱	۱۴	۲	۱۷	۳	۱۲		
کد ۴	نام ۴								

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 ۵ 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	
معدل	ترم	نام دانشجو	کد دانشجو
۱۴	۱	نام ۱	کد ۱
۱۵	۲	نام ۱	کد ۱
۱۸	۱	نام ۲	کد ۲
۱۳	۲	نام ۲	کد ۲
۱۵	۳	نام ۲	کد ۲
۱۷	۴	نام ۲	کد ۲
۱۴	۱	نام ۳	کد ۳
۱۷	۲	نام ۳	کد ۳
۱۲	۳	نام ۳	کد ۳
		نام ۴	کد ۴

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. مثلاً 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: ۱۶:۳۲ ۱۳۹۱/۱۱/۱۳

با تشکر از مطلب خوبتان.

این نوع مباحث رو با orm ها و کلاس های دات نتى بهتر ميشه توضيح داد. مثلاً يك مشتري داريم با چندتا تلفن. يك دانشجو داريم با چندتا ترم و درس و نمره. اين چندتا رو ميشه به صورت يك icollection تعريف كرد در يك كلاس بجای اينكه پشت سر هم خاصيت اضافه كنيم. يا حتى زمانيكه مشتري سه تا تلفن داره مشكلى نداره تماشى در همان جدول اصلى قرار بگيره. در ef به اين نوع ها، [complextype](#) گفته ميشه (يك خاصيت تو در تو در كلاس، حاليكه خاصيت كلاس خودش از نوع يك كلاس هست اما اين كلاس تبديل به يك جدول جدا نميشه) يا مثلاً در nhibernate به اون [component mapping](#) هم مى گن.

نویسنده: سمیرا

تاریخ: ۲۱:۲۰ ۱۳۹۲/۱۰/۱۰

با سلام

میدونید چرا رابطه های دو صفته BCNF هستن؟

وابستگی تابعی

برای وارد شدن به بحث نظری نرمال سازی نیاز هست با مفهوم وابستگی تابعی آشنا شویم. وابستگی تابعی یک مبحث نسبتاً 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 شهر است 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 انتظار 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:

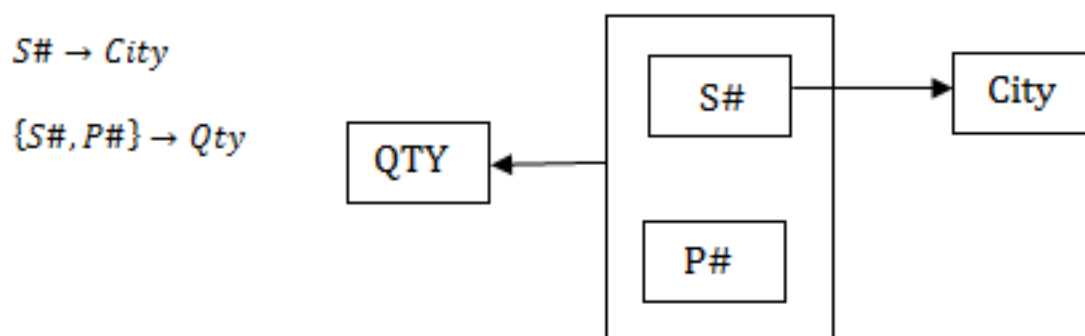
$X \rightarrow Y$

اگر و تنها اگر در هر مقدار مجاز و ممکن از r ، هر مقدار X متناظر با دقیقاً 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. مثلاً عرضه کننده ای با مقدار S1 تنها با شهر London در تناظر است. و به ازای هر عرضه کننده قطعه تنها یک QTY خواهیم داشت مثلاً 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. مثلاً 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:



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:

معدل	ترم	نام دانشجو	کد دانشجو
۱۴	۱	نام ۱	کد ۱
۱۵	۲	نام ۱	کد ۱
۱۸	۱	نام ۲	کد ۲
۱۳	۲	نام ۲	کد ۲
۱۵	۳	نام ۲	کد ۲
۱۷	۴	نام ۲	کد ۲
۱۴	۱	نام ۳	کد ۳
۱۷	۲	نام ۳	کد ۳
۱۲	۳	نام ۳	کد ۳
		نام ۴	کد ۴

