UNIT 16

아키텍처

DB2의 아키텍처, 프로세스 모델, 메모리 모델에 대해 소개합니다. 무공유 아키텍처를 지원하므로 다중 서버에 다중 데이터베이스 파티션을 구축하면, 성능과 확장성이 좋은 병렬 처리 환경의 데이터베이스를 운영할 수 있습니다.

DB2 9.7 운영자 가이드

Administrator Edition

- 아키텍처 개요
- 단일 데이터베이스 파티션 아키텍처
- 다중 데이터베이스 파티션 아키텍처
- 데이터베이스 시스템
- 단일 데이터베이스 파티션의 프로세스 모델
- 다중 데이터베이스 파티션의 프로세스 모델
- 인스턴스 수준의 프로세스
- 데이터베이스 수준의 프로세스
- 응용프로그램 수준의 프로세스
- 메모리 모델
- 인스턴스 공유 메모리
- 데이터베이스 공유 메모리
- 응용프로그램 공유 메모리
- 응용프로그램 개별 메모리
- 스레드 모니터링
- 메모리 사용량 모니터링

TOPIC 1601 아키텍처 개요

Point

DB2는 다중 스레드 기반의 안정적이고 고성능을 추구하는 아키텍처에 기반하고 있습니다.

Tip

9.1까지 프로세스 기반의 모델이며, 9.5이후에는 스레드기반의 프로세스 모델로 변경되었습니다. 이는 자원의 효율적 사용과 동시에 성능을 보다 향상시킵니다.

Tip

64비트를 지원하므로 대용량의 버퍼 풀을 생성할 수 있고, 정렬 작업을 원활하게 수행할 수 있습니다.

Tip

*프리페처*는 디스크에서 데이터를 검색하고 응용프로그램에 데이터가 필요하기 전에 버퍼 풀로 이동합니다.

Tip

*페이지 클리너*는 버퍼 풀에서 디스크로 다시 데이터를 이동합니다. 페이지 클리너는 응용프로그램 에이전트와 관계가 없는 백그라운드 EDU입니다.

Tip

데이터베이스 서버에서 접속을 요청하는 응용프로그램을 지역 응용프로그램이라고 하고, 클라이언트에서 요청하는 응용프로그램을 원격 응용프로그램이라고 합니다.

1 DB2 9.5이전 DB2프로세스 모델은 아래와 같습니다.

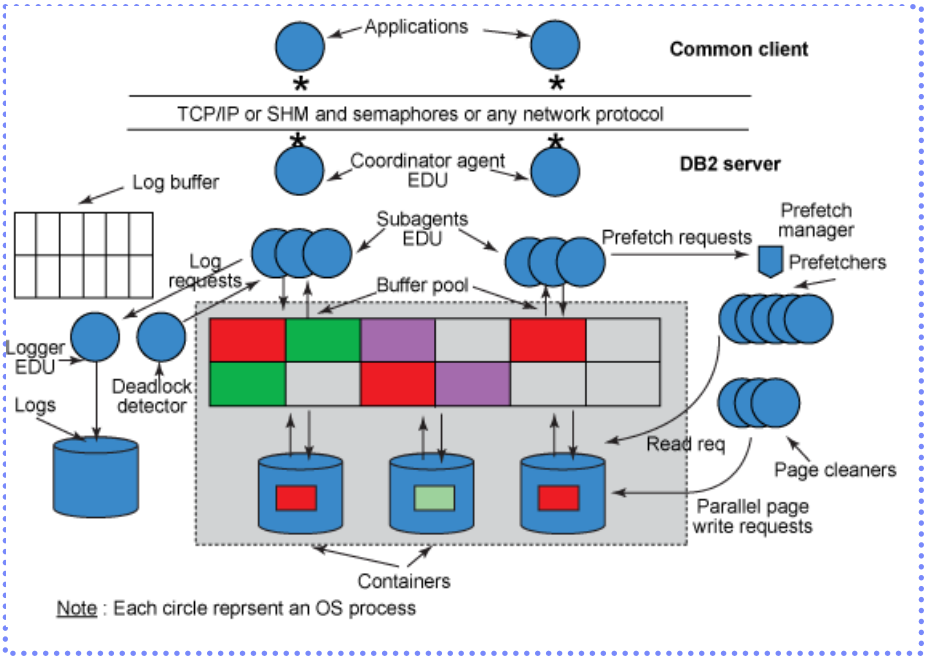

Figure 1601A ••• DB2 9.5 이전 프로세스 모델

2 DB2 9.5부터 다중 스레드 기반의 모델로 변경되었습니다.

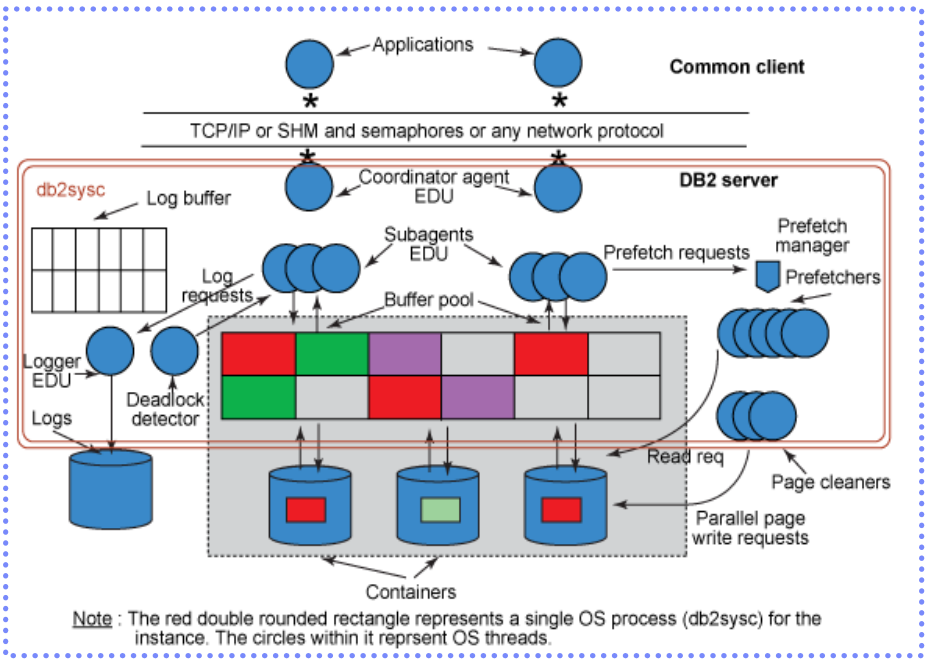

Figure 1601B ••• DB2 9.5 이후 새로운 프로세스 모델

TOPIC 16 02 단일 데이터베이스 파티션의 아키텍처

Point

단일 데이터베이스 파티션 환경에서 접속을 요청한 응용프로그램은 에이전트 프로세스를 통하여 엔진에게 SQL문의 처리를 요청할 수 있습니다. 파티션내 병렬 기능을 이용하면, SQL문을 병렬로 처리할 수 있습니다.

Tip

SMP 머신에서 파티션내 병렬 기능을 이용하면, 여러 개의 Subagent를 이용하여 쿼리를 병렬로 처리할 수 있습니다.

Tip

에이전트 프로세스의 생성과 제거로 인한 오버헤드를 줄이기 위해 에이전트 프로세스를 위한 POOL 을 운영할 수 있습니다.

Tip

데이터가 여러 디스크에 걸쳐 스트라이핑이 되어있다면, 여러 개의 프리페처를 지정하여 여러 디스크를 동시에 액세스할 수 있도록 합니다.

1 단일 데이터베이스 파티션으로 구성된 인스턴스의 아키텍처는 다음과 같습니다.

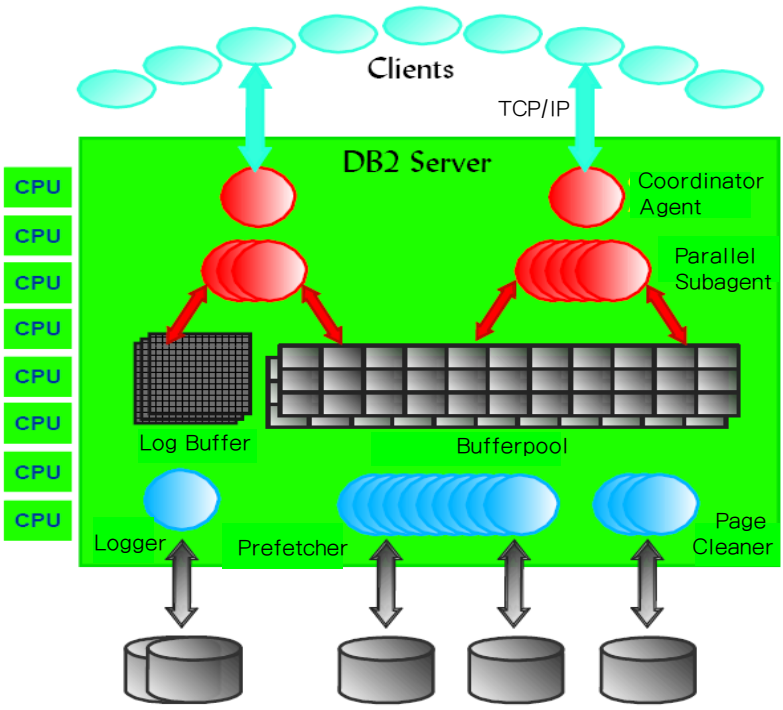

Figure 1602A ••• 단일 데이터베이스 파티션의 아키텍처

2 DB2 엔진은 고유한 기능을 담당하는 여러 개의 EDU(Engine Dispatchable Unit)로 구성되어 있으며, 각 EDU는 UNIX에서는 스레드로 구현됩니다.

3 응용프로그램이 데이터베이스에 접속을 요청하면, 전용 Coordinator Agent 프로세스가 생성되어 응용프로그램을 대신하여 데이터베이스에게 필요한 SQL문의 실행을 요청하는 역할을 하게 됩니다. 클라이언트와 서버 사이의 통신에는 TCP/IP, APPC, Netbios 등의 프로토콜이 지원됩니다.

4 에이전트 프로세스가 필요한 데이터를 가져오기 위하여 프리페치 큐를 통해 비동기 방식의 읽기 요청을 하면, 프리페처 프로세스는 big-block I/O를 통해 요청된 페이지들을 버퍼풀로 미리 가져오므로 에이전트 프로세스는 디스크 I/O 대기 시간을 줄일 수 있습니다.

5 페이지 클리너는 백그라운드 EDU로 특정 상황이 될 때마다 버퍼풀의 Dirty Page를 디스크로 미리 반영하여 버퍼풀의 가용 공간을 확보함으로써 버퍼풀을 사용하는 에이전트 프로세스가 대기하는 일이 없도록 합니다.

6 변경된 데이터 페이지와 로그 레코드는 성능을 위해 디스크로 즉시 반영되지 않습니다. 변경된 데이터를 디스크에 반영할 때는 반드시 로그 파일에 먼저 기록됩니다. 이러한 방식을 Write Ahead Logging이라고 합니다.

TOPIC 16 03 다중 데이터베이스 파티션의 아키텍처

Point

다중 데이터베이스 파티션 환경에서 접속을 요청한 응용프로그램은 코디네이터 에이전트와 파티션별 서브에이전트를 통하여 엔진에게 SQL문의 처리를 요청합니다. 파티션간 병렬 처리가 적용되므로 SQL문을 병렬로 처리할 수 있습니다.

Tip

다중 데이터베이스 파티션 환경에서는 파티션간 병렬 처리가 자동적으로 적용되며, 파티션내 병렬 처리 기능과 함께 사용할 수도 있습니다.

1 다중 데이터베이스 파티션으로 구성된 인스턴스의 아키텍처는 다음과 같습니다.

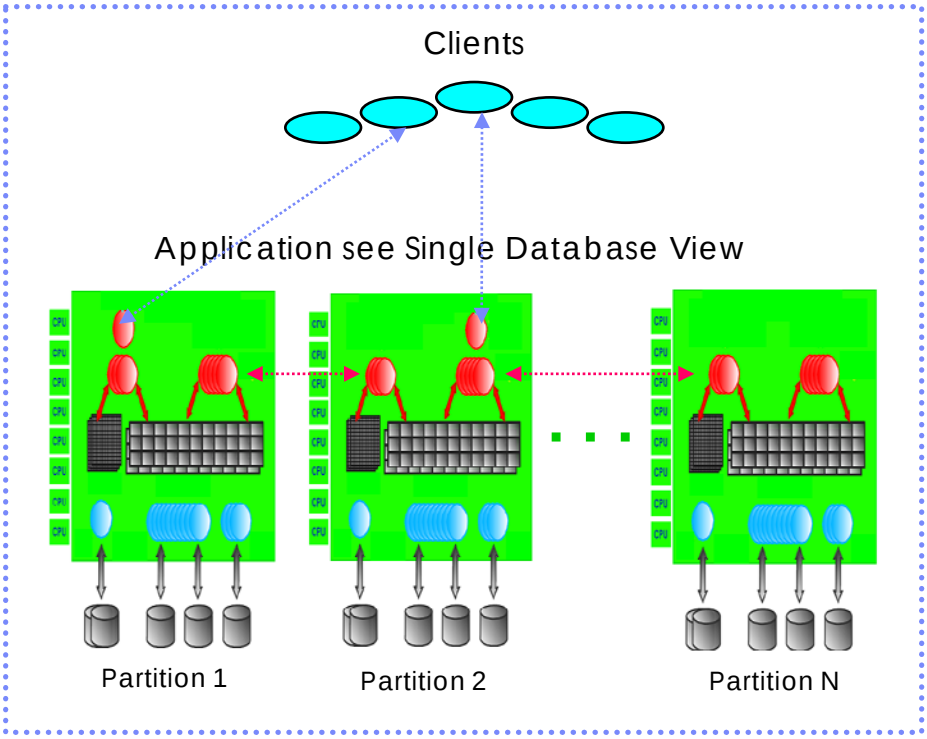

Figure 1603A ••• 다중 데이터베이스 파티션의 아키텍처

2 DB2 UDB의 Data Partitioning Feature는 다중 데이터베이스 파티션 기능을 이용하여 병렬 데이터베이스를 구축합니다. 다중 데이터베이스 파티션에 생성된 데이터베이스는 사용자에게는 한 개의 논리적인 데이터베이스로 인식되지만, 각 파티션에 물리적으로 데이터베이스를 생성합니다. 파티션별로 생성된 데이터베이스는 일반적인 단일 데이터베이스와 동일하게 독립적인 자원을 사용할 수 있으므로 각 파티션은 버퍼풀, 잠금 관리, 디스크 등을 독립적으로 운영하게 됩니다.

