Microsoft®
SQL Server™ 2005

SQL Server Service Broker 소개

Microsoft®

요약

본 백서에서는 Microsoft SQL Server 2005에 새로 추가된 Service Broker에 대해 소개합니다. 새로운 Service Broker 기능을 통해 내부 또는 외부 프로세스에서 Transact-SQL에 대한 확장을 사용하여 보장된 비동기 메시지를 주고 받을 수 있습니다.

Contents

들어가는 글

Microsoft SQL Server 2005의 기능 중 하나인 Service Broker를 통해 내부 또는 외부 프로세스에서 Transact-SQL DML(Data Manipulation Language)에 대한 확장을 사용하여 보장된 비동기 메시지를 주고 받을 수 있습니다. 메시지는 보낸 사람과 동일한 데이터베이스에 있는 대기열, 동일한 SQL Server 인스턴스에 있는 다른 데이터베이스 또는 동일한 서버나 원격 서버에 있는 다른 SQL Server 인스턴스로 보내집니다.
Service Broker에 대해 제대로 이해하기 위해서는 대기열, 대화 상자, 대화 그룹 및 활성화에 대한 기본 개념을 숙지해야 합니다. 본 섹션에서는 이들 개념을 간단히 설명합니다.

대기열

Service Broker는 메시지 발신자와 메시지 수신자 간에 느슨한 연결을 제공하는 대기열을 이용합니다. Service Broker가 메시지가 대상에 도달하도록 보장하기 때문에 발신자는 SEND 명령어를 이용하여 대기열에 메시지를 배치시킨 다음 응용 프로그램을 계속 실행시킬 수 있습니다.
대기열은 뛰어난 스케줄링 유연성을 제공합니다. 예를 들어 발신자는 여러 수신자들이 병렬로 처리할 수 있도록 여러 메시지를 전송할 수 있습니다. 수신자들은 메시지가 전송될 때까지 메시지를 처리하지 못할 수도 있지만 수신 메시지가 대기열에 배치되기 때문에 수신자들은 자체 속도로 메시지를 처리할 수 있고 발신자는 수신자들이 메시지 처리를 완료할 때까지 기다릴 필요가 없습니다.

대화 상자

Service Broker는 2개의 종단점 간 양방향 메시지 스트림인 대화 상자를 구현합니다. 대화 상자에 있는 모든 메시지들은 순서대로 나열되며 대화 상자 메시지들은 항상 보낸 순서대로 전달됩니다. 순서는 트랜잭션, 입력 스레드, 출력 스레드, 크래시 및 재시작 과정 전반에서 유지됩니다. 일부 메시지 시스템은 다수의 트랜잭션이 아닌 단일 트랜잭션으로 전송 또는 수신된 메시지에 대한 메시지 순서를 보장하여 이에 따라 Service Broker 대화 상자는 고유한 특성을 갖게 됩니다.
모든 메시지에는 관련 대화 상자를 식별하는 고유의 대화 핸들(conversation handle)이 포함되어 있습니다. 예를 들어 주문 입력 응용 프로그램은 선적 응용 프로그램, 재고 응용 프로그램 및 대금 청구 응용 프로그램 등과 함께 대화 상자가 동시에 열리도록 할 수 있습니다. 모든 응용 프로그램의 메시지는 고유의 대화 핸들을 보유하고 있기 때문에 어떤 응용 프로그램이 각각의 메시지를 전송했는지 쉽게 구분할 수 있습니다.