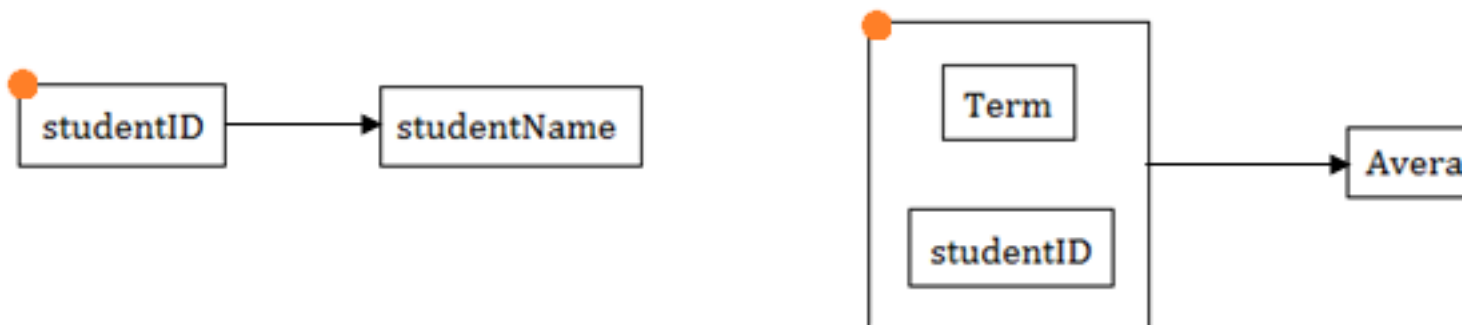
کلید اولیه این جدول از ترکیب دو ستون کد دانشجو و ترم تشکیل شده است. معدل را کلید اولیه تعیین می‌کند یعنی معدل وابسته به مقدار کلید اولیه است، اما نام دانشجو وابستگی به کلید اولیه ندارد و به جای آن وابسته به ستون کد دانشجو است. در نتیجه طبق تعریفی که داشتیم این جدول به شکل دوم نرمال نیست. این جدول دقیقا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

نام دانشجو	کد دانشجو
نام ۱	کد ۱
نام ۲	کد ۲
نام ۳	کد ۳
نام ۴	کد ۴

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

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 خواهیم آورد و همزمان با بررسی معایب آن شکل سوم نرمال را نیز معرفی خواهیم نمود.

مرجع

کتاب پایگاه داده‌ی C.J. Date

نظرات خوانندگان

نویسنده: senaps

تاریخ: ۱۸:۴۷ ۱۳۹۱/۱۱/۱۳

خوب من خیلی خوشحالم....

من همیشه دیتابیس رو به همین شکل طراحی میکنم! (یعنی حداقل جداولم حد نرمال دوم رو دارن!) حالا تا ببینم در آینده چی میشه 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 جدول های اخر این پست کار میکنم....

نویسنده: محمد سلم آبادی

تاریخ: ۲۰:۵۸ ۱۳۹۱/۱۱/۱۳

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: ۱۷:۴ ۱۳۹۲/۰۵/۲۶

با سلام؛ شما ترکیب کد دانشجو و ترم رو کلید اصلی در نظر گرفتید؛ به نظرتون بهتره که کلید اصلی رو به 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 آبادی

تاریخ: ۲۲:۳۲ ۱۳۹۲/۰۵/۲۶

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 همیشه آن دو ستون (کد دانشجو 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 منحصر بفرد تکریمی 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 می آید:

```
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 tinyint,
    check (average between 0 and 20),
    unique (student_id, term_id)
)
```

ستونی به نام identity وظیفه PK را به عهده می گیره. 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 داده ها منحصر بفرد 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 آوردن معدل جستجو داشته باشید. خوب لازم است که بر اساس آن دو ستون جستجو داشته باشید نه آن ستونی که به عنوان PK در نظر گرفته شده. پس محتویات ستون identity کاملاً مصنوعی و غیر طبیعی بوده و بطور مستقیم قابل استفاده نیست.

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

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

به این نوع کلیدها [surrogate key](#) هم می‌گویند.

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

بله همینطور. سایت ویکی توضیحات خوبی راجب معایب و مزایای استفاده از surrogate key داده. به مرور بسیار جزئی که به معایب و مزایا داشتیم متوجه شدیم که در پست قبلیم از معایب به normalization و از مزایای به performance اش اشاره ای داشتیم.

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.

کد دانشجو	نام دانشجو

نام رشته	نوع رشته	تعداد کل واحدها

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
دانشجو ۱	نام ۱	رشته ۱	نوع ۱
دانشجو ۲	نام ۲	رشته ۱	نوع ۱
دانشجو ۳	نام ۳	رشته ۱	نوع ۱
دانشجو ۴	نام ۴	رشته ۲	نوع ۳
دانشجو ۵	نام ۵	رشته ۲	نوع ۳
دانشجو ۶	نام ۶	رشته ۳	نوع ۲

خصیصه "نوع رشته" به کلید اولیه جدول وابستگی تابعی دارد ولی از نوع متعددی (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، مثلاً نمی‌توانیم بیان کنیم که رشته ریاضی از نوع علوم پایه است مگر آن که دانشجویی باشد در رشته ریاضی مشغول به تحصیل است.