Tip

한 개의 서버에 기본적으로 4개의 논리적 파티션이 생성될 수 있으며, 필요시 4개 이상도 생성할 수 있습니다.

3 데이터베이스 파티션은 동일한 머신에 생성되는 논리적 파티션과 다른 머신에 생성되는 물리적 파티션으로 구분됩니다. 동일한 머신에 존재하는 데이터베이스 파티션들은 공유 메모리를 이용하여 통신하고, 다른 머신에 존재하는 데이터베이스 파티션끼리는 고속의 네트웍을 통해서 필요한 부분만 통신합니다.

Tip

SQL문을 처리하는 각 Subagent들의 실행은 모든 파티션을 통하여 병렬로 처리됩니다. DB2가 제공하는 여러 가지 유틸리티들도 모두 병렬로 처리됩니다.

4 응용프로그램이 접속을 요청하면, 접속한 데이터베이스 파티션에 Coordinator Agent 프로세스가 생성되고, 다른 파티션에는 Subagent 프로세스들이 생성됩니다. SQL문은 Global optimizer에 의한 최적화된 후에 각 Subagent들에게 전송됩니다. Subagent가 해당 파티션의 데이터베이스에 대해서만 SQL문의 요청을 처리하여 결과를 Coordinator Agent 프로세스에게 반환하면, Coordinator Agent는 응용프로그램에게 최종 결과를 반환합니다.

5 MPP 머신으로 구축한 DB2의 다중 데이터베이스 파티션 환경은 완전한 Shared Nothing Architecture를 제공하므로 확장성이 좋습니다.

TOPIC 1604 데이터베이스 시스템

Point

DB2 UDB 데이터베이스 시스템은 엔진, 인스턴스, 데이터베이스로 구성됩니다. DB2 엔진은 서버 머신에 설치된 실제적인 제품 모듈입니다. 인스턴스는 DB2 엔진을 사용하기 위한 논리적인 환경이며, 데이터베이스는 실제적인 데이터를 저장합니다.

1 DB2 UDB 데이터베이스 시스템은 엔진, 인스턴스, 데이터베이스로 구성됩니다. 엔진, 인스턴스, 데이터베이스, 테이블스페이스 컨테이너 등은 물리적으로 분리되어 있으므로 제품의 재설치, 인스턴스의 재생성시에도 데이터베이스와 관련된 물리적인 파일은 보존되고, 필요시에 다시 사용할 수 있습니다.

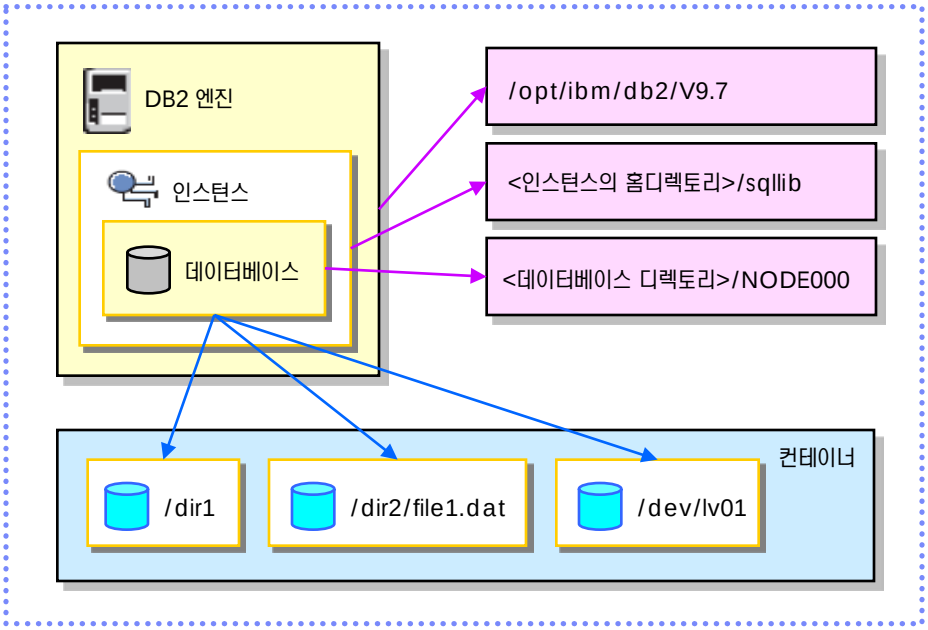

Figure 1604A ··· 데이터베이스 시스템과 대응되는 디렉토리

Tip

V9부터 설치 경로는 임의로 바꿀 수 있습니다. Default경로는 /opt/ibm/db2/V9.7입니다.

2 DB2 UDB 제품 모듈은 AIX 머신에서는 default 경로인 /opt/ibm/db2/V9.7 디렉토리에 설치되고, Windows 머신에서는 원하는 디렉토리에 설치됩니다. 한 개의 서버 머신에서 서로 다른 디렉토리에 각각의 다른 버전의 제품의 설치가 가능합니다.

Tip

인스턴스를 생성할 때, Windows 머신에서는 사용자 계정이 필요하지 않습니다. 또한, 설치시에 DB2 라는 이름으로 기본적인 인스턴스가 생성됩니다.

3 한 개의 서버 머신에서 여러 개의 인스턴스를 생성할 수 있습니다. AIX 머신에서는 인스턴스마다 사용자 계정이 필요합니다.

4 root 사용자가 db2icrt 명령어를 이용하여 인스턴스를 생성하면, 인스턴스 사용자 계정의 홈 디렉토리에 sqllib 라는 서브디렉토리가 생성됩니다. sqllib에 생성된 일부 디렉토리와 파일은 제품이 설치된 /opt/ibm/db2/v9.7 디렉토리의 일부 서브디렉토리와 심볼릭 링크(symbolic link)로 연결되거나 복사됩니다.

5 인스턴스 사용자가 db2start 명령어를 이용하여 인스턴스를 기동시키면, sqllib에 존재하는 파일들을 이용하여 인스턴스와 관련된 여러 개의 엔진 프로세스가 생성되고, 인스턴스 전역 메모리가 할당됩니다.

Tip

다중 파티션 환경인 경우에는 한 개의 논리적인 데이터베이스를 생성하면, 데이터베이스 파티션 개수만큼의 물리적인 데이터베이스가 생성됩니다.

6 인스턴스 사용자가 create database 명령어를 이용하여 데이터베이스를 생성하면, 특정 디렉토리에 데이터베이스 구성 파일을 비롯한 물리적인 디렉토리와 파일들이 생성됩니다.

7 데이터베이스에 접속한 후에 create tablespace 명령어를 이용하여 테이블스페이스와 컨테이너를 지정하면, 실제 데이터를 물리적으로 독립적인 공간에 저장할 수 있습니다.

TOPIC 16 05 단일 데이터베이스 파티션의 프로세스 모델

Point

단일 데이터베이스 파티션 환경에서 DB2 UDB의 프로세스는 인스턴스 수준, 데이터베이스 수준, 응용프로그램 수준으로 구분됩니다. 수준별 프로세스는 인스턴스 기동, 데이터베이스 활성화, 응용프로그램 접속시에 생성됩니다.

1 단일 데이터베이스 파티션으로 구성된 인스턴스에서는 다음과 같은 프로세스가 생성됩니다.

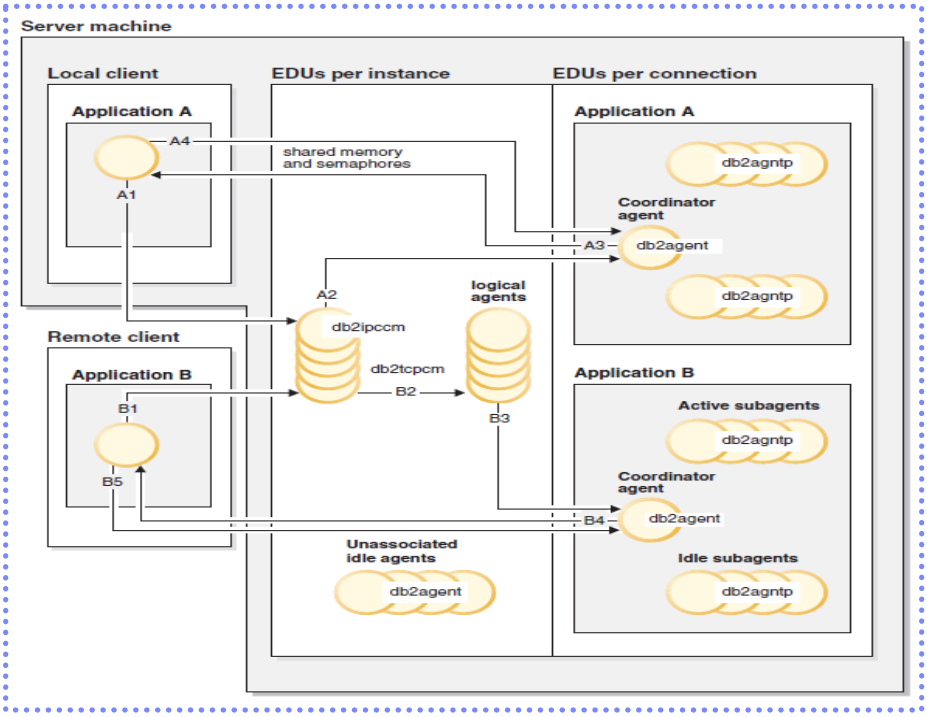

Figure 1605A ••• 단일 데이터베이스 파티션의 프로세스 모델

2 단일 데이터베이스 파티션 환경에서 DB2 프로세스는 인스턴스 레벨 에서만 보여진다.

수 준	설 명
인스턴스	인스턴스 수준의 프로세스는 db2start 명령을 실행하여 인스턴스를 기동할 때 생성되며 db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 가 있습니다. 그리고 동시에 db2sysc, db2tcpcm, db2pfchr, db2pclnr 등의 EDUs가 실행 됩니다.
데이터베이스	데이터베이스 수준에서는 EDUs로 보여지며 activate db 명령어에 의해 데이터베이스가 활성화될 때 생성되고, deactivate db 명령어에 의해 데이터베이스가 비활성화되면 제거됩니다. db2evmgi, db2fw0, db2wlmd, db2pfchr, db2pclnr, db2dlock, db2lfr, db2loggw, db2loggr, db2taskd, db2stmm 등의 EDUs가 있습니다.
응용프로그램	응용프로그램 수준에서도 EDUs로 보여지며 지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청하는 경우에 생성됩니다. db2agent, db2agntp 등의 EDUs가 있습니다.

3 데이터베이스 엔진과 관련된 프로세스들은 방화벽을 통해 외부의 응용프로그램 프로세스들과 다른 address space를 사용하도록 구성되어 있으므로, 데이터베이스 제어 블록 및 중요한 데이터베이스 파일과 분리되도록 설계되어 있습니다.

Tip

최초의 connect 요청은 데이터베이스를 간접적으로 활성화시키고, 최종의 connect reset 요청은 데이터베이스를 비활성화시킵니다.

Tip

데이터베이스를 액세스하는 응용프로그램은 반드시 데이터베이스에 접속하므로, connection과 응용프로그램은 동일한 의미로 사용됩니다.

Tip

특정 명령어나 유틸리티를 실행하기 위하여 attach 명령어를 이용하여 인스턴스에 접속하는 경우에도 응용프로그램 수준의 프로세스가 생성됩니다.

Tip

db2agentg 프로세스는 DB2 Connect 제품으로 호스트에 접속하거나, 특정한 인스턴스를 게이트웨이로하여 다른 인스턴스의 데이터베이스에 접속을 요청할 때 생성됩니다.

TOPIC 1606 다중 데이터베이스 파티션의 프로세스 모델

Point

다중 데이터베이스 파티션 환경에서 DB2 UDB의 프로세스는 인스턴스 수준, 카탈로그 파티션 수준, 데이터베이스 파티션 수준, 응용프로그램 수준으로 구분됩니다. 수준별 프로세스는 인스턴스 기동, 데이터베이스 활성화, 응용프로그램 접속시에 생성됩니다.

Tip

다중 데이터베이스 파티션을 구성하려면 DB2 UDB ESE 제품에 DPF 옵션이 필요합니다.

1 다중 데이터베이스 파티션으로 구성된 인스턴스에서는 다음과 같은 프로세스가 생성됩니다.