대화 그룹

Service Broker는 특정 작업에 이용되는 모든 대화 상자를 그룹핑하는 방법을 제공합니다. 이 방식은 대화 그룹을 이용합니다. 앞의 주문 입력 예제에서 특정 주문과 관련된 모든 대화 상자는 단일 대화 그룹으로 그룹화될 것입니다. 대화 그룹은 대화 그룹 식별자로서 구현되며 이는 대화 그룹을 구성하는 모든 대화 상자의 모든 메시지에 포함됩니다. 메시지가 대화 그룹의 모든 대화 상자에서 수신되면 대화 그룹은 수신 트랜잭션의 잠금 기능을 통해 잠깁니다. 트랜잭션이 진행되는 동안 잠금 상태에 있는 스레드만이 대화 그룹에 있는 모든 대화 상자의 메시지를 수신할 수 있습니다. 이는 확장성을 위해 다수의 스레드를 사용하더라도 모든 특정 주문이 한 번에 하나의 스레드에서만 처리되기 때문에 주문 입력 응용 프로그램에서 훨씬 손쉽게 쓰기 작업을 수행할 수 있습니다. 이는 곧 여러 스레드에서 단일 주문을 동시에 처리함에 따라 발생하는 문제에 대해 응용 프로그램이 복원력을 갖추도록 할 필요가 없다는 것을 의미합니다.
대화 그룹 식별자의 가장 일반적인 용도 중 하나는 특정 프로세스와 관련된 상태를 레이블링하는 것입니다. 시간이 지남에 따라 프로세스에

많은 메시지가 포함된다면 프로세스 전반에 걸쳐 실행되는 응용 프로그램의 인스턴스를 유지하는 것이 바람직하지 않을 수도 있습니다. 예를 들어 다음 메시지가 수신된 주문과 관련되어 있는 경우 메시지 간에 주문 처리와 관련된 모든 전역 상태가 데이터베이스에 저장되어 검색된다면 주문 입력 응용 프로그램은 보다 효과적으로 확장될 것입니다. 대화 그룹 식별자는 상태 테이블의 기본 키로 사용되어 모든 메시지와 관련된 상태를 신속하게 검색할 수 있습니다.

활성화

Service Broker의 활성화 기능을 이용해 특정 서비스를 대상으로 하는 메시지를 처리할 저장 프로시저를 지정합니다. 서비스를 위한 메시지가 도착하면 Service Broker는 메시지를 처리할 수 있는, 실행되는 저장 프로시저가 있는지 여부를 확인합니다. 실행 중인 메시지 처리 저장 프로시저가 없을 경우 Service Broker는 저장 프로시저를 실행합니다. 그러면, 저장 프로시저는 대기열이 비워지고 종료된 이후에 메시지를 처리합니다. 또한 Service Broker는 저장 프로시저가 처리할 수 있는 것보다 빨리 메시지가 도착했다고 판단하면 수신 메시지의 속도에 맞춰 충분히 실행될 때까지(또는 최대 구성 수에 도달할 때까지) 추가적인 저장 프로시저 인스턴스를 실행합니다. 이는 수신 메시지를 처리하는 데 항상 적절한 수의 자원을 활용할 수 있도록 보장합니다.

비동기식 대기열 메시징을 활용해야 하는 이유

대기열은 유연한 작업 예약을 지원하기 때문에 성능 및 확장성을 획기적으로 향상시킬 수 있습니다. 그 방법을 확인하려면 주문 입력 예제를 검토하십시오. 주문의 일부는 주문 헤더, ATP(Available To Promise) 및 주문 줄과 같이 주문이 완료되었다고 간주되기 전까지 처리되어야 합니다. 그러나 다른 부분들은 현실적으로 주문이 커밋되기 전에 처리될 필요가 없습니다(예: 대금 청구, 운송, 재고 관리 등). 주문의 "지연될 수 있는" 부분을 보장하면서도 비동기식으로 처리할 수 있다면 주문의 핵심 부분이 더욱 신속하게 처리될 수 있습니다.
또한 비동기식 메시징은 병렬 처리 성능을 향상시킬 수 있는 기회를 제공할 수 있습니다. 예를 들어 고객의 신용도와 주문한 품목의 재고를 확인해야 하는 경우 두 프로세스를 동시에 실행하여 전반적인 응답 시간을 향상시킬 수 있습니다.
또한 큐잉을 통해 시스템은 처리 작업을 보다 균등하게 분배하고 서버가 요구하는 최대 용량을 낮출 수 있습니다. 예를 들어 일반적으로 수신되는 주문 속도는 다음과 같습니다.

