همه چیز درباره ی Connection Pooling

به طور خلاصه، Connection Pooling تکنیکی است که به منظور استفاده از Connection های فعال موجود به جای ایجاد Connection جدید به دیتابیس، به منظور صرفه جویی در زمانی دستیابی به داده های بانک اطلاعاتی استفاده می شود.

تکنیک Pooling در طراحی سیستم های تجاری و بزرگ، ارزش و اهمیت خاصی پیدا می کند.

البته ممکن است در ظاهر، سیستم های طراحی شده بدون استفاده از این تکنیک دارای توان بازدهی خوبی باشند اما به طور حتم در دراز مدت از کارایی و بازدهی آنها کاسته خواهد شد.

به عنوان مثال متوجه خواهید شد که سیستم طراحی شده، قادر به پاسخگویی و پردازش همزمان هزار تراکنش در دقیقه توسط ده کاربر است اما توان عملیاتی همین سیستم در پردازش درخواست های هزار کاربر که ممکن است در هر دقیقه موجب ده تراکنش شوند، به شکل محسوسی افت خواهد کرد.

این مشکل از آنجا ناشی می شود که ایجاد هر Connection به دیتابیس، اصطلاحا موجب ایجاد **سربار** بر روی سیستم کلاینت (یا در برنامه های وب، بر روی سرور) می شود.

این سربار به دلیل انجام مراحل مختلف قبول درخواست Connection به دیتابیس به وجود می آید; همانند تایید مجوز دسترسی به دیتابیس.

بیشتر مکانیزم هایی که به منظور ارتباط با دیتابیس ها مورد استفاده قرار می گیرند، به منظور جلوگیری از ایجاد این سربار، از تکنیک Connection Pooling استفاده می کنند.

Pool در فرهنگ لغت به معنای استخر و حوض و در جای دیگر به معنای استفاده مشترک از چیزی، بیان شده است.

همان طور که قبلا گفته شد، Connection Poolingاز Connection های فعال و موجود، جهت ایجاد ارتباط با دیتابیس استفاده می کند و در حقیقت لیستی از Connection های موجود را برگشت می دهد.

در تکنیک Pooling ، زمانی که کلاینت درخواست ایجاد ارتباط با دیتابیس را به برنامه می فرستد، به جای ایجاد مجددConnection ، این Connection از Pool استخراج شده و عملیات اتصال انجام می پذیرد.

شایان ذکر است که درگیری کمتر CPU با منابع و صرفه جویی در پهنای باند شبکه، از مزایای دیگر استفاده از این تکنیک است.

درADO.NET ، کلاس یا شی خاصی به منظور استفاده از تکنیک Connection Pooling وجود ندارد، در عوض، پرووایدرهایی که همراه با .Net Framework عرضه شده اند، به طور خودکار این عمل را انجام می دهند.

در پرووایدرهای SQL Server وOracle ، این تکنیک به طور خودکار اعمال و اجرا می شود.

در پرووایدرهای OLE DB وODBC ، این تکنیک با محدودیت و قابلیت های کمتری انجام می پذیرد.

توجه: هر چند که تکنیک Connection Pooling در اکثر پرووایدرها به شکل خودکار انجام می پذیرد، اما برنامه نویس قادر به ایجاد تنظیماتی در Connection String به منظور تغییر در برخی خواص Pooling می باشد.

اولین کاری که در این تکنیک انجام می پذیرد، پردازش Connection String ارسال شده به برنامه است. Connection String بررسی می شود. در صورتی که Connection String ای **در Pool** مطابق با Connection Stringارسال شده توسط کلاینت پیدا شود، از Connection String موجود در Pool جهت اتصال به دیتابیس استفاده خواهد شد.

اگر هیچ Connection ای در Pool وجود نداشته باشد، Connection String دریافت شده به منظور انجام عمل تطابق با درخواست های بعدی، در Pool قرار خواهد گرفت.

توجه مهم Connection Pooling :به بزرگ و کوچک بودن حروف، حساس است.

همچنین، Connection Pooling به ترتیب قرار گیری پارامترهای Connection String نیز حساس است.

