

iranphp articles

بانکهای اطلاعاتی تراکنشی
رامین فرماني
ramin.farmani@gmail.com
.....

عنوان مقاله :
نگارنده :
آدرس پست الکترونیک :
تاریخ نگارش :

بانکهای اطلاعاتی تراکنشی:

می خواهیم در این مقاله برایتان از بانکهای اطلاعاتی تراکنشی صحبت کنیم در حقیقت این نام طولانی سطح پیشرفته بانکهای اطلاعاتی (در اینجا MySQL است ما در این مقاله مروری گذرا بر انواع Join ها خواهیم داشت تا اگر خدا بخواهد در پایان این مقاله متوجه بشویم که MySQL چیزی فراتر از یک کوری ساده و اینزرت و آپدیت کردن است هم چنان مانند گذشته اساس کار بر روی راهنمای رسمی MySQL قرار دارد که به عنوان مرجع به همه خوانندگان پیشنهاد می شود.

در ابتدا می خواهیم برایتان کمی از جوینها بگویم بعد جوین دو تیبیل را به هم و بعد چند تیبیل و همینطور ادامه می دهیم تا به یک دیدگاه کامل از جوین ها (صد البته در (MySQL برسیم :

متأسفانه اکثر کاربران تازه کار تصور می کنند بانکهای اطلاعاتی تفاوت چندانی با فایل ندارد و امکان بخصوصی نیز در آنها وجود ندارد در این مقاله سعی می کنم کمی با این تفکر مقابله کنم و تلنگری بزنم بر کسانی که هنوز احساس نیاز به مطالعه یک کتاب در مورد MySQL نکردند همینجا متذکر می شوم تمام صحبت هایی که در این مقاله گفته می شود تنها نگاهی گذرا به انبوهی از قابلیت ها و امکانات موجود در پایگاه های داده مانند MySQL است و به علاقه مندان توصیه می کنم جهت کسب اطلاعات بیشتر به منابع رسمی پایگاه های داده مورد نظر مراجعه کنند :

خوب حالا با هم شروع می کنیم :

فرض بفرمائین که دو تیبیل با ساختار زیر داریم
خوب حالا ما می خواهیم با داشتن نام یک نفر نام کمپانی و آدرس او را نیز بدست آوریم به عقیده شما چکار می توانیم بکنیم اگر تا بحال با جوین ها کار نکرده باشین راه حلی که ارائه می دهید گرفتن دو کوری ساده و کنترل یک به یک رکوردهاست اما من راه بهتری پیشنهاد می کنم ملاحظه کنین :

```
SELECT * FROM companies, contracts WHERE companies.company_id = contracts.company_id
```

خوب این نوع جوینی که دیدید یکی از ساده ترین و پایه ای ترین انواع جوین هاست که به اینر جوین (inner join) موسوم است همانطور که مشاهده می شود برای جوین کردن دو تیبیل از "استفاده کردم و هرجا ستونی را نیاز داشتم با فراخوانی نام تیبیلش از آن استفاده کردم همانطور که مشاهده می کنید روش کار بسیار ساده و واضح است شما می توانین با اضافه کردن "AND contracts.name = 'Jay Greenspan'" به انتهای کوری به دنبال یک نام مشخص هم بگردین .

خوب حالا می خواهیم با همین اینر جوین چندین تیبیل را به هم جوین کنیم توجه دارید که در بیشتر مواقع نیاز به متصل کردن چندین تیبیل به هم دارید که این مهم را نیز هم اکنون به اتفاق فرامیگیرم مجددا ساختاری مطابق زیر در نظر بگیرید .