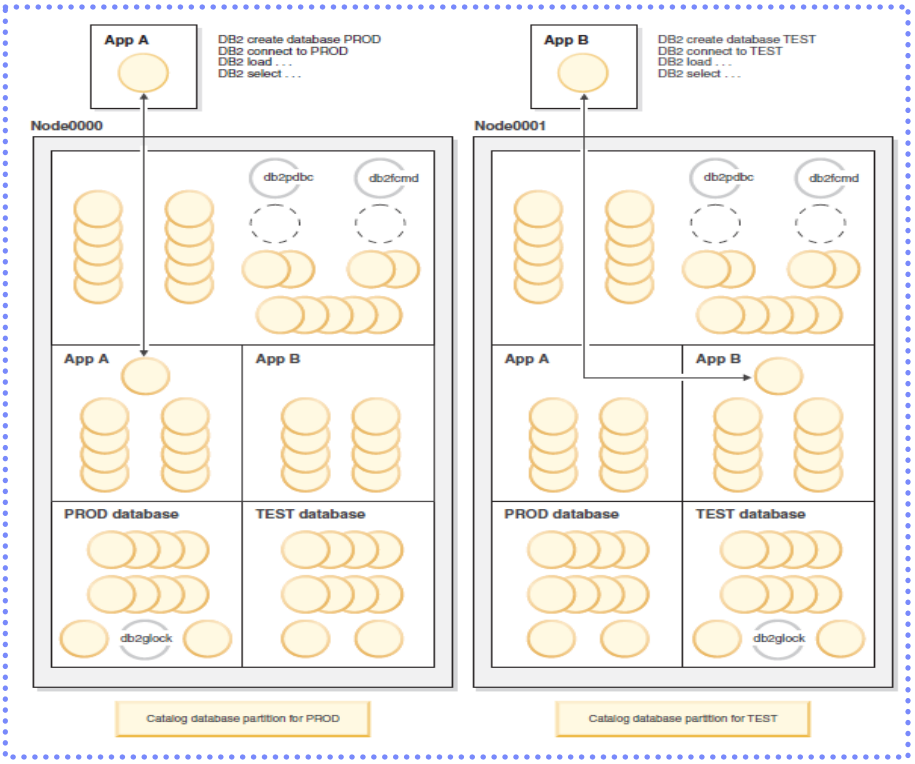

Figure 1606A ••• 다중 데이터베이스 파티션의 프로세스 모델

Tip

db2pdbc, db2panic 등은 다중 파티션 환경에서만 존재하는 프로세스입니다.

Tip

db2glock EDU는 카탈로그 파티션에만 존재합니다.

2 다중 데이터베이스 파티션 환경에서 DB2 프로세스는 인스턴스, 카탈로그 파티션, 데이터베이스 파티션, 응용프로그램의 4가지 수준으로 분리됩니다.

수 준	설 명
인스턴스	단일 데이터베이스 파티션 환경과 동일한 시점에 생성되고 제거됩니다. db2pdbc 와 db2fcmd 등의 EDUs가 추가됩니다.
카탈로그 파티션	단일 데이터베이스 파티션 환경과 동일한 시점에 생성되고 제거됩니다. db2glock EDUs가 추가됩니다.
데이터베이스 파티션	단일 데이터베이스 파티션 환경과 동일한 시점에 생성되고 제거됩니다.
응용프로그램	단일 데이터베이스 파티션 환경과 동일한 시점에 생성되고 제거됩니다.

TOPIC 1607 인스턴스 수준의 프로세스

Point

인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

Tip

※DB2 V9.5 부터는 프로세스 방식이 아닌 쓰레드 방식으로 운영되므로 유닉스의 $ ps -ef 명령으로는 실행중인 EDU(Engine Dispatchable Unit) 의 확인이 불가능합니다.

※DB2 V9.5 이후 제품에서는 $ db2pd -edus 로 확인합니다.

Tip

특정 인스턴스를 위한 엔진 프로세스들이 문제가 없는지를 확인하려면, ps 명령어로 db2sysc 프로세스를 확인하면 됩니다.

Tip

응용프로그램의 재접속이 많아지면, db2agent 프로세스의 생성이 빈번해지고, db2gds가 db2agent를 생성하고 삭제하기 위한 kernel CPU의 사용량이 증가합니다.

1 인스턴스 수준의 프로세스에는 db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등이 있습니다. 인스턴스 사용자가 db2start를 실행한 후에 ps 명령어를 이용하여 확인할 수 있습니다.

```
$ login <인스턴스 사용자>
$ db2start
$ ps –ef | grep -i <인스턴스명>
```

2 인스턴스가 기동될 때, 최초로 생성되는 기본 프로세스는 다음과 같습니다.

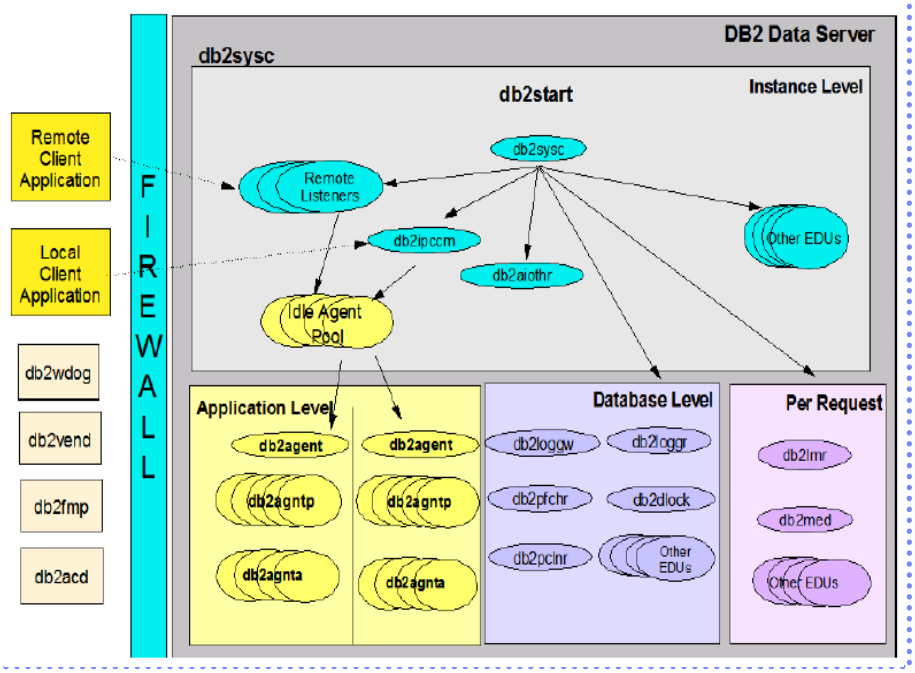

Figure 1607A DB2 프로세스 모델

프로세스 이름	설 명
db2sysc (Linux) db2syscs (Win)	DB2 9.5 이상의 시스템에서 db2start 명령과 동시에 발생되는 프로세스이다. 이 한 개의 프로세스로 모든 파티션의 thread를 처리 하도록 multi-thread로 되어 있으며, 여기에는 모든 Engine Dispatchable Units (EDUs) 이 thread로 구성되어 있다. 따라서 이 프로세스 없이는 데이터베이스가 실행될 수 없다.
db2acd	Health Monitor, 자동 유지보수 유틸리티 및 관리 태스크 스케줄을 관장하는 autonomic computing daemon 이다. db2hmon 에서 db2ace로 바뀐 것이다.
db2wdog	UNIX, LINUX 운영 체제에서 비정상 종료를 처리를 감시한다.
db2vend	EDUS에서 처리 할 수 없는 3rd party vendor 의 응용프로그램을 처리한다. Ex) userexit
db2fmp	stored procedures 또는 user defined functions(UDF) 와 같이 DB2의 외부에서 실행되는 코드를 처리한다. db2fmp 프로세스는 항상 별도의 프로세스이지만 실행하는 루틴의 유형에 따라 멀티스레드일 수 있다. DB2 8.7 이하에서의 db2udf 와 db2dari 프로세스가 db2fmp 프로세스로 대체되었다.

TOPIC 1607 인스턴스 수준의 프로세스

Point
인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

Tip
특정한 인스턴스에서 지원할 프로토콜의 유형은 db2set 명령어를 이용하여 DB2COMM 이라는 레지스터리 변수에 한 가지 이상을 지정할 수 있습니다.

Tip
DB2COMM 레지스터리 변수와 함께 각 프로토콜별로 필요한 구성 변수를 적절히 설정하지 않으면, db2start 명령어로 인스턴스를 기동할 때, 해당하는 통신 리스너 프로세스를 생성할 수 없습니다.

Tip
DISCOVERY 기능은 클라이언트 머신에서 GUI 도구인 구성 지원 프로그램을 이용하여 요청할 수 있으면, DAS가 시작되어 있어야 합니다.

3 지역 또는 원격에서 데이터베이스에 대한 접속을 요청하면, 클라이언트와 서버 머신 사이에 사용되는 통신 프로토콜의 유형에 따라 다음과 같은 통신 리스너 스레드가 db2agent 라는 EDU를 생성하여 응용프로그램과 엔진을 연결시켜 줍니다.

EDU 명	생성시점	설명
db2ipccm	db2start 실행시	DB2 IPC Communication Manager 입니다. 데이터베이스 서버 머신에서 지역적으로 접속을 요청하는 응용프로그램을 지원하는 통신 리스너 EDU입니다. 한 개의 지역 응용프로그램으로 데이터베이스의 접속을 요청받으면, db2agent EDU를 생성하여 해당 응용프로그램 프로세스와 생성한 db2agent EDU를 연결합니다. 연결이 완료되면, db2agent EDU가 지역 응용프로그램 프로세스와 엔진을 연결하는 역할을 하게 되고, db2ipccm은 새로운 접속 요청을 대기하게 됩니다.
db2tcpcm	db2start 실행시	DB2 TCP Communication Manager 입니다. TCP/IP 프로토콜을 이용하여 접속을 요청하는 원격 응용프로그램을 지원하는 통신 리스너 EDU입니다. db2ipccm과 동일한 방법으로 작동합니다.
db2snacm	db2start 실행시	DB2 SNA/APPC Communication Manager 입니다. SNA/APPC 프로토콜을 이용하여 접속을 요청하는 원격 응용프로그램을 지원하는 통신 리스너 EDU입니다. db2ipccm과 동일한 방법으로 작동합니다.
db2tcpdm	db2start 실행시	DB2 TCP Discovery Manager 입니다. 클라이언트 머신에서 TCP/IP 프로토콜을 이용하여 원격 데이터베이스 서버의 인스턴스를 자동적으로 발견해내는 기능인 DISCOVERY 요청을 처리하는 통신 리스너입니다.

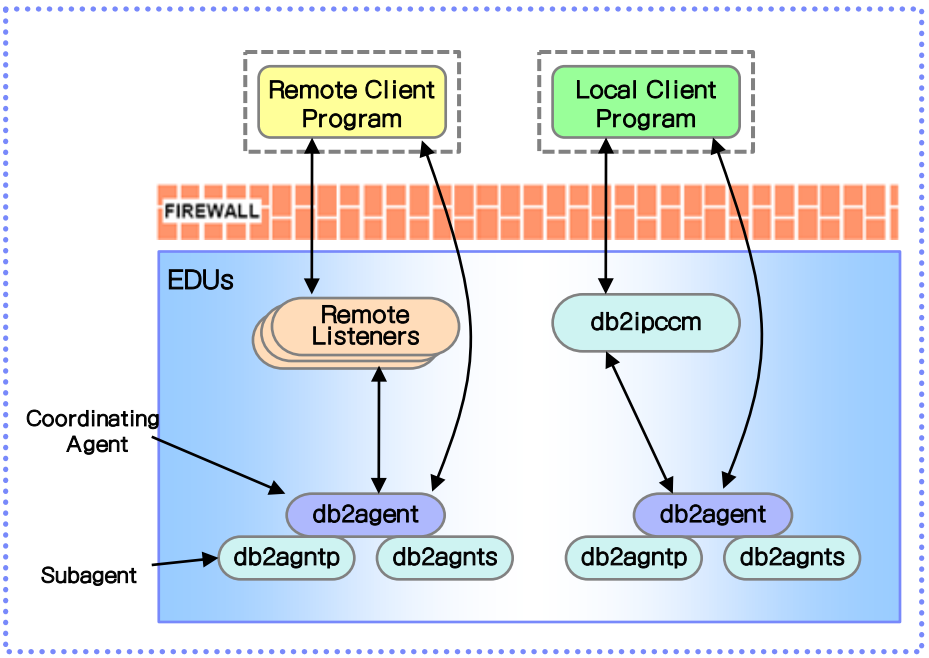

Figure 1607B ••• 통신 리스너와 db2agent EDU

TOPIC 1607 인스턴스 수준의 프로세스

Point

인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

Tip

UserExit 프로그램으로 로그를 아카이브하려면, LOGARCHMETH1 데이터베이스 구성 변수에 아카이브용 디렉토리명을 지정해도 됩니다.

Tip

UserExit 프로그램을 사용하지 않고, 아카이브 로깅을 설정하는 경우에는 LOGARCHMETH1 데이터베이스 구성 변수에 LOGRETAIN 이라고 지정해도 됩니다.

Tip

UserExit 프로그램을 사용하지 않고, 아카이브 로깅을 설정하는 경우에는 LOGRETAIN 데이터베이스 구성 변수에 ON 이라고 지정해도 됩니다.

4 아카이브 로깅만 사용되거나, UserExit 프로그램을 이용하여 아카이브 로깅을 하는 경우에는 다음과 같은 프로세스가 관련됩니다.

프로세스명	생성시점	설명
db2cart	db2start 실행시	아카이브 로깅에서 UserExit이 설정되어 있는 경우에 사용됩니다. 로그 파일을 아카이브해야 하는 시점을 결정하고, Disk 또는 Tape 등의 저장 매체로 로그 파일의 아카이브를 실제로 수행하는 user exit 프로그램을 호출합니다. UserExit 프로그램을 이용한 아카이브 로깅을 사용하려면 데이터베이스별로 설정합니다. 데이터베이스 구성 변수인 LOGRETAIN을 ON으로 설정하고 USEREXIT도 ON으로 설정합니다.
db2fmtlg	db2start 실행시	DB2 Format Log 입니다. UserExit 프로그램을 사용하지 않지만, 로그 파일을 현재의 로그 디렉토리에 아카이브하는 경우에 로그 파일을 미리 포맷하여 할당합니다. DB2 엔진 프로세스는 특정한 로그 파일에 대한 기록을 완료하고, 다른 로그 파일로 연속적으로 기록하는 경우에 새로운 로그 파일이 생성될 때까지 기다리지 않고 처리를 계속할 수 있도록 합니다. UserExit 프로그램을 이용하지 않고 현재의 로그 디렉토리에 아카이브만 하게 하려면, 데이터베이스 구성 변수인 LOGRETAIN을 ON으로 설정하고, USEREXIT을 OFF로 설정합니다.