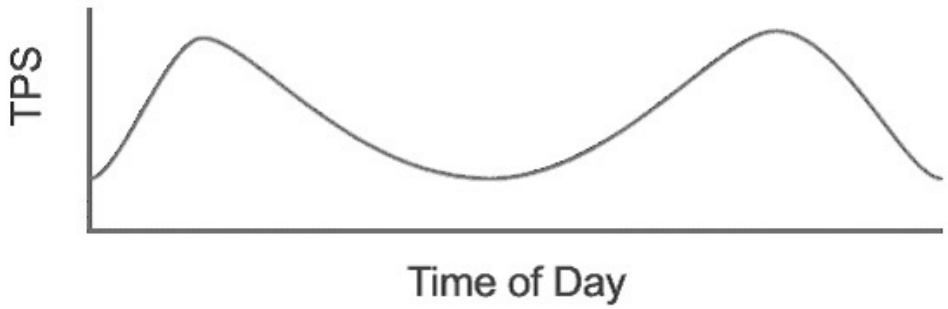

그림 1
첫 날과 마지막 날에 최대를 이룹니다. 각 주문이 작성될 때 운송 시스템에 입력된다면 운송 시스템의 로드는 다음과 비슷할 것입니다.

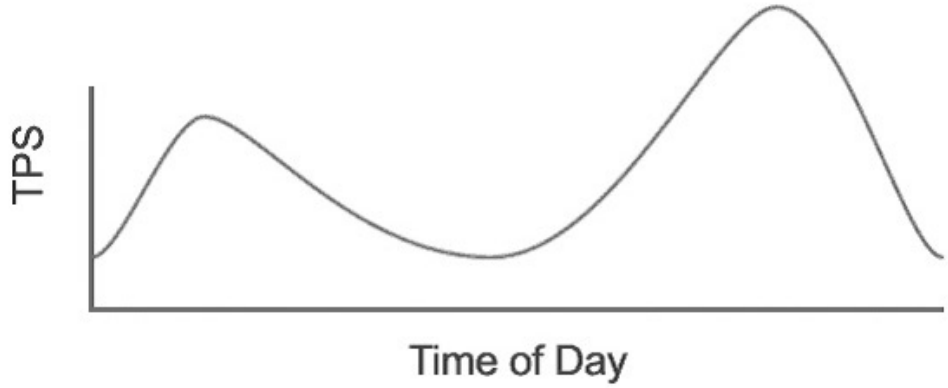

그림 2

발송을 위한 선적 관련 문서 작업을 수행해야 하는 오후에는 로드가 절정에 달하게 됩니다. 대기열을 통해 운송 시스템이 주문 입력 시스템과 연결된다면 일부 작업을 한가한 주문 시간대로 전환하여 폭주하는 작업량을 고르게 조정할 수 있습니다.