به عنوان مثال، اگر دو Connection String یکسان با رعایت حروف بزرگ و کوچک و تعداد پارامترها به برنامه ارسال شود اما ترتیب قرار گیری پارامترها در دو Connection String متفاوت باشد، تکنیک Connection Poolingاعمال نمی شود. (پرووایدر ODP.NET که همان پرووایدر دیتابیس Oracle می باشد، از این قائده مستثنا است)

نکته مهم: به منظور اطمینان از اعمال تکنیکPooling ، حتما Connection String را در یک فایل یا رجیستری قرار داده و از نوشتن مستقیم آن در کد برنامه اجتناب کنید.

برنامه نویسان ASP.NET باید Connection String را در فایل Web.Config قرار دهند.

مثال زیر نحوه عمل تکنیک Connection Pooling را مشخص می کند:

کد:

```
string conString1 = "Data Source=localhost;" +
  "Initial Catalog=FirstDB;Integrated Security=SSPI");

string conString2 = "Data Source=localhost;" +
  "Initial Catalog=SecondDB;Integrated Security=SSPI");

SqlConnection con = new SqlConnection();

con.ConnectionString = conString1;
con.Open();
```

در اینجا (با فرض اینکه Pool خالی است و برای اولین بار درخواست اتصال به دیتابیس ارسال می شود)، رشته حاوی conString1 در Pool قرار خواهد گرفت. پس یک Pool ایجاد شد. نام آن را Pool A می گذاریم.

کد:

```
con.Close()
```

Connectionبه Pool A برگشت داده می شود.

کد:

```
con.ConnectionString = conString2;
con.Open();
```

دو خط فوق را در نظر بگیرید.
در خط اول، Connection String ای به شی con نسبت داده می شود که با Connection String مرتبه اول (conString1) متفاوت است.
به همین دلیل، در هنگام ارسال درخواست اتصال به برنامه، یک Pool جدید که حاوی conString2 است ایجاد می شود. نام آن را Pool B می گذاریم.

کد:

```
con.Close()
```

Connectionبه Pool B برگشت داده می شود.

کد:

```
con.ConnectionString = conString1;
con.Open();
```

حال، دو خط فوق را در نظر بگیرید.

از آنجایی که قبلا conString1 به عنوان یک Pool در مخزن Pool ها قرار گرفته، نیاز به ایجاد مجدد Connectionنیست و Pool A استخراج شده و از آن جهت برقراری ارتباط با دیتابیس استفاده می شود. (صرفه جویی در زمان)
کد:

```
con.Close()
```

Connectionبه Pool A برگشت داده می شود.

نکاتی در مورد مثال فوق:

۱) در هنگام فراخوانی متدClose ، کانکشن فعال در Pool قرار خواهد گرفت.

در هنگام فراخوانی متدOpen ، پرووایدر، مخزن Pool ها را به منظور یافتن یک Pool مطابق با Connection String ارسالی جستجو می کند.

۲) استفاده از Pool تنها زمانی انجام می پذیرد که دو Connection دقیقا شبیه به هم باشند.

۳) گر هیچ Pool ای در مخزن Pool ها وجود نداشته باشد، اولین Pool در زمانی فراخوانی متد Open ایجاد شده و در مخزن Pool ها قرار خواهد گرفت.

به طور پیش فرض، حداکثر ۱۰۰ Pool در مخزن Pool ها قرار خواهد گرفت. (این مقدار توسط برنامه نویس قابل تغییر است)

تکنیک Connection Pooling در دیتابیس های SQL Server وOracle :

مدیریت Pooling در دیتابیس های SQL Server و Oracle به طور خودکار توسط پرووایدرهای آنها انجام می پذیرد. اما برنامه نویس می تواند با ایجاد تغییراتی درConnection String ، برخی از خواص آن را به دلخواه تغییر دهد.

موفق باشید.

نویسنده : بهروز راد

کلیه حقوق این مقاله متعلق به نویسنده و سایت برنامه نویس مي باشد .
استفاده از مطالب این مقاله در صورت ذکر منبع مجاز است .
www.barnamenevis.org