J2EE환경에서 DB2 for z/OS 성능 향상

최병석, 최현영, 민성기

고려대학교 컴퓨터과학기술대학원
e-mail : cbsid.choi@samsungfire.com,{neongas,sgmin}@korea.ac.kr

Performance Improvement of DB2 for z/OS In J2EE Environment

Byung-seok Choi, Hyon-young Choi, Sung-Gi Min
Dept. of Software Engineering
Graduate School of Computer Science & Technology
Korea University

요 약

J2EE 환경 기반의 IBM DB2 for z/OS 데이터 베이스를 적용한 어플리케이션에서 성능이 상대적으로 저하되는 문제가 발생 하였다. 즉, J2EE (Java 2, Enterprise Edition) 환경에서 동일한 어플리케이션을 Unix 서버의 오라클 데이터 베이스와 IBM DB2에 적용하여 비교한 결과 DB2부문에 대한 TPS가 작고 Transaction Elapsed Time이 오래 걸리며 CPU Utilization이 낮았다. 이러한 초기 결과값을 Baseline으로 설정하여 DB2를 데이터 베이스 서버로 사용하는 경우 어플리케이션의 성능을 개선 하기 위한 요소를 도출하고 성능 향상을 위한 조정 작업을 통해 DB2의 성능을 개선했다.

서론

1.1 메인프레임 상호 운용성

국내 많은 대형 금융기관은 상대적으로 안정된 IBM Host중심으로 COBOL기반의 CICS(Customer Information Control System)나 IMS(Information Management System) 환경으로 시스템을 운영하면서 서버를 바탕으로 하는 개방형 시스템과 상호 운영 체제를 유지하고 있다. 메인프레임 상호 운용성은 메인프레임 아키텍춰 위에서 웹서비스를 구축하는 것보다 총 소요비용이 적게 들고 낙후된 COBOL기술에서 개방형 표준과 J2EE로 전환하는 전략이 Java 기술을 사용한 쉬운 통합과 상호 운영이 가능하기 때문이다. 메인프레임과 개방형 시스템의 상호 운영을 위한 통합기술 채택은 매우 중요하며 메세징 교환 플랫폼과 함께 관련 시스템의 구성 요소의 성능이 계속 향상되고 있다.

데이터 저장 방식도 메인프레임 환경의 관계형 데이터 베이스가 통합되어 운영되고 있으며 데이터의 축적과 활용 목적에 따라 개방형 시스템 환경의 데이터 베이스 구조로 점진적 이전이 진행되고 있다.

따라서, 개방형 시스템인 J2EE환경으로 개발하여 적용한 기업형 어플리케이션은 메인프레임 상호 운영 환경에서 메인프레임 시스템의 관계형 데이터 베이스를 이용하는 사례가 늘어 나고 있고 개방형 시스템의 관계형 데이터 베이스를 Access 하는 것 보다 성능에 대한 차이가 발생되어 사례를 통해 접근 하고자 한다.

1.2 성능요소와 조정

메인프레임 관계형 데이터 베이스 시스템인 IBM DB2를 이용하여 J2EE환경으로 개발 적용한 기업형 어플리케이션에서는 상대적으로 성능 저하 문제가 발생 하였고 개방형 시스템 환경과 메인프레임 DB2에 대한 구성요소와 옵션 및 어플리케이션에 적용한 내용에 대해 성능에 영향을 주는 관련 요인을 확인하고 분석을 통해 조정함으로써 성능저하 문제를 예시한 다음 접근 방법을 확인하고 실제 시스템에 적용하여 성능 조정사례 및 개선요인을 제시한다.

이 논문을 통하여 J2EE환경에서 사용하는 DB2 시스템 관련 DBA나 개발자에게 성능 향상을 위한 접근방법과 절차를 소개함으로써 수행성능 문제를 해결하는데 초점을 두었다.

1.3 논문의 구성

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구들을 정리하였다. 그리고 3장에서는 초기에 적용 했을 때 성능에 관련 되는 요소들을 확인하고 검증 할 수 있는 방법을 설명한다. 4장에서는 각각의 성능 요소를 분석하고 성능 저하를 해결하는 가장 최적의 각각의 성능요소 상태와 종합적인 연계를 감안하여 개선한 내용을 제시하고 개방형과 메인프레임에 적용한 초기값과 개선후 값을 비교하여 설명하고 5장에서는 결론을 서술한다.

관련 연구

1.1 멀티티어 시스템 성능 조정의 개요

기존 메인프레임 환경으로만 구축된 DB2 시스템의성능 요소는 어플리케이션디자인, DB2 서브시스템, CICS 환경, MVS(Multiple Virtual Storage) 시스템으로 나뉘며 이 중 어플리케이션 디자인이 가장 가능한 조정의 기대치를 제공한다[2], [11]

그러나 메인 프레임 DB2 시스템과 J2EE 환경의 서버 시스템간 상호 운영되는 멀티티어(Multi-Tire) 시스템을 구축하는 경우, 메인프레임 환경의 DB2 성능 요소 외에 J2EE환경의 서버 성능 요소를 추가하고, 각 구성 요소 하나 하나가 최적의 상태로 성능 조정이 되어야 하며, 동시에 각각의 연계성 또한 중요하게 고려 되어야 한다. 이와 관련하여 분석 검토 되어야 할 각 구성 요소별 성능 요인을 CICS와 DB2로 구축된 메인프레임 환경의 시스템 성능 요소와 J2EE 시스템 환경에서의 성능요소를 구분하여 접근해 보고 네트워크 성능 요소를 별도로 확인한다.