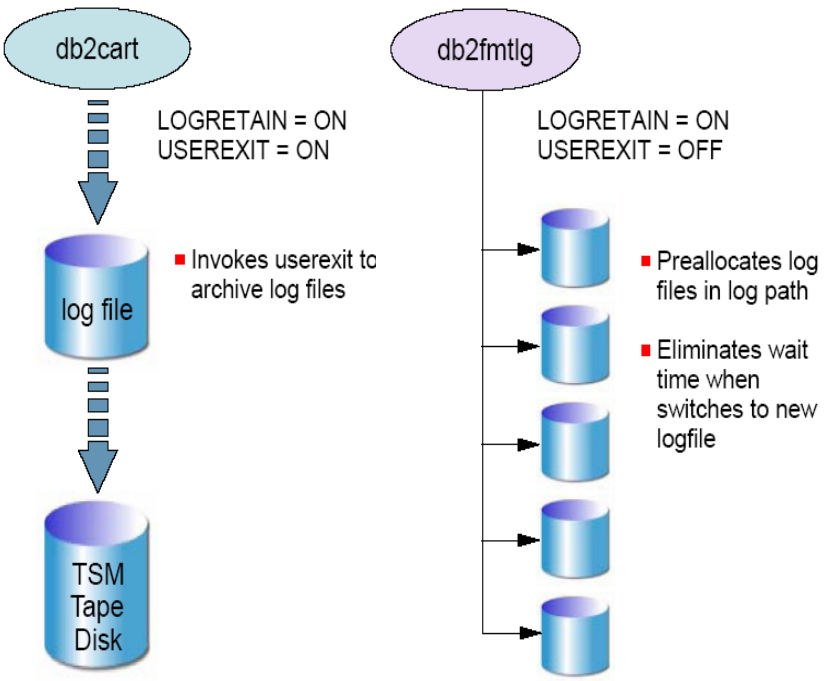

Figure 1607C ••• db2cart와 db2fmtlg 프로세스

TOPIC 1607 인스턴스 수준의 프로세스

Point

인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

Tip

기존 컨테이너의 크기만 늘리는 경우에는 db2rebal 프로세스가 호출되지 않습니다.

Tip

새로운 컨테이너를 추가하는 경우에도 BEGIN NEW STRIPE SETS 옵션을 이용하면, db2rebal 프로세스가 호출되지 않지만, 데이터의 균등한 분배를 위하여 권장하지 않습니다.

5 특정한 DB2 명령어 또는 유틸리티를 실행할 때 생성되는 프로세스는 다음과 같습니다.

프로세스명	생성시점	설명
db2chkau	db2audit 시작시	DB2 Check Audit 입니다. db2audit 유틸리티가 DB2 audit 로그 파일의 각 항목을 audit 버퍼에서 $instance_home/sqllib/security/db2audit.log 파일로 로깅할 때 사용됩니다.
db2govd	db2gov 시작시	DB2 Governor Daemon 입니다. db2gov 유틸리티가 governor configuration으로 지정한 규칙에 따라 데이터베이스에서 수행되는 응용프로그램을 모니터링하고, 지정한 임계값을 초과한 응용프로그램에 대한 조치를 취하기 위해 사용됩니다. db2govd 유틸리티는 데이터베이스에 대한 응용프로그램이 아니므로, 데이터베이스에 접속하는 것이 아니라, 인스턴스에 접속합니다.
db2rebal	케이블스페이스에 컨테이너 추가시	DB2 Rebalancer 입니다. 기존의 테이블스페이스에 컨테이너를 추가할 때 호출되어, 기존 데이터를 추가된 컨테이너로 균등하게 분산시키는 작업을 수행합니다.

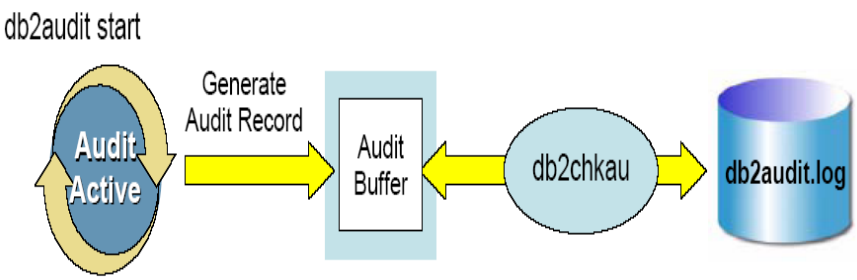

Figure 1607D ••• db2chkau 프로세스

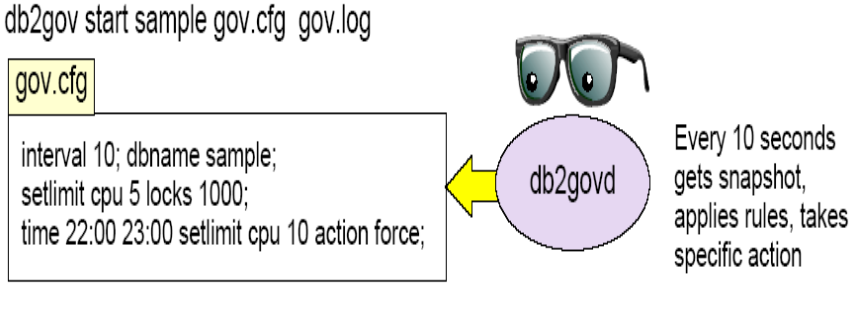

Figure 1607E ••• db2govd 프로세스

TOPIC 1607 인스턴스 수준의 프로세스

Point

인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

6 DPF를 사용하여 다중 데이터베이스 파티션 환경을 구성한 경우에만 생성되는 프로세스들은 다음과 같습니다.

프로세스명	생성시점	설명
db2fcmd	db2start 실행시	DB2 Fast Communication Manager Daemon 입니다. 파티션간의 통신에 사용됩니다.
db2glock	db2start 실행시	DB2 Global Deadlock Detector 입니다. 개별 파티션에 있는 db2dlock 프로세스로부터 수집된 정보를 모아서 데이터베이스 파티션간의 교착 상태가 발생했는지를 점검합니다. 카탈로그 파티션에 존재합니다.
db2panic	db2start 실행시	DB2 Panic 입니다. 특정한 데이터베이스 파티션에서 agent의 한계에 도달하게 되면 긴급 요청을 처리합니다.
db2pdbc	db2start 실행시	DB2 Parallel Database Controller 입니다. 원격 데이터베이스 파티션간의 병렬 요청을 처리합니다. 개별 데이터베이스 파티션마다 한 개씩 존재합니다.

Tip

trans1.subsection1은 trans2.subsection2에 의해 잠금 대기 상태에 있고, trans1.subsection2에게 메시지를 반환할 수 없으므로, trans1은 종료되지 못하고 있습니다.

Tip

trans2.subsection1은 trans1.subsection2에 의해 잠금 대기 상태에 있고, trans2.subsection2에게 메시지를 반환할 수 없으므로, trans2도 종료되지 못하고 있습니다.

7 global deadlock 의 작동 원리를 이해하기 위하여 다음과 같은 상황을 가정하도록 합니다. 두 개의 트랜잭션 Trans1과 Trans2가 있습니다. 두 트랜잭션은 각각 한 개의 SQL문에 대한 subsection1과 subsection2 를 두 개의 파티션에서 병렬로 실행시키려고 합니다.
trans1은 파티션2에서 subsection2의 작업을 먼저 끝내고, 파티션1에서 subsection1이 작업 결과 또는 메시지를 반환하기를 대기하고 있지만, subsection1은 파티션1에서 trans2의 subsection2가 lock을 걸고 있기 때문에 잠금 대기 상태에 있습니다.
trans2는 파티션1에서 subsection2의 작업을 먼저 끝내고, 파티션2에서 subsection1이 작업 결과 또는 메시지를 반환하기를 대기하고 있지만, subsection1은 파티션2에서 trans1의 subsection2가 lock을 걸고 있기 때문에 잠금 대기 상태에 있습니다.
다중 파티션 환경에서 서로 잠금을 대기하는 교착 상황이 발생하게 되면, db2glock 프로세스가 점검하여 한 쪽의 응용프로그램을 강제로 종료시킵니다.

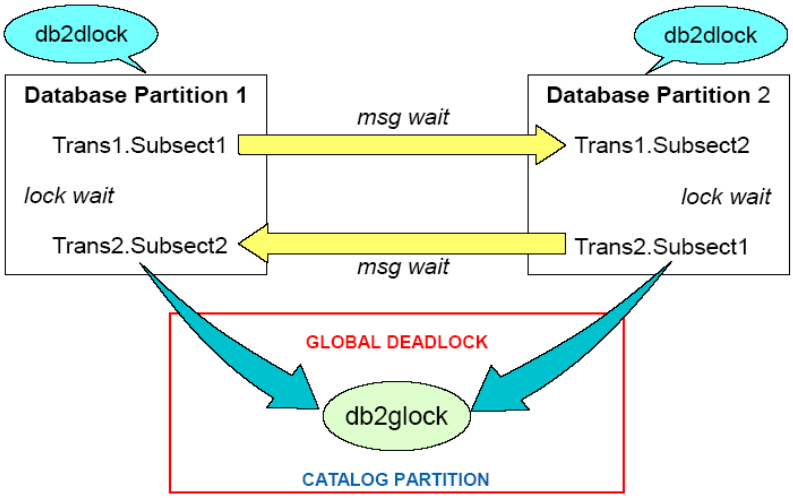

Figure 1607F ••• db2dlock과 db2glock 프로세스

TOPIC 1607 인스턴스 수준의 프로세스

Point

인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

Tip

db2fmcd 와 db2fmd 프로세스는 UNIX 시스템 에서만 존재합니다.

8 Fault Monitor 와 연관된 프로세스입니다.

프로세스명	생성시점	설명
db2fmcd	시스템 boot 시	DB2 Fault Monitor Coordinator Daemon 입니다. 각 데이터베이스 서버 머신에 한 개씩 존재하며, db2fmcd 프로세스에 이상이 없는지를 모니터링 합니다. 사용자가 kill 명령을 이용하여 강제로 db2fmd 프로세스를 제거하더라도, db2fmd 프로세스는 db2fmcd 프로세스에 의해 다시 생성됩니다.
db2fmd	db2start 실행시	DB2 Fault Monitor Daemon 입니다. DB2 인스턴스를 모니터링하는 데몬 프로세스로 DB2 인스턴스가 db2stop 명령어에 의하여 정상적으로 중지되는 것을 제외한 경우에 다시 엔진을 기동시키는 역할을 합니다. HA 환경을 구성하는 경우에는 db2fmcd 와 db2fmd 프로세스가 자동적으로 시작되지 않도록 해야 합니다.

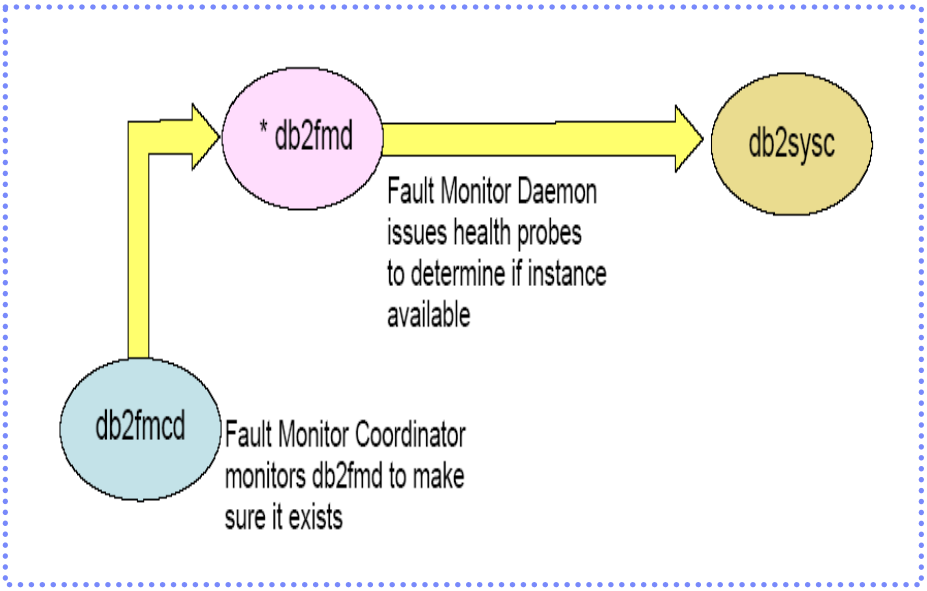

Figure 1607G db2fmcd과 db2fmd 프로세스

9 db2fmcd 프로세스는 기본적으로 시스템 부팅시에 자동적으로 시작되도록 /etc/inittab 파일에 등록되어 있습니다. 자동적으로 시작되는 것을 원하지 않는다면, 해당 명령행을 주석으로 처리하도록 합니다.

```
$ login root
$ cat /etc/inittab
...
#respawn:/opt/ibm/db2/V9.7/bin/db2fmcd
...
```

TOPIC 1607 인스턴스 수준의 프로세스

Point

인스턴스 수준의 프로세스는 db2start 명령으로 인스턴스를 기동할 때 생성됩니다. db2fmcd, db2wdog, db2sysc, db2ckpwd, db2acd, db2fmd 등의 프로세스가 있습니다.

Tip

원격 클라이언트의 응용프로그램에서 DB2 서버의 데이터베이스로 접속하려면, connect 문을 사용할 때 반드시 user와 using 옵션으로 사용자명과 암호를 제공해야 합니다.

Tip

Parallel Sysplex는 워크로드를 처리하기 위해 특정한 멀티시스템 하드웨어 구성 요소 및 소프트웨어 서비스로 서로 통신하고 협력하는 z/OS 또는 OS/390 시스템 세트입니다.

Tip

connection concentration 기능을 사용하려면, 인스턴스 구성 변수인 MAX_CONNECTIONS 의 값을 MAX_COORDAGENTS 보다 크게 설정합니다. 9.7에서는 기본값이 AUTOMATIC입니다.

