每个DBMS工作者采用的不同进程模型分为从简单到复杂的三种：每个DBMS工作者采用一个进程；每个DBMS工作者采用一个线程；进程池；尽管被简化了但是商业DBMS一般采用这三种

#################################

**2.1.1 每个DBMS工作者拥有一个进程**

每个DBMS工作者拥有一个进程的模型（如图2-1所示），早期被用于DBMS开发，并且今天仍被一些商业系统所采用。这个模型由于DBMS工作者被直接映射到系统进程，因而相对容易实现。系统调度管理DBMS工作者的运行时间，而DBMS编程人员也可以利用系统的一些保护措施来排除标准错误，如内存溢出。而且，不同的编程工具，比如调试器和存储检测器，都与这种进程模型相适应。使这个模型变复杂的是DBMS连接所需要的内存中的数据结构，包括锁机制和缓冲池（在第6.3节和第5.3节将详细讨论）。这些共享的数据结构必须分配在所有DBMS进程都可以访问的共享内存中。这需要系统支持（一般系统均支持）和DBMS相关的代码。实际中，由于这种模型需要广泛使用共享内存，使得地址空间分离的优势被削弱了。

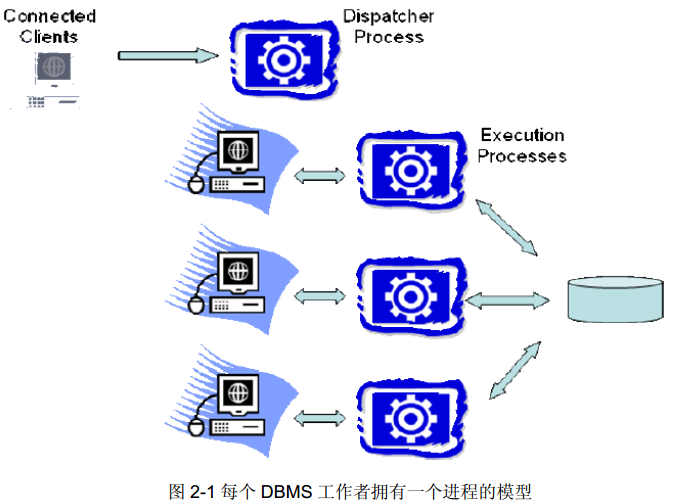


图2-1每个DBMS工作者拥有一个进程的模型

如果把“扩展到大量并发连接”作为衡量标准，那么，每个DBMS工作者拥有一个进程的模型并不十分有效。这是因为，进程相对于线程而言，拥有更多的环境变量并且消耗更多的存储空间。进程切换需要切换安全的上下文环境、存储空间变量、文件和网络句柄列表以及其他一些进程上下文。这在线程切换时是不需要的。尽管如此，每个DBMS工作者拥有一个进程的模型，还是比较受欢迎的，并得到IBM DB2、PostgreSQL和Oracle的支持。

**2.1.2 每个DBMS工作者拥有一个线程**

在每个DBMS工作者拥有一个线程的模型（图2.2）中，一个多线程进程负责所有的DBMS工作者的工作。一个调度线程监听新的DBMS客户端连接。每个连接都被分配一个新的线程。当每个客户端提交SQL请求时，该请求都由对应的DBMS工作者线程来执行。这个线程在DBMS进程中运行，一旦运行结束，结果将返回给客户端，然后该线程等待来着这个客户端的下一个请求。

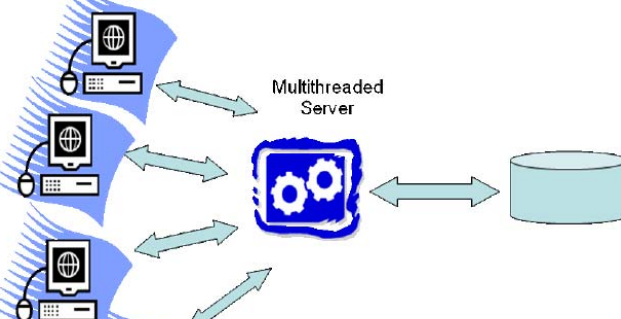


图2-2每个DBMS工作者拥有一个线程的模型

多线程编程在这种架构下有以下困难：操作系统对线程不提供溢出和指针的保护；调试困难，尤其是在运行情况下更是如此；由于不同操作系统在线程接口和多线程扩展性方面的不同，使得软件可移植性较差。“每个DBMS工作者拥有一个线程模型”中的很多编程问题，同样存在于每个DBMS工作者拥有一个进程模型中，因为它们都需要共享内存的使用。

