

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 می دهد 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 داده های فعلی جدول می شود برداشت های 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 مجموعه های دلخواهی از مجموعه خصیصه های r باشند آنگاه می گوییم Y به صورت تابعی وابسته به X است و آن را به صورت زیر می نویسیم:

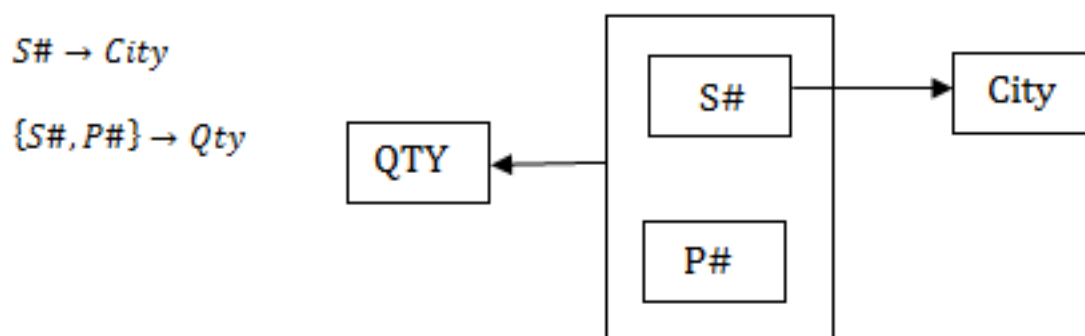
$X \rightarrow Y$

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 همنطور 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 مزایا به performance اش اشاره ای داشتیم.