10 OS와 관련된 작업을 수행하는 프로세스는 다음과 같습니다.

프로세스명	생성시점	설명
db2ckpw	db2start 실행시	DB2 Check Password 입니다. DB2 UDB는 OS의 인증 시스템을 이용하여 사용자에 대한 인증을 실행합니다. 원격 클라이언트 응용프로그램에서 제공된 사용자명과 암호가 OS에 등록된 사용자 정보와 일치하는지를 점검하여 접속의 허용 여부를 결정합니다. AIX에서 사용자 인증을 위해 /etc/security/passwd 파일을 사용합니다.
db2resyn	db2start 실행시	DB2 Resynchronization 입니다. 2단계 확약(2-phase commit) 방식을 사용하는 응용프로그램에 대한 동기화를 담당합니다. 단일 또는 다중 파티션 환경에서 모두 사용됩니다.
db2srvlst	db2start 실행시	DB2 Server List 입니다. OS/390과 같은 시스템에 대한 address list를 관리하는데 사용됩니다. Parallel Sysplex와 함께 사용됩니다.
db2syslog	db2start 실행시	DB2 System Logger 입니다. OS의 시스템 오류 로그 파일에 오류를 기록합니다.
db2disp	db2start 실행시	DB2 Dispatcher 입니다. connection concentration 기능을 사용하는 경우에 생성됩니다. 응용프로그램에 할당되어 있는 logical coordinating agent와 사용 가능한 coordinate agent를 연결시키는 역할을 합니다.

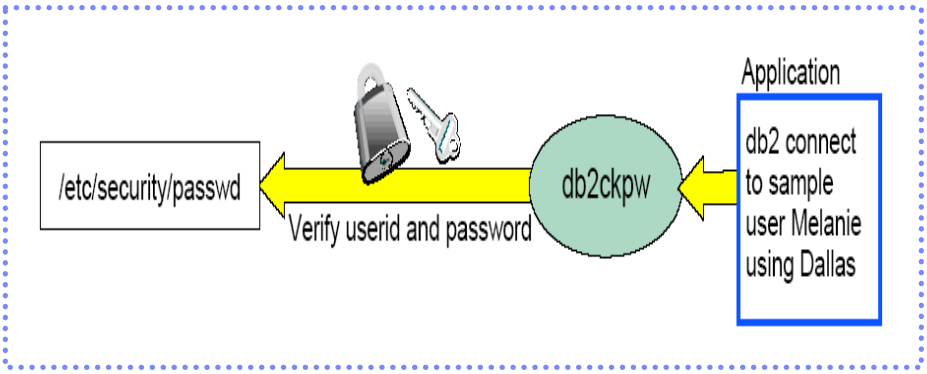

Figure 1607H ··· db2ckpw 프로세스

TOPIC 1608 데이터베이스 수준의 프로세스

Point

데이터베이스 수준의 프로세스는 ACTIVATE DB 명령어에 의해 데이터베이스가 활성화될 때 생성됩니다. db2pfchr를 비롯하여 db2pclnr, db2loggr, db2loggw, db2logts, db2dlock 등의 프로세스가 있습니다.

Tip
NUM_IOSERVER 데이터베이스 구성 변수의 기본값은 AUTOMATIC 입니다.

Tip
NUM_IOCLEANERS 데이터베이스 구성 변수의 기본값은 AUTOMATIC 입니다.

Tip
db2agent 프로세스가 버퍼풀의 페이지를 요구하는 시점 직전에 디스크로 변경 사항이 기록된 페이지를 good victim page 라고 하며, db2agent 프로세스는 버퍼풀 전체를 검색할 필요가 없게 됩니다. good victim page의 수가 기준값 이하로 떨어지면 페이지 클리너 프로세스를 호출합니다.

Tip
버퍼풀에서 그 값이 변경되었으나, 테이블스페이스의 컨테이너에 아직 반영되지 않은 값이 포함된 페이지를 Dirty Page 라고 합니다.

Tip
CHNGPGS_THRESH 데이터베이스 구성 변수의 기본값은 60%입니다. 갱신 작업이 많다면, CHNGPGS_THRESH 변수를 기준값 이하로 설정하여 버퍼풀에 가용 페이지를 충분히 확보하는 것이 유리합니다.

Tip
LSN은 로그 시퀀스 번호로 엔진이 트랜잭션을 식별하고 추적하게 합니다. MINBUFFLSN은 이미 커미트되었지만, 테이블스페이스 컨테이너로 반영되지 않은 가장 오래된 LSN을 표시합니다. 즉, 버퍼풀에서 가장 오래된 Dirty Page의 LSN을 의미합니다.

Tip
SOFTMAX 데이터베이스 구성 변수의 기본값은 100% 로 로그 파일 한 개의 크기만큼입니다.

1 db2pfchr 와 db2pclnr 프로세스는 버퍼풀과 테이블스페이스 컨테이너 사이의 비동기적인 I/O를 담당합니다.

프로세스명	생성시점	설명
db2pfchr	데이터베이스 활성화시	DB2 Bufferpool Prefetcher 입니다. 프리페처는 테이블과 인덱스의 데이터를 테이블스페이스 컨테이너인 디스크로부터 데이터베이스 버퍼풀로 비동기적인 방법으로 미리 읽어들이는 역할을 합니다. 각 응용프로그램을 대신하는 db2agent 프로세스가 db2pfchr 프로세스에게 프리페치 요청을 보내면, db2pfchr는 Big-block I/O 방식으로 데이터를 효율적으로 읽어들입니다. 데이터베이스 구성 변수인 NUM_IOSERVER 를 이용하여 데이터베이스별로 활동할 프리페처의 개수를 지정합니다.
db2pclnr	데이터베이스 활성화시	DB2 Bufferpool Page Cleaner 입니다. 데이터베이스 버퍼풀에 가용 공간을 확보하기 위해 버퍼풀로부터 비동기적인 방식으로 테이블스페이스 컨테이너에 변경된 데이터를 기록합니다. 특정한 페이지 클리너가 작업을 시작하면, 다른 페이지 클리너들도 동시에 작업을 수행합니다. 작업이 완료되면, 페이지 클리너 프로세스들은 sleep 상태로 대기하게 됩니다. 데이터베이스 구성 변수인 NUM_IOCLEANERS 를 이용하여 데이터베이스별로 활동할 페이지 클리너의 개수를 지정합니다.

2 db2pclnr 프로세스는 다음과 같은 경우에 활동하게 됩니다.

활동시점	설명
Dirty Page Threshold	버퍼풀의 Dirty Page 의 비율이 CHNGPGS_THRESH 데이터베이스 구성 변수의 값을 초과할 때 db2agent 프로세스가 호출됩니다.
LSN Gap Threshold	MINBUFFLSN과 현재LSN의 차이가 SOFTMAX 데이터베이스 구성 변수의 값을 초과할 때 log writer 프로세스가 호출합니다.

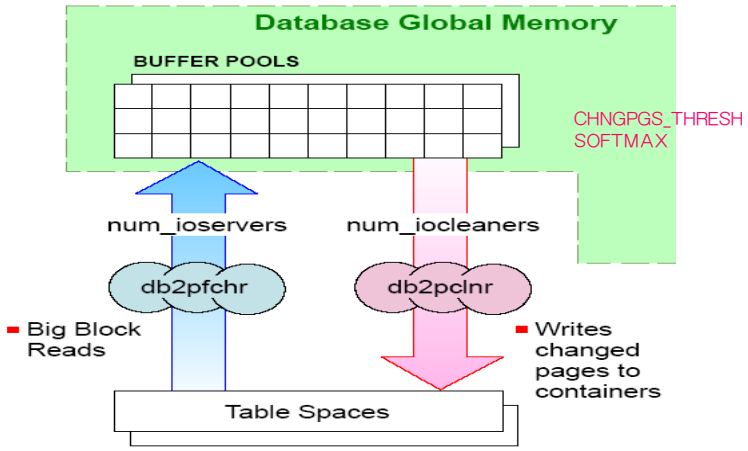

Figure 1608A ··· db2pfchr와 db2pclnr 프로세스

TOPIC 1608 데이터베이스 수준의 프로세스

Point

데이터베이스 수준의 프로세스는 ACTIVATE DB 명령어에 의해 데이터베이스가 활성화될 때 생성됩니다. db2pfchr를 비롯하여 db2pclnr, db2loggr, db2loggw, db2logts, db2dlock 등의 프로세스가 있습니다.

Tip

버퍼풀 또는 로그 버퍼에 있는 데이터가 해당되는 테이블스페이스의 컨테이너에 기록될 때에는 반드시 로그 파일에 먼저 기록이 되어야 합니다. 이것을 WAL (Write Ahead Logging) 프로토콜이라고 합니다.

3 데이터베이스 로그 파일의 I/O 와 연관된 프로세스입니다.

프로세스명	생성시점	설명
db2loggr	db2start 실행시	DB2 Database Log Reader 입니다. rollback 문의 요청에 의한 트랜잭션 롤백, restart db 명령어 요청에 의한 응급 복구, rollforward db 명령어에 의한 아카이브 로그 적용시에 로그 파일로부터 데이터베이스 로그 레코드를 읽어들이는 역할을 합니다.
db2loggw	db2start 실행시	DB2 Database Log Write 입니다. 로그 버퍼에 공간이 없는 경우에 로그 버퍼에 저장된 로그 레코드를 로그 파일로 기록하는 역할을 합니다. 버퍼풀의 Dirty Page가 디스크로 반영되거나 COMMIT 문이 요청되는 경우에도 작동됩니다.

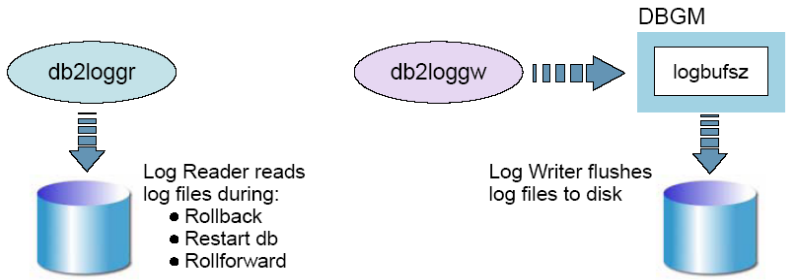

Figure 1608B ··· db2loggr 와 db2loggw 프로세스

Tip

db2logts는 레지스터리 변수인 DB2_COLLECT_TS_REC_INFO를 설정할 때 사용됩니다.

4 테이블스페이스의 복구와 연관된 프로세스입니다.

프로세스명	생성시점	설명
db2logts	db2start 실행시	DB2 Log Tablespace입니다. 테이블스페이스가 변경될 당시에 어떤 로그가 활성화 상태였는지에 대한 history 정보를 수집하는데 사용됩니다. 수집된 정보는 DB2TSCHG.HIS 파일에 저장되며, 테이블스페이스에 대한 롤포워드 복구시에 해당 테이블스페이스의 복구 작업에 불필요한 로그 파일은 적용하지 않으므로 복구의 성능을 향상시키는데 사용됩니다.

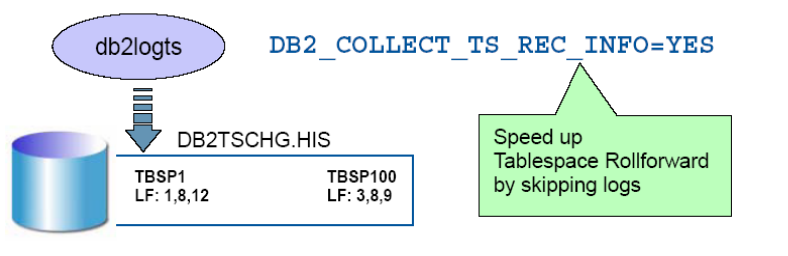

Figure 1608C ··· db2logts 프로세스

TOPIC 1608 데이터베이스 수준의 프로세스

Point

데이터베이스 수준의 프로세스는 ACTIVATE DB 명령어에 의해 데이터베이스가 활성화될 때 생성됩니다. db2pfchr를 비롯하여 db2pclnr, db2loggr, db2loggw, db2logts, db2dlock 등의 프로세스가 있습니다.

Tip

LOGARCHMETH1 데이터베이스 구성 변수는 V8.2 부터 지원됩니다.

Tip

USEREXIT 데이터베이스 구성 변수를 이용하여 사용자가 작성한 user exit program 을 이용하는 방법도 여전히 유효합니다.

5 아카이브 로그 파일의 USEREXIT 과 연관된 프로세스입니다.

프로세스명	생성시점	설명
db2logmgr	db2start 실행시	LOGARCHMETH1 데이터베이스 구성 변수에 아카이브 로그 디렉토리명이 지정된 경우에 사용됩니다. 기존에 사용자가 정의한 user exit program을 사용하지 않고, 엔진이 직접 로그 파일을 아카이브합니다.

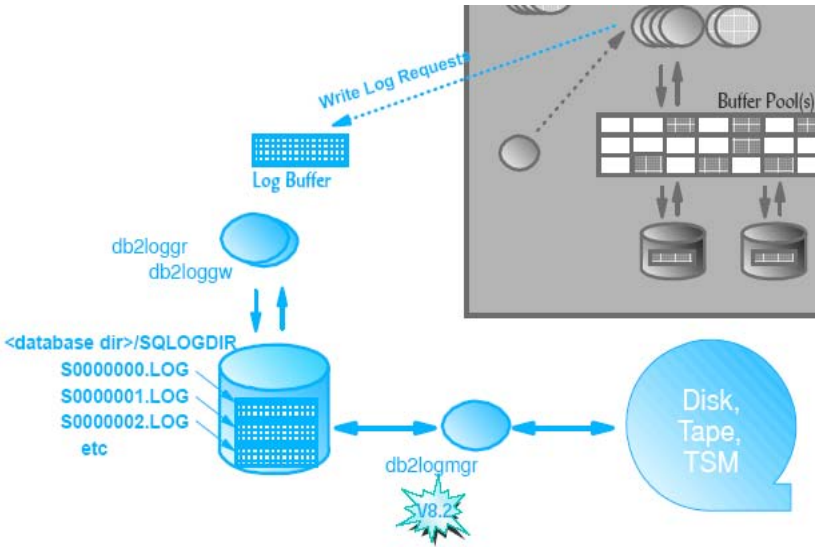

Figure 1608D db2logmgr 프로세스

Tip

확장 스토리지 영역은 데이터베이스 구성 변수인 ESTORE_SEG_SZ 와 NUM_ESTORE_SEGS 가 설정된 경우에만 사용할 수 있습니다. 64비트 머신에서는 필요하지 않습니다.