尽管不同操作系统在线程API的多样性方面，近年来不断缩小，但不同平台之间微小的差别还是不断带来调试上的麻烦。如果不考虑实现上的困难，那么，每个DBMS工作者拥有一个线程的模型，还是可以很好地扩展到高并发系统中的，并且这种模型在一些现代DBMS产品中也被使用了，如IBM DB2、微软SQL Server、MySql、Informix和Sybase。

**2.1.3 进程池**

这个模型是“每个DBMS工作者一个进程”这种模型的变体。我们知道，每个DBMS工作者拥有一个进程的模型的优点是编程实现较为简单，但是，每一个连接都需要一个进程却是一个缺点。使用进程池（如图2-3所示）后，不必每个DBMS工作者都分配一个进程，而是由进程池来管理所有DBMS工作者。一个中央进程控制所有的DBMS客户端连接，每个从客户端到来的SQL请求将被分配一个进程池中的进程。SQL请求处理完之后，结果返回客户端，进程回到缓冲池中准备分配给下一个请求。缓冲池的大小是一定的，且通常不可变。如果一个请求到来而没有进程空闲，那么新的请求必须等待进程。

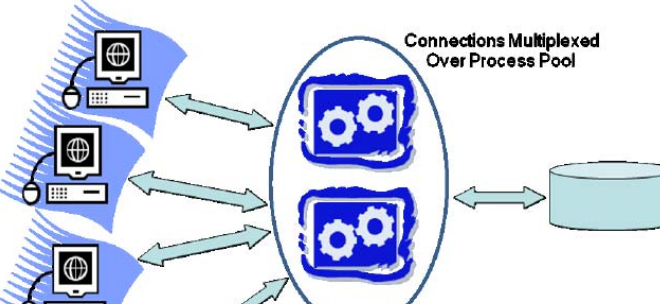


图2-3 进程池

进程池拥有每个DBMS工作者一个进程模型的所有优点，而且，由于只需要少量的进程，所以，其内存使用效率也很高。进程池一般被设计成大小可动态变化的，以此应对大量的高并发请求。当请求负载较低时，进程池可以缩减为较少的等待进程。跟每个DBMS工作者一个进程一样，很多现代DBMS产品也支持进程池

############################################

**每个DBMS工作者采用一个进程**： DB2在不支持线程的系统采用这种方式，支持线程的系统采用每一个工作者采用一个线程，这也是Oracle的进程模型默认设计。Oracle也支持上文的进程池。PostgreSQL在所有的系统都只运行每个DBMS工作者拥有一个进程模型。

**每个DBMS工作者采用一个线程**：这种还分为两种：

1**每个DBMS工作者拥有一个系统线程**：IBM DB2运行在良好的线程支持系统上采用这种设计，MySQL也是采用这种设计；

2**每个DBMS工作者采用一个DBMS线程**：这种的DBMS工作者调度通过系统进程或者线程去调度轻量级线程实现的。规避了一些系统调度的一些问题但是开发成本高，没有良好的开发工具，需要运营商长期维护。

它有两种子模型：

○1通过系统进程来调度DBMS线程：轻量级线程的调度是由一个或多个系统进程来完成的。Sybase和Informix支持该模型。现今很多系统采用这个模型来开发实现DBMS线程的调度，以此调度DBMS工作者发挥多处理器的作用。然而，并不是所有采用这个模型的系统都实现了线程迁移：将DBMS线程再分配给不同的系统进程（如考虑到负载均衡）。

○2通过系统线程来调度系统进程：微软SQL Sever以可选择的方式支持这种模型（默认的模型为DBMS工作者多路复用线程池）。SQL Sever中这个选项叫做Fibers，被用做应对高频事务处理，但是很少被使用。

**进程/线程池：**

在这个模型中，DBMS工作者共用一个进程池。随着系统线程支持性能的不断提高，依

赖线程池而不是进程池的新的变种开始出现。在后面这种变种模型中，DBMS工作者复用

系统的一个系统线程池：

1. DBMS工作者共用进程池：这个模型相对于每个DBMS工作者拥有一个进程的模型

而言，有更高的内存使用效率。在系统没有良好的线程支持的情况下，它也易于与

系统对接，并且在大量用户的情况下表现稳定。这个模型是Oracle数据库的可选项，

Oracle建议在大量用户并发操作的情况下使用。Oracle默认的模型是每个DBMS

工作者拥有一个进程。这两个选项对于Oracle所适用的大多数操作系统而言都是支

持的。

2. DBMS工作者共用线程池：微软SQL Server默认使用该模型，且超过99%的SQL

Server产品使用该模型来运行。为有效支持上万的并发连接用户，SQL Server提

供可选支持，允许系统线程调度DBMS线程。