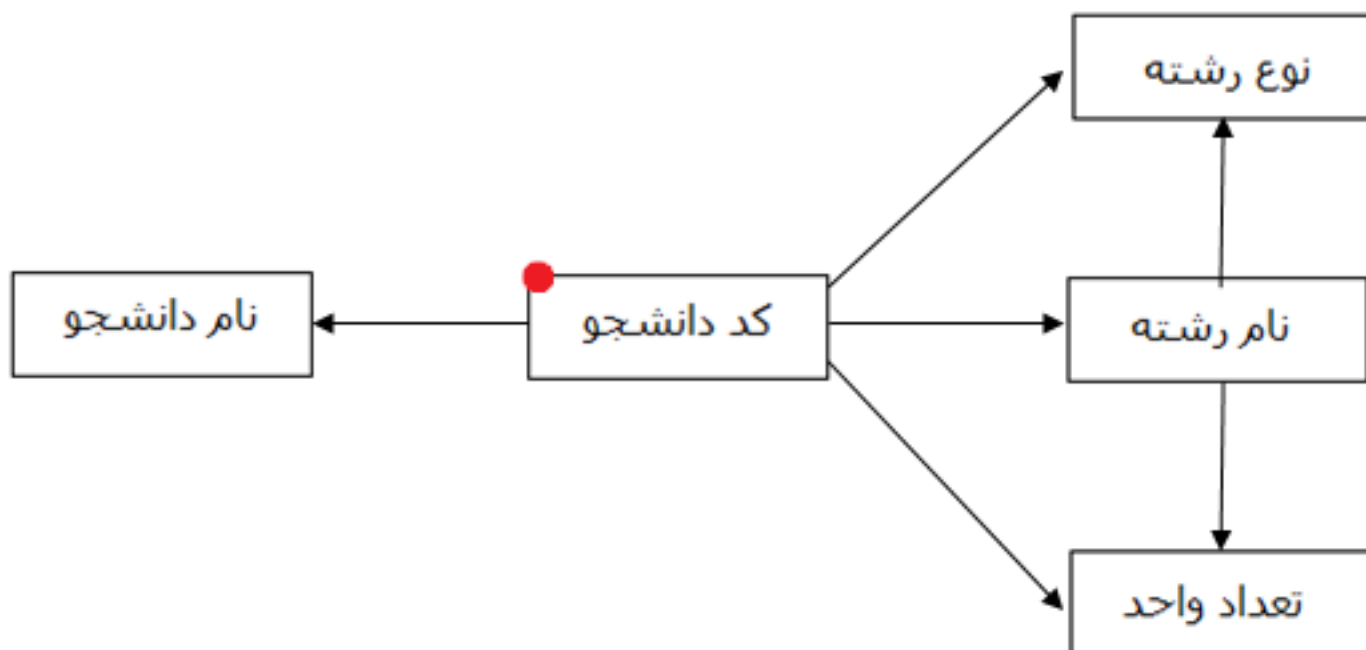
ایراد در عمل delete: با حذف یک دانشجو نه تنها اطلاعات مربوط به دانشجو بلکه اطلاعات مربوط به رشته تحصیلی نیز ممکن است حذف شود. مثلاً با حذف سطر مربوط به دانشجوی شماره 6 تمام اطلاعات مربوط به رشته شماره 3 نیز حذف خواهد شد.

ایراد در عمل update: اگر فرضاً بخواهیم نوع رشته ای به نام رشته 1 را تغییر دهیم به جای یک سطر باید چندین سطر (سه سطر در داده‌های نمونه) را بروز رسانی کنیم.

تعریف شکل نرمال سوم

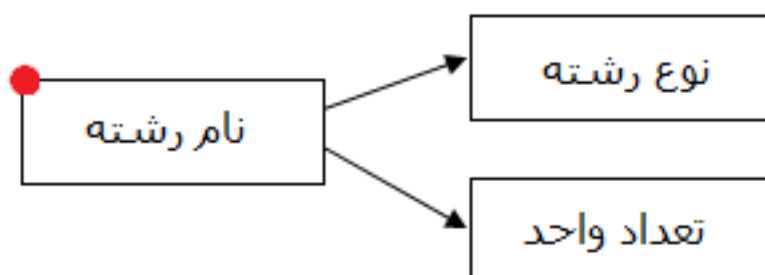
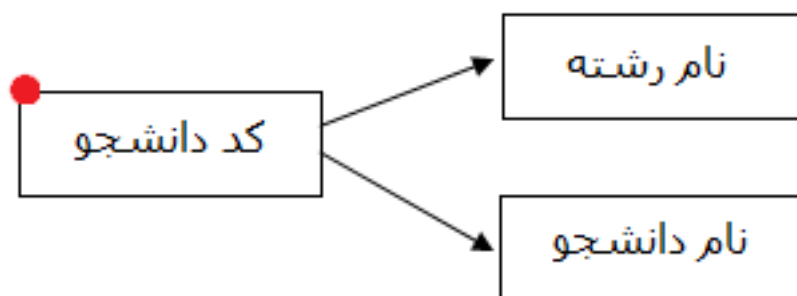
یک متغیر رابطه ای به شکل سوم نرمال است اگر به شکل دوم نرمال بوده و وابستگی‌های با واسطه (متعدی) نداشته باشد.

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 می باشد:

کد دانشجو	نام دانشجو	نام رشته

نام رشته	نوع رشته	تعداد واحد

موفق باشید

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۴ ۱:۷

خوب، اگر این سه قسمت رو بخوایم 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 می‌رسد 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 اصلاً 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.

یک نکته‌ای رو چند وقت قبل حین کار با ef بهش برخوردم که جالب بود. از دید ef ، کلید خارجی 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)