6 데이터베이스 수준의 프로세스에는 다음과 같은 프로세스들도 있습니다.

프로세스명	생성시점	설명
db2dlock	db2start 실행시	DB2 Local Deadlock Detector 입니다. 개별 파티션마다 존재하며, LOCKLIST를 스캔하여 교착 상태를 검출하는 역할을 합니다. 교착 상태에 있는 응용프로그램 중에서 희생자(victim)로 선택된 응용프로그램은 롤백됩니다. 희생자를 결정하는 우선 순위는 내부적인 기준에 의해 결정됩니다. db2dlock은 Z잠금을 보유한 응용프로그램 또는 load, redistribute, online reorg 등을 실행하는 세션은 희생자 프로세스가 되지 않도록 고려합니다.
db2estor	db2start 실행시	DB2 Extended Storage Manager 입니다. 데이터베이스 버퍼풀과 확장 스토리지 사이에서 데이터 페이지를 복사할 때 사용됩니다. NUM_ESTORE 데이터베이스 구성 변수가 설정된 경우에만 사용됩니다.
db2event	이벤트 모니터 시작시	DB2 Event Monitor 입니다. 활성화된 이벤트 모니터에 대해 각각 한 개씩 할당되어 이벤트 모니터 정보를 수집합니다.

TOPIC 1609 응용프로그램 수준의 프로세스

Point

응용프로그램 수준의 프로세스는 지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청할 때 생성됩니다. db2agent를 비롯하여 db2agntp, db2agnta, db2agentg 등의 프로세스가 있습니다.

Tip

특정한 데이터베이스에 접속할 수 있는 응용프로그램의 최대 개수는 데이터베이스 구성 변수인 MAXAPPLS로 조절합니다.

Tip

특정한 인스턴스에서 생성될 수 있는 Coordinator Agent 프로세스의 최대 개수는 인스턴스 구성 변수인 MAX_COORDAGENTS 로 조절합니다.

Tip

에이전트 프로세스의 총 개수는 인스턴스 구성 변수인 MAXAGENTS를 이용하여 조절합니다.

1 통신 리스너 프로세스는 지역 또는 원격 응용프로그램이 데이터베이스에 대한 접속을 요청하면 다음과 같은 에이전트 프로세스를 생성합니다.

프로세스명	생성시점	설명
db2agent	데이터베이스 접속시	DB2 Coordinator Agent 입니다. 접속을 요청한 응용프로그램을 대신하여 엔진에게 응용프로그램의 요청을 전달하고, 엔진이 반환한 결과를 응용프로그램에게 전달합니다. 응용프로그램별로 한 개의 db2agent 프로세스로 관리합니다. Coordinator Agent 또는 간단하게 Agent 프로세스라고 합니다.
db2agntp	데이터베이스 접속시	DB2 Subagent 입니다. 병렬 처리가 가능한 환경인 경우에 Coordinator Agent는 응용프로그램으로부터 전달받은 데이터베이스 요청들을 Subagent 에게 분배하여, 응용프로그램을 대신하여 엔진에게 요청을 전달하도록 합니다. Subagent 들이 엔진으로부터 받은 결과를 다시 Coordinator Agent 에게 반환하면, Coordinator Agent 가 종합하여 응용프로그램에게 최종적인 결과를 반환합니다.
db2agentg	데이터베이스 접속시	DB2 Gateway Agent 입니다. DB2 Connect 제품을 이용하여 호스트 데이터베이스에 접속하거나, 다른 인스턴스를 경유하여 데이터베이스에 접속을 요청하는 경우에는 db2agent 대신에 db2agentg 프로세스가 생성됩니다.

Tip

병렬 처리의 유형은 파티션내 (Intra-Partition) 병렬 처리와 파티션간 (Inter-Partition) 병렬 처리로 구분됩니다.

Tip

파티션내 (Intra -Partition) 병렬 처리는 단일한 데이터베이스 파티션에서 항 개의 SQL문이 병렬로 처리될 수 있게 합니다. 인스턴스 구성 변수인 INTRA_PARALLEL 옵션을 적용하였을 경우에만 지원됩니다.

Tip

파티션간 (Inter -Partition) 병렬 처리는 다중 데이터베이스 파티션 환경에서 가능합니다. 파티션내 병렬 처리와 함께 지원되도록 할 수 있습니다.

2 단일 데이터베이스 파티션에서 INTRA_PARALLEL 인스턴스 구성 변수를 이용하여 파티션내 (Intra-Partition) 병렬 처리를 지정한 경우에 db2agent와 db2agntp 프로세스의 관계는 다음과 같습니다.

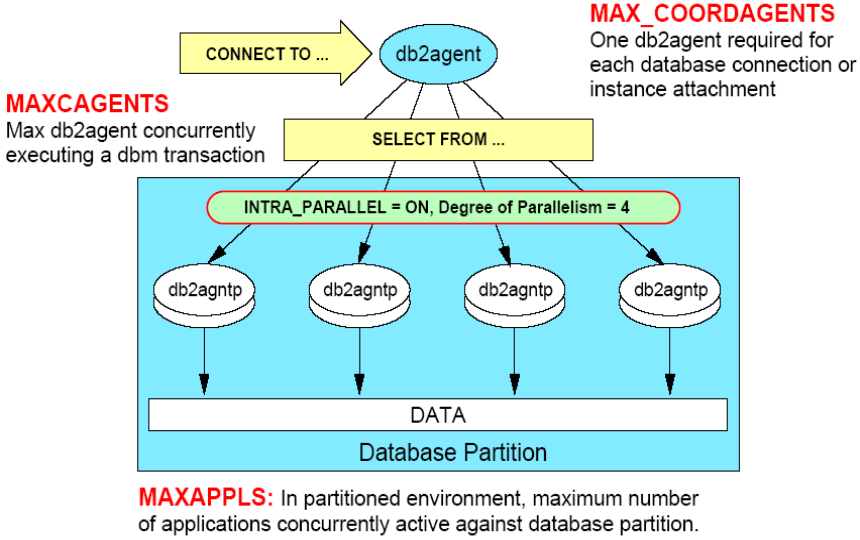

Figure 1609A ··· db2agent 와 db2agntp 프로세스

TOPIC 1609 응용프로그램 수준의 프로세스

Point

응용프로그램 수준의 프로세스는 지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청할 때 생성됩니다. db2agent를 비롯하여 db2agntp, db2agnta, db2agentg 등의 프로세스가 있습니다.

Tip

Agent Pooling을 사용하면, 응용프로그램이 접속을 요청할 때마다 에이전트 프로세스를 생성하는 오버헤드를 줄일 수 있습니다. 인스턴스 구성 변수인 NUM_POOLAGENTS와 NUM_INITAGENTS로 조절합니다.

3 Subagent로 생성되어 사용이 완료된 프로세스는 성능을 위해 즉시 제거하지 않고 Agent Pool 에 저장하였다가 재사용할 수 있습니다.

프로세스명	생성시점	설명
db2agnta	특정 응용프로그램에 대한 작업 종료시	Agent Pool 에서 응용프로그램에게 할당되기를 기다리는 Agent를 Idle Agent라고 하고, 다른 Coordinator Agent에 의해 요청되어 다시 Subagent로 사용됩니다.

4 응용프로그램이 접속을 요청했을 때, Agent 프로세스를 할당하는 우선 순위는 다음과 같은 방법으로 결정됩니다.

1. 요청한 응용프로그램과 연관이 있으면서 Idle 상태에 있는 Subagent를 할당합니다.
2. 다른 응용프로그램과 연관이 있으면서 Idle 상태에 있는 Subagent 를 할당합니다.
3. MAXAGENTS 구성 변수의 값에 도달하지 않았다면 새로운 Subagent를 생성합니다.
4. 다른 응용프로그램과 연관이 있으면서 Idle한 Subagent를 가로채기 합니다.

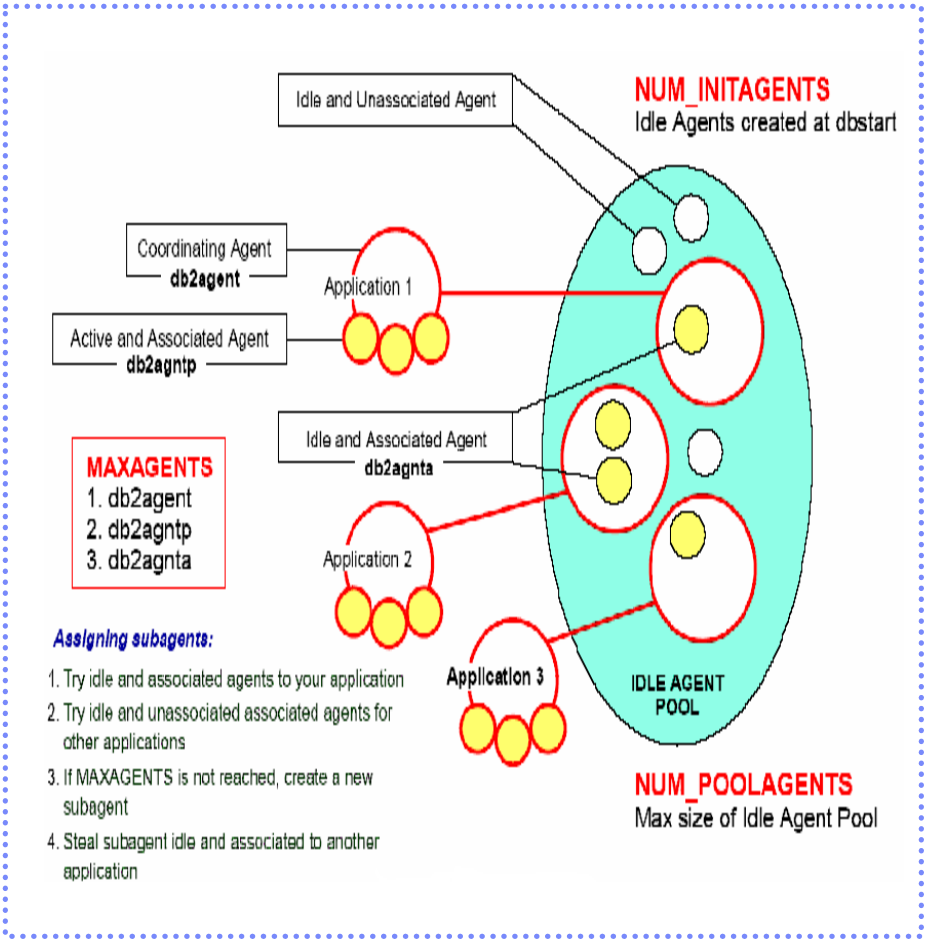

Figure 1609B ••• Agent Pool과 db2agnta 프로세스

TOPIC 1609 응용프로그램 수준의 프로세스

Point

응용프로그램 수준의 프로세스는 지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청할 때 생성됩니다. db2agent를 비롯하여 db2agntp, db2agnta, db2agentg 등의 프로세스가 있습니다.

5 병렬 복구 처리를 위한 프로세스입니다.

프로세스명	생성시점	설명
db2agnsc	데이터베이스 접속시	DB2 Parallel Recovery Agent 입니다. 로그 파일에서 읽어온 로그 레코드의 내용에 따라 rollforward restart db, rollforward 명령을 병렬로 처리하여 복구 유틸리티의 성능을 높이기 위해 사용됩니다. 병렬 처리에 사용되는 Agent 프로세스의 개수는 SMP 머신의 CPU 개수를 고려하여 엔진이 결정합니다.

6 아래 그림에서 Coordinator Agent는 읽어온 로그 레코드를 병렬 프리페처를 이용하여 버퍼풀로 저장합니다. Parallel Recovery Agent 들은 버퍼풀의 데이터를 이용하여 병렬로 복구 작업을 진행합니다.

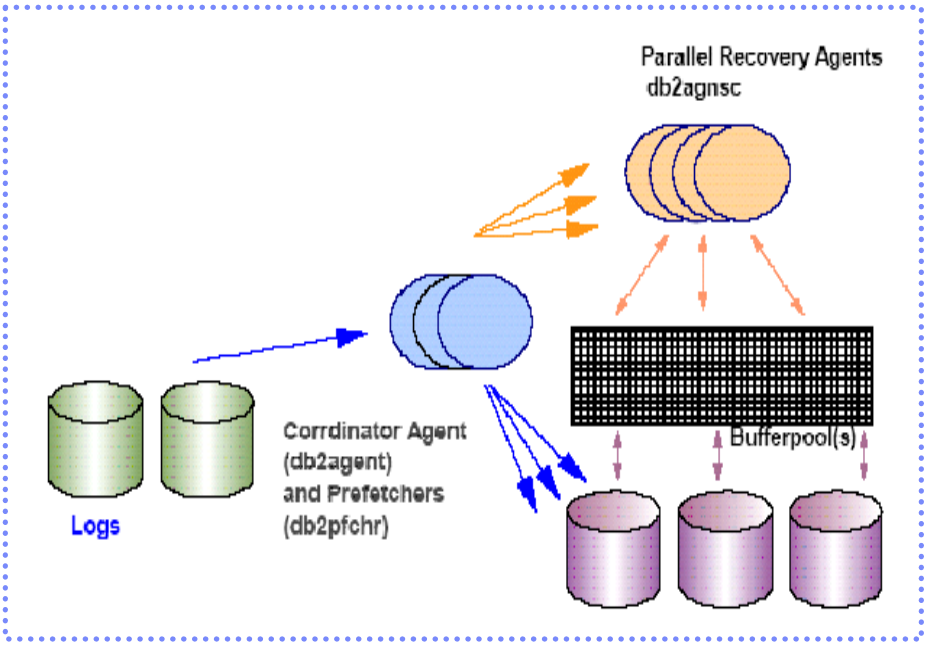

Figure 1609C • • • db2agnsc 프로세스

TOPIC 16-10 메모리 모델

Point

DB2 UDB의 메모리는 인스턴스 공유 메모리, 데이터베이스 공유 메모리, 응용프로그램 공유 메모리, 응용프로그램 개별 메모리로 구성됩니다. 각 수준의 메모리는 인스턴스 기동, 데이터베이스 활성화, 응용프로그램 접속시에 할당됩니다.