اگر که بخواهید آدرس تمامی شرکتهایی که در ایالت کالیفرنیا هستند و مشاورین متخصص دارند را بدست آورید شما نیاز به متصل کردن هر چهار تیبیل به هم دارید که این نیز با اینر جوین به سادگی هر چه تمام تر قابل انجام است

```
SELECT * FROM companies, locations, experties, companies_experties WHERE state = 'CA' AND  
companies_experties.expertise_id = 3 AND companies.company_id =  
companies_expertise.expertise_id AND  
companies.company_id = location.company_id AND companies_expertise.expertise_id =  
expertise.expertise_id
```

خوب این مثال تقریباً یک مثال پیچیده بود و امیدوارم شما را از این نظر خسته نکرده باشم و کاملاً هم متوجه قضا و آنچه اتفاق می افتد شده باشید روال کار درست مانند اینر جوین قبلیست و به همین دلیل لزومی بر توضیح بیشتر نمی بینم بنابراین می رویم به سراغ اوتر جوین (outer join) مانند گذشته تصور کنید که دو تیبیل مانند زیر داریم :

خوب تیبیل سمت چپ بالا نام نویسندگان کتاب است که چنانچه متاهل باشند توسط آی دی در نظر گرفته شده به تیبیل همسران در سمت راست پائین متصل می شوند حال شما می خواهید نام این نویسندگان را به همراه همسرانشان اگر متاهل باشند نمایش دهید خوب اگر بخواهید مانند قبل از اینر جوین استفاده کنید داریم

```
SELECT * FROM contact, spouses WHERE contact.spouse_id = spouses.spouse_id
```

اما این جوین اینجا به کار ما نمی آید زیرا که نتیجه برگشتی از جوین بالا فقط شامل ردیف اول می شود اما ما می خواهیم نام تمامی نویسندگان چه مجرد و چه متاهل را داشته باشیم در این مواقع ما از جوینهایی موسوم به (Left Outer Join) بهره می گیریم که مطابق زیر استفاده می شود:

```
SELECT * FROM contact LEFT JOIN spouses ON contact.spouse_id = spouses.spouse_id
```

Outer Join ها در پایگاه های اطلاعاتی مختلف متفاوت است و ممکن است جوین هایی در یکی باشد که در دیگری وجود ندارد. . .
آخرین جوینی که می خواهیم برایتان توضیح دهم سلف جوین است (Self Join) فرض کنید تیبل ما چیزی شبیه این باشد:
خوب همانطور که می بینید در این تیبل ما چند نام داریم که چنانچه این افراد همسری داشته باشند آی دی همسرشان در spouse_id ذخیره شده است
می خواهیم ببینیم آیا کسی هست که شلوارش دوتا شده باشد خوب برا این کار ساده شاید مجبور باشید برای هر نام یک کوری جداگانه بگیرید و به هر حال حداقل مجبورید که دوبار کوری بگیرید آیا راه دیگری هست؟ بله سلف جوین این امکان را فراهم می کند که بسادگی این دو کوری را در یک کوری خلاصه کنیم
به کوری زیر توجه نمائید:

```
SELECT t1.first_name, t1.last_name, t2.first_name, t2.last_name FROM contacts t1,  
contacts t2 WHERE t1.spouse_id = t2.spouse_id AND t1.contact_id < t2.contact_id
```

تنها نکته ای که فکر می کنم لازم است توضیح دهم قسمت انتهایی کوری است در این کوری چون spouse_id هر کس با خودش برابر است جز نتایج به حساب می آورد که این مشکل را با قرار دادن قسمت آخر حل کردیم اما چرا باید t1.contact_id < t2.contact_id و چرا نباید t1.contact_id != t2.contact_id توضیح این مطلب ساده است اگر بجای < از != استفاده می کردیم یک بار t1 در elliot قرار می گرفت و t2 در john و در نتیجه می آمدند و بار دیگر elliot در t2 قرار می گرفت و john در t1 و مجددا در نتیجه کوری قرار می گرفتند به همین جهت به جای != از < استفاده کرده ایم

به انتها این مقاله کوتاه رسیدیم امیدوارم این تلاش موجب فراهم آمدن تفکری تازه در شما گردد و شاید آغازگر تلاشی نو. به هر شکل تمام تلاش من بر این بود تا با رعایت ایجاز در گفتار شما را با امکانی جدید آشنا کنم که امیدوارم اینگونه باشد.