**آیا زبان برنامه نویسی انتخاب شده برنامه نویسی همروند را پشتیبانی می کند؟ در اینصورت در مورد سازوکارهای موجود از جمله سمافورها، قفل ها، مکانیزم های ارسال پیام، و ریسه ها، و دیگر سازوکارهای موجود برای پشتیبانی برنامه نویسی همروند توضیح دهید.**

sql به طور مستقیم برنامه‌نویسی همروند را پشتیبانی نمی‌کند، اما برخی از سیستم‌های مدیریت پایگاه داده که sql را پیاده‌سازی می‌کنند، قابلیت‌هایی را برای اجرای همروند برنامه‌های sql فراهم می‌کنند.

برنامه‌نویسی همروند به این معنی است که چندین برنامه یا فرآیند به طور همزمان یا موازی اجرا شوند. این می‌تواند منجر به افزایش کارایی و عملکرد برنامه‌ها شود، اما همچنین می‌تواند مشکلاتی را در مورد هماهنگی، همزمانی و امنیت داده‌ها ایجاد کند.

به عنوان مثال، اگر چندین برنامه sql به طور همزمان به یک پایگاه داده دسترسی داشته باشند و بخواهند داده‌های یکسان را بخوانند یا تغییر دهند، ممکن است با مشکلاتی مانند رقابت، تداخل، بلوکه شدن یا مردود شدن مواجه شوند. برای حل این مشکلات، سیستم‌های مدیریت پایگاه داده مکانیزم‌هایی را برای کنترل دسترسی همروند به داده‌ها ارائه می‌دهند. این مکانیزم‌ها عبارتند از:

* قفل‌ها: قفل‌ها ابزاری هستند که برای جلوگیری از دسترسی همزمان به یک داده یا منبع توسط چندین برنامه استفاده می‌شوند. قفل‌ها می‌توانند از نوع انحصاری یا اشتراکی باشند. قفل انحصاری به این معنی است که فقط یک برنامه می‌تواند به داده دسترسی داشته باشد و هیچ برنامه دیگری نمی‌تواند آن را بخواند یا تغییر دهد. قفل اشتراکی به این معنی است که چندین برنامه می‌توانند به داده دسترسی داشته باشند و آن را بخوانند، اما هیچ برنامه‌ای نمی‌تواند آن را تغییر دهد.
* معاملات: معاملات واحدهای منطقی از عملیات sql هستند که باید به طور کامل اجرا شوند یا اصلا اجرا نشوند. معاملات باید چهار خاصیت ACID را داشته باشند: اتمی بودن، سازگاری، ایزولاسیون و دوام. اتمی بودن به این معنی است که یا تمام عملیات معامله انجام می‌شوند یا هیچکدام انجام نمی‌شوند. سازگاری به این معنی است که معامله باید داده‌ها را از یک حالت سازگار به حالت سازگار دیگر ببرد. ایزولاسیون به این معنی است که معامله باید از تأثیر معاملات دیگر جدا شود. دوام به این معنی است که نتایج معامله باید به طور دائمی در پایگاه داده ذخیره شوند.
* سطح ایزولاسیون معامله: سطح ایزولاسیون معامله مشخص می‌کند که چه میزان تداخل بین معاملات مجاز است. سطح‌های مختلف ایزولاسیون می‌توانند مشکلات مختلف همروندی را حل یا ایجاد کنند. مشکلات همروندی عبارتند از: خواندن ناپایدار، خواندن ناتمام، خواندن فریبنده و نوشتن فریبنده. سطح‌های ایزولاسیون معامله عبارتند از: خواندن نامتعارف، خواندن تکرار شونده، خواندن تأیید شده و سریال.
* مکانیزم‌های دیگر: برخی از سیستم‌های مدیریت پایگاه داده مکانیزم‌های دیگری را برای پشتیبانی برنامه‌نویسی همروند در sql ارائه می‌دهند. برای مثال،

Oracle E-Business Suite از API‌های پردازش همروند استفاده می‌کند که امکان اجرای همروند برنامه‌های sql را با استفاده از صف‌های پیام و سمافورها فراهم می‌کند. همچنین، برخی از سیستم‌ها از روش‌هایی مانند برچسب‌زمان، چندنسخه‌ای و اعتبارسنجی برای کنترل همروندی استفاده می‌کنند.

* ریسه: ریسه‌ها در SQL به عنوان یک مکانیزم قفل‌گذاری عمل می‌کنند که می‌توانند به فرایندها یا ریسه‌ها اجازه دهند که به بخش‌های مختلف یک منبع دسترسی پیدا کنند. برای مثال، اگر یک فرایند یا ریسه بخواهد یک ستون خاص را در یک جدول به روز رسانی کند، می‌تواند یک ریسه را بر روی آن ستون قرار دهد. این کار باعث می‌شود که هیچ فرایند یا ریسه دیگری نتواند به آن ستون دسترسی پیدا کند و تغییری در آن ایجاد کند. اما این کار مانع از دسترسی به بقیه ستون‌ها یا ردیف‌های جدول نمی‌شود. این روش می‌تواند از تداخل بین عملیات‌های مختلف جلوگیری کند و از ایجاد تراکنش‌های ناقص جلوگیری کند.
* سمافور: سمافورها در SQL به عنوان یک متغیر صحیح عمل می‌کنند که تعداد فرایندها یا ریسه‌هایی را نشان می‌دهند که می‌توانند به یک منبع مشترک دسترسی پیدا کنند. برای مثال، اگر یک سمافور مقدار ۳ داشته باشد، به این معنی است که فقط ۳ فرایند یا ریسه می‌توانند به یک جدول یا یک ستون خاص در پایگاه داده دسترسی داشته باشند. وقتی یک فرایند یا ریسه به منبع مورد نظر دسترسی پیدا می‌کند، سمافور یک واحد کاهش می‌یابد. وقتی یک فرایند یا ریسه از منبع استفاده کرده و آن را رها می‌کند، سمافور یک واحد افزایش می‌یابد. اگر سمافور مقدار صفر داشته باشد، به این معنی است که هیچ فرایند یا ریسه دیگری نمی‌تواند به منبع دسترسی پیدا کند و باید منتظر بماند تا یکی از فرایندها یا ریسه‌های فعال منبع را رها کند. این روش می‌تواند از رقابت برای دسترسی به منابع جلوگیری کند و از ایجاد داده‌های ناهماهنگ جلوگیری کند.

**منابع:**

* [https://www.sqlshack.com](https://www.sqlshack.com/concurrency-problems-theory-and-experimentation-in-sql-server/)
* <https://dba.stackexchange.com>
* <https://docs.oracle.com>
* <https://www.sqlshack.com>
* <https://reintech.io>