Tip

특정한 인스턴스에서 동시에 활성화 될 수 있는 데이터베이스의 최대 개수는 인스턴스 구성 변수인 NUMDB를 이용하여 조절합니다.

Tip

특정한 데이터베이스에 동시에 접속할 수 있는 응용프로그램의 최대 개수는 MAXAPPLS 데이터베이스 구성 변수를 이용하여 조절합니다.

Tip

특정한 인스턴스에서 존재할 수 있는 에이전트 프로세스의 최대 개수는 인스턴스 구성 변수인 MAXAGENTS를 이용하여 조절합니다.

1 DB2 메모리는 다음과 같이 4가지로 구성됩니다.

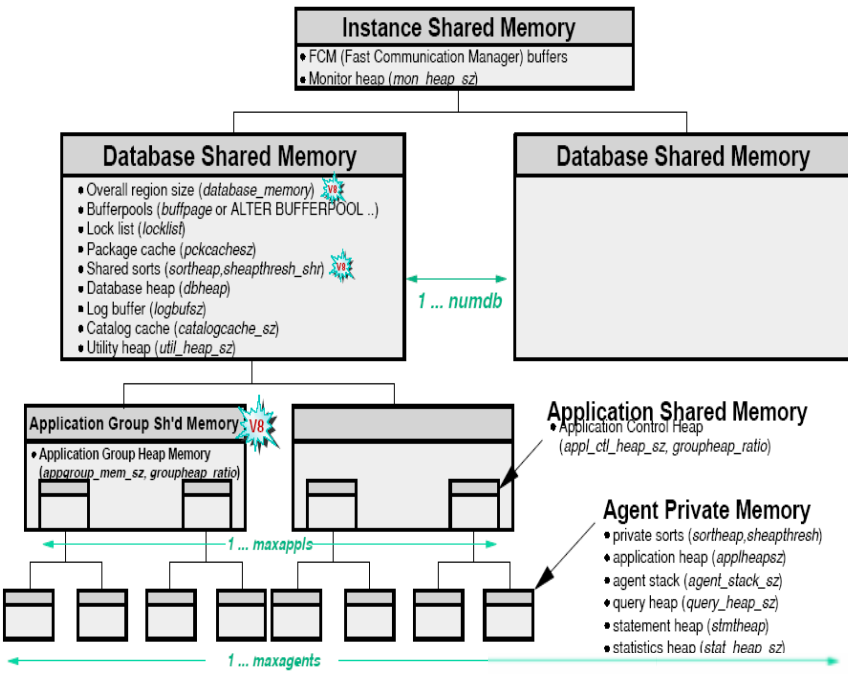

Figure 1610A ••• 메모리 모델

2 각 메모리별 특성은 다음과 같습니다.

메 모 리	설 명
인스턴스 공유 메모리	db2start 명령을 실행하여 인스턴스를 기동할 때 할당되고, db2stop 명령어로 인스턴스를 중지할 때 해제됩니다. Monitor heap, FCM, Audit buffer 등이 포함됩니다.
데이터베이스 공유 메모리	activate db 명령어에 의해 데이터베이스가 활성화될 때 할당되고, deactivate db 명령어에 의해 데이터베이스가 비활성화되면 해제됩니다. Buffer pool 을 비롯하여 Database Heap, Catalog Cache, Log Buffer, Lock List, Package Cache, Shared Sort heap, Utility Heap 등이 포함됩니다.
응용프로그램 공유 메모리	병렬 처리가 가능한 환경에서 응용프로그램의 첫 번째 에이전트 프로세스가 데이터베이스에 연결을 요청하는 경우에 할당되며, 해당 응용프로그램과 연관된 모든 에이전트 프로세스 사이의 정보를 교환을 위해 사용되는 영역입니다. Application control heap 이 포함됩니다.
응용프로그램 개별 메모리	지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청하는 경우에 할당되고, 접속을 종료하면 해제됩니다. Sort Heap, Statement Heap, Application Heap, Statistics Heap, Query Heap 등이 포함됩니다.

TOPIC 16.11 인스턴스 공유 메모리

Point

인스턴스 공유 메모리는 db2start 명령을 실행하여 인스턴스를 기동할 때 할당됩니다. Monitor heap, FCM, Audit buffer 등이 포함됩니다.

1 인스턴스 공유 메모리는 db2start 명령어로 엔진 기동시에 할당되며, 인스턴스에 존재하는 모든 EDU들은 이 영역을 액세스할 수 있습니다.

```
$ login <인스턴스 사용자>
$ db2start
```

2 할당된 인스턴스 공유 메모리의 양은 db2mtrk 명령어로 확인할 수 있습니다.

```
$ db2mtrk -i -v
메모리 추적 설정: 16:30:07에서 2009/08/17
인스턴스용 메모리
  Other Memory의 크기는 13697024바이트입니다.
  FCMBP Heap의 크기는 786432바이트입니다.
  Database Monitor Heap의 크기는 65536바이트입니다.
  총계: 14548992바이트
```

Tip

-i 옵션은 인스턴스 수준의 메모리를 대상으로 지정하고, -v 옵션은 상세한 메모리 사용 정보를 표시합니다.

3 인스턴스 공유 메모리의 양은 get snapshot for dbm 명령어로 확인할 수 있습니다.

```
$ db2 get snapshot for dbm
데이터베이스 관리 프로그램의 메모리 사용:
 노드 번호                    = 0
  메모리 풀 유형                = 기타 메모리
    현재 크기(바이트)           = 13828096
    상위 워터 마크(바이트)      = 13893632
    구성된 크기(바이트)         = 32768000
 노드 번호                    = 0
  메모리 풀 유형                = FCMBP 힙
    현재 크기(바이트)           = 786432
    상위 워터 마크(바이트)      = 786432
    구성된 크기(바이트)         = 917504
 노드 번호                    = 0
  메모리 풀 유형                = 데이터베이스 모니터 힙
    현재 크기(바이트)           = 327680
    상위 워터 마크(바이트)      = 327680
    구성된 크기(바이트)         = 327680
```

Tip

한 개의 물리적인 서버 머신에서 정의된 논리적인 데이터베이스 파티션들은 FCM을 공유할 수 있습니다. AIX에서는 DB2_FCM_FORCE_BP를 설정합니다.

4 인스턴스 공유 메모리는 다음과 같은 메모리 영역으로 구성됩니다.

메모리명	할당시점	설명
Monitor heap	db2start 실행시	스냅샷 모니터, 이벤트 모니터 등을 이용하여 모니터링 정보를 수집할 때 사용되는 메모리 영역으로 MON_HEAP_SZ 인스턴스 구성 변수를 이용하여 조절합니다.
FCM	db2start 실행시	다중 데이터베이스 파티션 환경에서 파티션간의 통신에 사용되는 메모리 영역으로 FCM_NUM_BUFFERS 인스턴스 구성 변수를 이용하여 조절합니다.
Audit buffer	db2audit 실행시	db2audit 유틸리티가 수집한 데이터를 저장하는 audit 용 버퍼 영역입니다. db2chkau 프로세스는 audit 용 버퍼에 저장된 데이터를 db2audit.log 파일에 비동적으로 기록합니다. AUDIT_BUF_SZ 인스턴스 구성 변수를 이용하여 조절합니다.

Tip

db2chkau 프로세스는 audit 용 버퍼가 없으면 db2audit.log 파일에 동기적으로 기록하게 되므로 성능이 저하됩니다.

TOPIC 16-12 데이터베이스 공유 메모리

Point

데이터베이스 공유 메모리는 데이터베이스가 활성화시에 할당됩니다. Buffer pool, Database Heap, Catalog Cache, Log Buffer, Lock List, Package Cache, Shared Sort heap, Utility Heap 등이 포함됩니다.

Tip

-d 옵션은 데이터베이스 수준의 메모리를 대상으로 지정하고, -v옵션은 상세한 메모리 사용 정보를 표시합니다.

1. 데이터베이스 공유 메모리는 activate db 명령어로 데이터베이스 활성화시에 할당되며, 해당 데이터베이스에 연관된 모든 EDU들은 이 영역을 액세스할 수 있습니다.

```
$ login <인스턴스 사용자>
$ db2 activate db <데이터베이스명>
```

2. 할당된 데이터베이스 공유 메모리의 양은 db2mtrk 명령어로 확인할 수 있습니다.

```
$ db2mtrk -d -v
메모리 트랙 위치: 19:48:41의 2005/07/18
데이터베이스용 메모리: SAMPLE
  Backup/Restore/Util Heap의 크기는 16384바이트임
  Package Cache의 크기는 131072바이트임
  Catalog Cache Heap의 크기는 65536바이트임
  Buffer Pool Heap의 크기는 4341760바이트임
  Buffer Pool Heap의 크기는 655360바이트임
  Buffer Pool Heap의 크기는 393216바이트임
  Buffer Pool Heap의 크기는 262144바이트임
  Buffer Pool Heap의 크기는 196608바이트임
  Lock Manager Heap의 크기는 491520바이트임
  Database Heap의 크기는 3555328바이트임
  Other Memory의 크기는 0바이트임
  총계: 10108928바이트
```

3. 데이터베이스 공유 메모리의 양은 get snapshot for db 명령어로 확인할 수 있습니다.

```
$ db2 get snapshot for db on <데이터베이스명>
데이터베이스의 메모리 사용:
  메모리 풀 유형                = 백업/리스토어/유틸리티 힙
    현재 크기(바이트)            = 16384
    상위 워터 마크(바이트)       = 16384
    구성된 크기(바이트)          = 20496384
  메모리 풀 유형                = 패키지 캐시 힙
    현재 크기(바이트)            = 131072
    상위 워터 마크(바이트)       = 131072
    구성된 크기(바이트)          = 2147483648
  메모리 풀 유형                = 카탈로그 캐시 힙
    현재 크기(바이트)            = 65536
    상위 워터 마크(바이트)       = 65536
    구성된 크기(바이트)          = 2147483648
  …
```

TOPIC 16-12 데이터베이스 공유 메모리

Point

데이터베이스 공유 메모리는 데이터베이스가 활성화시에 할당됩니다. Buffer pool, Database Heap, Catalog Cache, Log Buffer, Lock List, Package Cache, Shared Sort heap, Utility Heap 등이 포함됩니다.

4 데이터베이스 공유 메모리는 다음과 같은 메모리 영역으로 구성됩니다.

메모리명	할당시점	설명
Buffer pool	데이터베이스 활성화시	데이터베이스 공유 메모리에서 가장 큰 메모리 영역으로 다중 버퍼풀을 생성하여 테이블스페이스별로 할당할 수 있습니다. 데이터베이스가 활성화될 때 할당되며, CREATE / ALTER / DROP BUFFERPOOL문에 의해 동적으로 버퍼풀을 추가하거나, 크기를 조절할 수 있습니다.
Database Heap	데이터베이스 활성화시	테이블, 인덱스, 테이블스페이스, 버퍼풀 등의 데이터베이스 오브젝트에 대한 Control Block 정보를 저장하고, Log Buffer와 Catalog Cache를 포함하는 영역으로 데이터베이스 구성 변수인 DBHEAP 을 이용하여 동적으로 조절할 수 있습니다.
Catalog Cache	데이터베이스 활성화시	SQL문을 컴파일하는 동안 SYSIBM.SYSTABLES, SYSIBM.SYSDBAUTH, SYSIBM.SYSROUTINES 등의 시스템 카탈로그의 정보를 저장하는 메모리 영역으로 Database Heap 에서 할당되며 데이터베이스 구성 변수인 CATALOGCACHE_SZ 를 이용하여 동적으로 조절할 수 있습니다.
Log Buffer	데이터베이스 활성화시	로그 레코드를 디스크의 로그 파일에 기록하기 전에 버퍼링하는 메모리 영역으로 Database Heap 에서 할당되며 데이터베이스 구성 변수인 LOGBUFSZ 를 이용하여 동적으로 조절할 수 있습니다.
Lock List	데이터베이스 활성화시	데이터베이스에 접속된 모든 응용프로그램이 보유하고 있는 테이블과 행 잠금 정보를 저장하는 메모리 영역입니다. 데이터베이스 구성 변수인 LOCKLIST 를 이용하여 동적으로 조절할 수 있습니다.
Package Cache	데이터베이스 활성화시	static 및 dynamic SQL문을 저장하는 캐시 영역으로 Static SQL문이 포함된 팩키지를 reload 하거나 dynamic SQL문을 재컴파일하는 내부적인 오버헤드를 줄여줍니다. PCKCACHESZ 데이터베이스 구성 변수를 이용하여 동적으로 조절할 수 있습니다.
Shared sort heap	데이터베이스 활성화시	INTRA_PARALLEL 인스턴스 구성 변수를 설정하거나 Connection Concentrator 기능을 사용한다면, SQL문에 대한 병렬 처리가 가능합니다. Shared Sort Memory는 Subagent들이 정렬 작업을 병렬로 처리하고, 그 결과를 공유하기 위해 사용하는 메모리 영역입니다. SHEAPTHRES_SHR 데이터베이스 구성 변수로 조절합니다.
Utility Heap	Utility 실행시	Load, Backup, Restore, Runstats 등의 유틸리티가 실행될 때 사용하는 메모리 영역으로 동적으로 조절이 가능합니다. 유틸리티가 수행될 때 할당되고, 유틸리티가 종료되면 반환됩니다. 데이터베이스 구성 변수인 UTIL_HEAP_SZ 를 이용하여 조절합니다.

Tip

32비트 데이터베이스에서는 한 개의 잠금이 40 또는 80 바이트를 차지하고, 64비트 데이터베이스에서는 64 또는 128 바이트를 차지합니다.