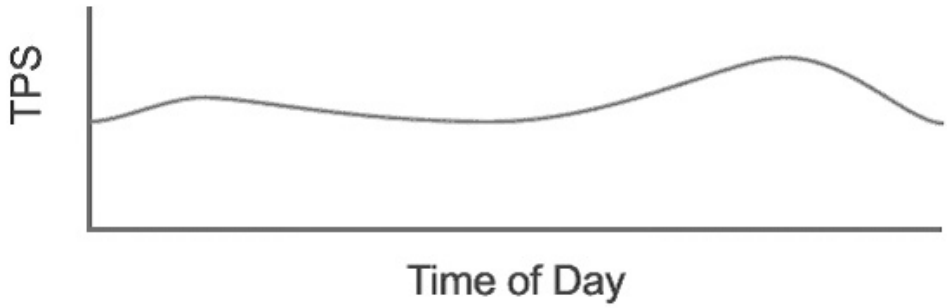

그림 3

트랜잭션 메시징을 활용해야 하는 이유

Service Broker는 메시지가 트랜잭션으로 전송 및 수신되는 것을 의미하는 트랜잭션 메시징을 지원합니다. 트랜잭션이 실패하면 송수신되는 메시지는 롤백되며 트랜잭션이 성공적으로 메시지를 처리하고 커밋할 때까지 적용되지 않을 것입니다.
적합한 곳에 송수신 메시지를 자유롭게 분산시킬 수 있고 트랜잭션이 커밋할 때까지 아무 것도 이루어지지 않는다는 점에서 트랜잭션 메시징을 통해 메시징 응용 프로그램을 더욱 간편하게 프로그래밍할 수 있습니다. 트랜잭션이 롤백되어 어느 것도 전송하지 않으면 수신된 메시지는 대기열로 되돌려져 다시 수신되고 처리될 것입니다.

Service Broker를 이용하여 어려운 메시징 문제를 해결하는 방법

메시징 응용 프로그램은 이미 오랫동안 이용되어 왔으며 이를 구축해야 하는 분명한 이유가 있습니다. 그렇다면 더 많은 메시징 응용 프로그램이 구축되지 않은 이유는 무엇일까요? 그것은 올바른 메시징 응용 프로그램을 구축하는 것이 어렵기 때문입니다. 그러나 SQL Server Service Broker는 메시지 주문, 조정, 멀티스레딩 및 수신자 관리 등을 비롯하여 가장 어려운 메시징 응용 프로그램 문제들을 해결할 수 있습니다.

메시지 주문

기존의 신뢰할 수 있는 메시징 응용 프로그램의 경우, 메시지가 비순차적으로 전달되기 쉽습니다. 예를 들어 응용 프로그램 A가 메시지 1, 2, 3을 전송합니다. 응용 프로그램 B는 1과 3을 수신 및 승인하지만 2에서 오류가 발생하여 응용 프로그램 A는 이를 다시 전송합니다. 그러나 이제 3을 수신한 이후에 2를 수신할 수 있게 되었습니다. 일반적으로 프로그래머들은 순서가 문제가 되지 않도록 응용 프로그램을 작성하거나 2가 도착할 때까지 3을 일시적으로 캐싱하여 메시지가 순조롭게 처리될 수 있도록 함으로써 이러한 문제를 처리했습니다. 반대로 Service Broker는 이를 투명하게 처리합니다. 따라서 대화 상자의 모든 메시지들은 메시지 시퀀스 간의 공백 없이 전송된 순서대로 수신됩니다.
관련된 문제로 중복 전달 문제를 들 수 있습니다. 앞의 예에서 응용 프로그램 B가 메시지 2를 수신했지만 응용 프로그램 A로 반환되는 승인 메시지가 유실되었다면 응용 프로그램 A는 2를 재전송하고 응용 프로그램 B는 메시지 2를 2번이나 수신하게 될 것입니다. Service Broker는 트랜잭션이 진행되는 도중에 전원이 나가더라도 메시지들이 결코 2번 전송되지 않도록 보장합니다.

조정

메시지는 일반적으로 독립형 엔터티이며 이는 메시지가 어떤 대화에서 형성되었는지 판단하기 어렵게 만듭니다. 예를 들어 인벤토리 업데이트가 필요한 인벤토리 서비스에 수천 개의 메시지를 전송할 수도 있습니다. 인벤토리 서비스는 이러한 메시지 중 일부에 거의 즉시 응답하고 다른 메시지에 응답하는 데에는 시간이 오래 걸릴 수 있으며 어느 응답 메시지가 인벤토리 요청에 해당되는지 판단하는 데 어려움을 겪게 될 수 있습니다.
이에 반해 Service Broker를 이용하면 대화 상자 핸들과 대화 그룹 식별자가 모든 메시지에 포함될 뿐만 아니라 각 응답이 해당되는 주문 및 요청을 쉽게 판단할 수 있습니다. (일부 메시징 시스템은 이러한 판단을 내릴 수 있는 상관 관계 ID를 보유하고 있지만 대화 상자에 필수적인 요소는 아닙니다.)