Tip

SHEAPTHRES_SHR 데이터베이스 구성 변수를 0으로 설정하면, 데이터베이스 구성 변수인 SHEAPTHRES의 값을 사용합니다.

TOPIC 16-13 응용프로그램 공유 메모리

Point

병렬 처리가 가능한 환경에서 응용프로그램의 데이터베이스에 연결을 요청하는 경우에 할당되며, 해당 응용프로그램과 연관된 모든 에이전트 프로세스 사이의 정보를 교환을 위해 사용되는 영역입니다. Application control heap 이 포함됩니다.

Tip

DFT_DEGREE 데이터베이스 구성 변수의 기본값은 1 이고, 인스턴스 구성 변수인 INTRA_PARALLEL 기본값은 NO 이므로 단일 데이터베이스 파티션 환경에서 기본적으로 SQL문은 병렬로 처리되지 않습니다.

Tip

DFT_DEGREE 데이터베이스 구성 변수의 값에서 지정하는 병렬 등급은 1 ~ 32767입니다. -1을 지정하면, 옵티마이저가 프로세서의 개수 및 쿼리 유형에 기초하여 파티션내 병렬 처리 등급을 결정하게 됩니다.

Tip

처음에는 기본값으로 시작하고, 복잡한 응용프로그램을 실행하거나 데이터베이스 파티션 개수가 많은 경우 또는 선언된 임시 테이블을 사용할 경우에는 값을 증가시킵니다.

1. 응용프로그램 공유 메모리는 응용프로그램의 첫 번째 에이전트 프로세스가 데이터베이스에 연결을 요청하는 경우에 할당되며, 응용프로그램이 완료되면, 반환됩니다. 해당 응용프로그램과 연관된 모든 EDU들이 액세스할 수 있습니다.

2. 이 영역은 다중 데이터베이스 파티션 환경이나 INTRA_PARALLEL 구성 변수를 이용한 파티션내 병렬 처리가 가능한 환경에서 할당됩니다. 이 메모리 영역의 용도는 특정한 응용프로그램의 요청을 처리하는 Subagent와 Coordinator Agent 간의 정보 교환입니다. 단일 데이터베이스 파티션 환경이거나 INTRA_PARALLEL 구성 변수를 이용하였지만 쿼리에 대한 병렬 처리 등급인 1인 경우에는 이 영역을 사용은 최소화됩니다.

```
$ login <인스턴스 사용자>
$ db2 update dbm cfg using INTRA_PARALLEL YES
$ db2 update db cfg for <데이터베이스명> using DFT_DEGREE <병렬 등급>
$ db2stop force
$ db2start
$ db2 get dbm cfg | grep INTRA_PARALLEL
 파티션내 병렬 처리 사용            (INTRA_PARALLEL) = YES
$ db2 get db cfg for <데이터베이스명> |  grep DFT_DEGREE
 병렬 처리 등급                     (DFT_DEGREE) = <병렬 등급>
```

3. SQL문의 파티션 내 병렬 처리 수준은 CURRENT DEGREE 특수 레지스터 또는 DEGREE 바인드 옵션을 사용하여 명령문 컴파일시 지정됩니다. 기본값은 DFT_DEGREE 데이터베이스 구성 변수에 지정된 값입니다. 실행 중인 응용프로그램의 파티션 내 병렬 처리의 최대 런타임 수준은 SET RUNTIME DEGREE 명령을 사용하여 지정됩니다.

4. 에이전트 공유 메모리는 다음과 같은 메모리 영역으로 구성됩니다.

메모리명	할당시점	설명
Application control heap	데이터베이스 연결시	다중 데이터베이스 파티션의 파티션간 병렬 처리 또는 INTRA_PARALLEL 인스턴스 구성 변수를 이용한 파티션내 병렬 처리가 가능한 환경에서 응용프로그램의 첫 번째 에이전트 프로세스가 데이터베이스에 연결을 요청하는 경우에 할당되며, 해당 응용프로그램과 연관된 모든 에이전트 프로세스 사이의 정보를 교환하기 위해 사용되는 영역입니다. 이 영역은 선언된 임시 테이블에 대한 디스크립터 정보를 저장하는 데도 사용됩니다. 명시적으로 삭제되지 않은 선언된 모든 임시 테이블에 대한 디스크립터 정보가 보존되므로 선언된 임시 테이블이 삭제될 때까지 메모리를 반환할 수 없습니다. 파티션마다 각 연결에 한 개씩 할당되며, APP_CTL_HEAP_SZ 데이터베이스 구성 변수로 조절할 수 있습니다.

TOPIC 16.14 응용프로그램 개별 메모리

Point

지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청하는 경우에 할당되고, 접속을 종료하면 해제됩니다. Sort Heap, Statement Heap, Application Heap, Statistics Heap, Query Heap 등이 포함됩니다.

Tip

-p 옵션은 에이전트 수준의 메모리를 대상으로 지정하고, -v옵션은 상세한 메모리 사용 정보를 표시합니다.

1 에이전트 개별 메모리는 connect 명령어로 데이터베이스에 접속할 때 생성되는 에이전트 프로세스에게 개별적으로 할당되는 메모리 영역입니다.

```
$ login <인스턴스 사용자>
$ db2 connect to <데이터베이스명>
```

2 할당된 데이터베이스 공유 메모리의 양은 db2mtrk 명령어로 확인할 수 있습니다.

```
$ db2mtrk -p -v
메모리 추적 설정: 16:43:40에서 2009/08/17
에이전트 3708용 메모리

   Other Memory의 크기는 720896바이트입니다.
   총계: 720896바이트
```

3 특정한 에이전트 개별 메모리의 양은 확인하려면, list applications 명령어로 에이전트에 대응되는 핸들 번호를 확인합니다.

```
$ db2 list applications
권한 ID  응용프로그램 이름  Appl. 핸들  응용프로그램 ID                          DB 이름  에이전트 수
-------- ---------------- ---------- ---------------------------------------- -------- ----------
ADF      db2bp.exe        146        *LOCAL.DB2.090817073144                  SAMPLE   1
```

4 get snapshot for application 명령어에서 agentid 옵션으로 특정한 에이전트 개별 메모리의 양을 확인합니다.

```
$ db2 get snapshot for application agentid <응용프로그램 핸들 번호>
응용프로그램의 메모리 사용:

  메모리 풀 유형                = 응용프로그램 힙
    현재 크기(바이트)           = 65536
    상위 워터 마크(바이트)      = 65536
    구성된 크기(바이트)         = 1048576

에이전트 프로세스/스레드 ID     = 3708
  에이전트 잠금 시간종료(초)    = -1
  에이전트의 메모리 사용:

    메모리 풀 유형              = 기타 메모리
      현재 크기(바이트)         = 720896
      상위 워터 마크(바이트)    = 917504
      구성된 크기(바이트)       = 3195011072
```

TOPIC 16-14 응용프로그램 개별 메모리

Point

지역 또는 원격 응용프로그램이 데이터베이스에 접속을 요청하는 경우에 할당되고, 접속을 종료하면 해제됩니다. Sort Heap, Statement Heap, Application Heap, Statistics Heap, Query Heap 등이 포함됩니다.

Tip

Sort heap 영역이 부족하게 되면, 임시 테이블스페이스에 임시 파일 또는 테이블을 생성하게 되므로 성능이 저하됩니다.

5 에이전트 개별 메모리는 다음과 같은 메모리 영역으로 구성됩니다.

메모리명	할당시점	설명
Sort Heap	정렬 작업,해쉬 조인	정렬 작업 또는 해쉬 조인(Hash Join)이 필요할 때 할당되고, 작업이 종료되면 메모리를 반환합니다. SORTHEAP 데이터베이스 구성 변수를 이용하여 동적으로 조절이 가능합니다.
Statement Heap	프리컴파일, 바인드	SQL 컴파일러의 작업 영역으로 사용되는 메모리 영역으로 프리컴파일 또는 바인드 작업시에 각 SQL문에 대해 할당되고, 작업이 종료되면 반환됩니다. STMTHEAP 데이터베이스 구성 변수를 이용하여 동적으로 조절이 가능합니다.
Application Heap	데이터베이스 접속시	db2agent 프로세스의 개별 작업 영역으로 사용되는 메모리로 에이전트와 서브에이전트 프로세스가 SQL문에 대한 팩키지 중에서 실행 가능한 섹션을 복사하여 저장하는데 사용됩니다. SQL문에 대한 실행 섹션은 요청된 SQL문을 수행하기 위해 필요한 정보를 제공하며, 이 메모리 영역에 캐시된 섹션은 재사용되므로 동일한 SQL문을 수행하는데 필요한 오버헤드를 줄일 수 있습니다. APPLHEAPSZ 데이터베이스 구성 변수를 이용하여 조절이 가능합니다.
Statistics Heap	Runstats 실행시	통계 자료를 갱신하기 위하여 runstats 명령어가 실행될 때만 Utility Heap으로 부터 할당받는 메모리 영역입니다. runstats 명령어에서 with distribution 옵션으로 분산 통계 정보를 수집하는 경우에는 더 많은 메모리를 사용하게 됩니다. STAT_HEAP_SZ 데이터베이스 구성 변수를 이용하여 조절이 가능합니다.
Query Heap	데이터베이스 접속시	db2agent 프로세스가 쿼리문, SQLCA, SQLDA, 펙키지의 이름과 생성자, 섹션 번호, 일관성 토큰 등을 저장하는데 사용됩니다. QUERY_HEAP_SZ 인스턴스 구성 변수를 이용하여 조절합니다.

TOPIC 16-15 Thread 모니터링

Point

9.5이전에는 ps명령어 또는 db2_local_ps 명령어로 Active한 EDU를 리스트할 수 있었습니다. 그러나 9.5부터는 db2sysc밖에는 표시되지 않습니다.

Tip

9.5이후 버전에 해당됩니다.

1 아키텍처가 변경됨에 따라 DBA도 관리방법이 변경됩니다.

```
$ ps -fu db2ins10
    UID    PID    PPID   C    STIME    TTY  TIME CMD
db2ins10 1237176 2109662   0   Feb 28      -  0:12 db2acd 0
db2ins10 1921136 2109662   0   Feb 28      -  0:14 db2sysc 0
db2ins10 2101494 1941686   0 14:22:34  pts/1  0:00 -ksh
db2ins10 2420958 2101494   0 15:25:33  pts/1  0:00 ps -fu db2ins10
```

2 db2sysc가 1921135 입니다. AIX에서 모든 Thread를 확인하려면 아래와 같이 입력합니다.

```
 $ db2pd -edu

Database Partition 0 -- Active -- Up 1 days 01:05:54
List of all EDUs for database partition 0

db2sysc PID: 1921136
db2wdog PID: 2109662
db2acd  PID: 1237176


EDU ID    TID       Kernel TID    EDU Name            USR         SYS
============================================================
    ========================
1801      1801        2216003      db2agent (idle) 0    0.706935    1.071737
1543      1543        5628153      db2resync 0          0.002641    0.004271
1286      1286        1851457      db2ipccm 0           0.082388    0.044037
1029      1029        2023571      db2licc 0            0.000211    0.001055
772       772         4390991      db2thcln 0           0.000244    0.000105
515       515         2621455      db2aiothr 0          2.740874    6.287562
2         2         1273899      db2alarm 0           0.274076    0.408226
258       258         2658531      db2sysc 0            2.085981    1.379128
```

TOPIC 16 16 Memory 사용량 모니터링

Point

다중 스레드 모델에서 메모리 사용량에 대한 모니터링 방법을 소개합니다.

Tip

9.5이후 버전에 해당됩니다.

1 4가지 방법이 제공됩니다.

```
1) db2pd -dbptnmem
2) db2 get snapshot for applications on sample
3) select * from table(admin_get_dbp_mem_usage())
4) db2mtrk -a and db2mtrk -p
```

2 db2pd 사용

```
$ db2pd -dbptnmem

Database Partition 0 -- Active -- Up 1 days 01:11:27

Database Partition Memory Controller Statistics

Controller Automatic: Y
Memory Limit:        13994636 KB
Current usage:       76608 KB
HWM usage:           332736 KB
Cached memory:       16064 KB

Individual Memory Consumers:

Name          Mem Used (KB) HWM Used (KB) Cached (KB)
========================================================
DBMS-db2ins10          46784       46784      10048
FMP_RESOURCES          22528       22528          0
PRIVATE                 7296        7296       6016
```

3 db2 get snapshot 사용

```
$ db2 get snapshot for applications on sample

Memory usage for application:

 Memory Pool Type                  = Application Heap
   Current size (bytes)            = 65536
   High water mark (bytes)         = 65536
   Configured size (bytes)         = 1048576

Agent process/thread ID            = 6463
 Agent Lock timeout (seconds)      = -1
 Memory usage for agent:

  Memory Pool Type                 = Other Memory
    Current size (bytes)           = 196608
    High water mark (bytes)        = 196608
    Configured size (bytes)        = 16710107136
```

TOPIC 16 16 Memory 사용량 모니터링

Point

다중 스레드 모델에서 메모리 사용량에 대한 모니터링 방법을 소개합니다.

Tip

9.5이후 버전에 해당됩니다.

4 SQL 사용

```
$ db2 "select * from table(admin_get_dbp_mem_usage())"
    DBPARTITIONNUM MAX_PARTITION_MEM
    CURRENT_PARTITION_MEM PEAK_PARTITION_MEM ---------- --
    ---------- ---------- ---------- 0 14330507264
    340590592 340852736 1 record(s) selected.
```

5 db2mtrk 사용 (db2mtrk –a 또는 db2mtrk –p)

```
 $ db2mtrk -a
Tracking Memory on: 2008/02/29 at 15:51:00

Application Memory for database: SAMPLE
  appshrh
  128.0K

 Memory for application 546
  apph       other
  64.0K      192.0K

 Memory for application 545
  apph       other
  64.0K      192.0K

 Memory for application 544
  apph       other
  64.0K      320.0K

 Memory for application 543
  apph       other
  64.0K      576.0K

 Memory for application 547
  apph       other
  64.0K      192.0K
```

Memo