멀티스레딩

메시징 응용 프로그램에서 가장 어려움을 겪고 있는 문제 중 하나는 멀티스레드 판독기(multithreaded reader)가 제대로 작동되도록 하는 것입니다. 여러 스레드가 동일한 대기열에서 수신될 경우 메시지가 순서에 맞게 수신되고 있더라도 순서가 뒤바뀌어 처리될 가능성이 항상 있습니다. 예를 들어 주문 헤더를 포함하고 있는 메시지가 스레드 A를 통해 수신되고 주문 줄을 포함하고 있는 메시지가 스레드 B를 통해 나중에 수신된 경우, 주문 줄 트랜잭션이 먼저 커밋을 시도하지만 아직 주문이 없기 때문에 참조 제약 조건을 적용하는 데 실패하게 될 것입니다. 주문 줄 메시지가 주문 헤더가 나타날 때까지 롤백을 수행하더라도 이는 여전히 리소스를 낭비하게 됩니다.
Service Broker는 메시지를 읽을 때 대화 그룹을 잠금 상태로 설정하여 트랜잭션이 커밋할 때까지 다른 어떤 스레드도 관련 메시지를 수신할 수 없도록 함으로써 멀티스레딩 문제를 해결합니다. Service Broker를 이용하면 멀티스레드 판독기를 간편하고 안정적으로 작동할 수 있습니다.

수신자 관리

여러 안정적인 메시징 시스템에서는 대기열에서 메시지를 수신하는 응용 프로그램이 메시지가 수신되기 전에 실행되어야 합니다. 대부분의 경우 사용자는 각 대기열에서 실행되어야 하는 응용 프로그램 인스턴스나 스레드의 수를 결정해야 합니다. 이 수치는 고정되어 있는 반면, 메시지 도착률이 서로 다르다면 지나치게 많거나 너무 적은 대기열 판독기가 대부분의 시간 동안 실행됩니다.
Service Broker는 메시지가 도착하면 필요에 따라 대기열 판독기를 활성화하여 수신자 관리 문제를 해결합니다. 또한 판독기에 장애가 발생하거나 시스템이 재부팅되면 판독기는 자동으로 대기열의 메시지를 읽기 시작합니다. Service Broker는 일반적으로 중간 계층 트랜잭션 프로세싱 모니터를 통해 처리되는 동일한 종류의 많은 응용 프로그램 관리 작업을 수행합니다.

데이터베이스 내에 메시징을 구축하는 이유

Service Broker가 데이터베이스 엔진의 일부로 포함되는 이유는 무엇인가? 메시지를 저장하기 위해 데이터베이스를 사용하는 외부 응용 프로그램이라면 제대로 작동되지 않을 수도 있는가? 데이터베이스가 Service Broker를 위치시킬 수 있는 가장 적합한 장소로 꼽히는 데는 몇 가지 이유가 있습니다.

대화 그룹 잠금

Service Broker는 대화 그룹을 잠그는 방식으로 여러 대기열 판독기를 실행할 수 있지만 일반적인 데이터베이스 명령을 통해 대화 그룹을 잠그는 경우 효율적으로 실행한다는 것은 거의 불가능합니다. Service Broker는 새로운 유형의 데이터베이스 잠금을 적절히 활용하며 Service Broker 명령만이 이러한 잠금 유형을 파악할 수 있습니다.

원격 트랜잭션 수신 처리

일부 메시징 시스템들은 트랜잭션 메시징이 대기열과 동일한 서버에서 실행 중인 응용 프로그램을 수신하도록 제한합니다. 이에 반해 Service Broker는 데이터베이스에 연결될 수 있는 모든 서버에서 전송된 원격 트랜잭션을 수신할 수 있도록 지원합니다.

일원화된 관리

트랜잭션 메시징 시스템의 문제점 중 하나는 메시지가 데이터와 다른 위치에 저장되면 메시지 저장소와 데이터베이스 중 한쪽이 백업을 통해 복원되는 경우 이를 동기화할 수 없다는 것입니다. Service Broker의 메시지 및 응용 프로그램 데이터를 위한 단일 데이터베이스를 이용하면 이러한 문제가 거의 발생하지 않습니다. 데이터, 메시지 및 응용 프로그램 논리가 모두 데이터베이스 내에 있다면 단일 백업 대상, 단일 보안 기능 설치 지점 및 단일 관리 대상을 유지하게 됩니다.

직접 대기열 송수신

Service Broker는 데이터베이스 엔진에 통합되어 있기 때문에 동일한 SQL Server 인스턴스 내 데이터베이스의 다른 대기열로 보내진 메시지가 수신 대기열에 직접 배치됨으로써 전송 대기열을 우회하고 성능을 획기적으로 향상시킬 수 있습니다.

공통의 언어 지원

응용 프로그램의 메시징 부분과 응용 프로그램의 데이터 부분은 Service Broker 응용 프로그램과 동일한 언어와 도구를 사용합니다. 이는 개발자에게 친숙한 Microsoft ADO(ActiveX(Data Objects)와 메시지 기반 프로그래밍을 위한 기타 데이터베이스 프로그래밍 기법을 활용합니다. SQL Server 2005에서 제공하는 CLR(Common Language Runtime) 저장 프로시저를 이용하면 Service Broker 서비스를 구현하는 저장 프로시저를 다양한 언어로 작성할 수 있으며 Service Broker가 지원하는 일원화된 관리의 이점을 활용할 수 있습니다.

단일 연결 실행 기능

Service Broker 명령은 해당 응용 프로그램에서 사용되는 다른 Transact-SQL과 마찬가지로 동일한 데이터베이스 연결을 통해 실행될 수 있습니다. 이는 메시징 시스템이 데이터베이스 외부에 있었다면 메시징 트랜잭션이 분산 트랜잭션이 되어야 하지만 단일 연결 방식을 이용함으로써 메시징 트랜잭션이 분산 트랜잭션이 될 필요가 없다는 것을 의미합니다.

보안 강화

Service Broker 메시지는 데이터베이스를 통해 내부적으로 처리되기 때문에 데이터베이스 개체에 대한 메시지 발신자의 액세스 권한을 손쉽게 확인할 수 있습니다. 메시지 시스템이 외부 프로세스에 있다면 메시지를 전송하는 각 사용자들을 위해 별도의 데이터베이스 연결이 필요했을 것입니다. 데이터베이스 및 메시징 시스템에 대한 동일한 ID를 보유하고 있기 때문에 더욱 쉽고 정확하게 보안을 설정할 수 있습니다.

데이터베이스 전반의 전송 대기열 확인

Service Broker는 데이터베이스 인스턴스의 컨텍스트에서 실행되기 때문에 인스턴스의 모든 데이터베이스에서 전송될 준비가 된 모든 메시지의 집계 뷰를 유지할 수 있습니다. 이 기능을 통해 모든 데이터베이스는 백업 및 관리를 위한 고유 전송 대기열을 유지하고 인스턴스 내 모든 데이터베이스 전반에서 리소스 활용의 공평성을 유지할 수 있습니다. 외부 메시징 매니저를 이용하는 경우, 이를 실행하는 것이 불가능하지는 않지만 매우 어렵습니다.

결론

Service Broker가 제공하는 고유의 기능과 긴밀한 데이터베이스 통합으로 Service Broker는 느슨하게 결합된 새로운 차원의 데이터베이스 응용 프로그램 서비스를 구축할 수 있도록 지원하는 이상적인 플랫폼으로 자리매김하고 있습니다. Service Broker는 데이터베이스 응용 프로그램에 대해 비동기식 대기열 메시징을 제공할 뿐만 아니라 안정적인 메시징을 실현하는 최첨단 기능을 대폭 강화했습니다.

문서 번호. SQL-200509-10

Microsoft®

한국마이크로소프트(유)

서울특별시 강남구 대치동 892번지 포스코센터 ▌전화 : 080-985-2000 ▌인터넷 : www.microsoft.com/korea