Java 어플리케이션에서 WAS(Web Application Server)를 통해 DB2를 Access할 때 DB2 Transit Time에 대한 성능 모니터링으로 상태를 점검하여 성능 요소와 관련된 DB2 관련 SQL 조정으로 성능 개선 가능성을 확인하고, DB2 서브시스템과 데이터 베이스에 대한 옵션의 조정으로 DB2 Elapsed Time을 점검한다.

그리고, J2EE 환경에서 웹서버, WAS, JDBC 드라이버 적용상태와 어플리케이션에서 SQL 통합으로 호출 수를 줄이는 방안과 상호 운영에 따른 코드 타입의 영향을 확인하며 각각 성능 요소의 최적 상태를 확인한다.

2.2 네트워크 성능 요소

서버 네트워크 환경 구축시 DB서버와 어플리케이션 서버의 위치는SQL Call을 네트워크를 통해 하므로 동일 LAN 환경에 위치하는 것과 WAN 환경에 위치 하느냐에 따라 처리 용량이 결정되는 것으로 통신선의 속도와 함께 중요한 요소의 하나이다. 그러나, 현실적으로 WAS서버의 이동이 어려워 동일 LAN상에서 테스트를 하지 못했고 기본적인 Ping테스트를 통해 간단하게 비교하였다.

2.3 성능 분석 시스템

기업의 비즈니스 룰과 컴포넌트를 연계한 시스템을 개발하여 J2EE 환경에 배치하였으며 시스템 환경 구성은 그림 1, 2와 같이 DB2 V7.0과 오라클 9i 데이터베이스를 사용하였고, 웹서버는 I-Plant 6.1을, WAS는 Weblogic 8.1을, 스트레스 테스트 도구로는 Load Runner를 사용하였다.[3], [5]

그림 1 시스템 환경 구성

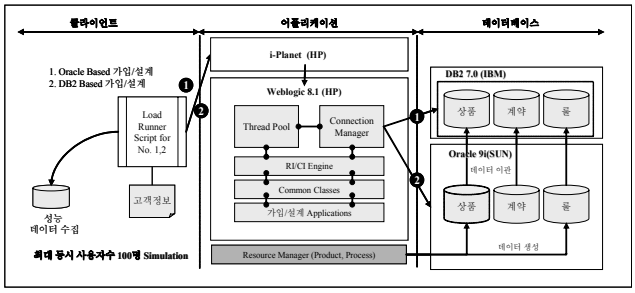


그림 2 시범 시스템 환경

동일한 웹서버, WAS환경에서 성능 비교를 위한 초기 테스트 결과는 표1과 같이 오라클이 WAS CPU율이 95%인 상황에서 121.1 TPS임에 비해 DB2는 WAS CPU율이 90%인 상황에서 89.4 TPS로 DB2가 성능이 저하됨을 알 수 있다.

DBMS 구 분	Version	JDBC Type	WAS Instance	Pool 수	Thread	JVM Size	Max User	WAS CPU율	TPS
Oracle	9i	4	2	50	50	1G	100	95%	121.1
DB2	7	4	2	50	50	1G	100	90%	89.4

표 1 성능 Base Line

3. 튜닝 요소 정의

3.1 CICS DB2 성능 모니터링

DB2의 성능 모니터링 결과 기본적으로 도출될 수 있는 것이 Lock및 Latch 시간과 갱신 확인 처리 시간 (Update Commit Time)이다.

DB2에서 Lock을 처리하는 동안 IRLM (Information Resource Lock Manager)에게Lock을 처리하도록 하는 명령어가 길어 자원을 많이 사용한다. 이와 반대로 순차적으로 처리 할 수 있도록 DBM1(Database Service)에서 Lock을 처리하는 것이 Latch이며 명령어가 Lock의 1/20로 빠른 편인 반면 CPU가 많이 소요된다. 이러한 Lock과 Latch시간은 거의 없는 것이 최적이며 상대적으로 높은 결과가 나올 경우 분석하여 조정한다.

또한, 갱신 확인 처리 횟수가 높으면 네트워크 통신량의 증가 및 DB2부하의 증가를 초래하며 보통 0.001x이하의 시간이 최적이며 상대적으로 높은 결과가 나올 경우 분석하여 조정 한다.

3.2 DB2 서브시스템 성능 요소

DB2 서브시스템의 성능은 4가지로 나누어서 하는데 첫 번째는 풀(Pool) 조정으로 버퍼 풀(Buffer Pool), EDM풀

(Environmental Descriptor Manager Pool), RID 풀(Record Identifier Pool), 정렬 풀(SORT Pool)을 조정하여야 한다.
두 번째는 로그(Log) 조정으로 출력 버퍼 크기(Output Buffer Size), 활동 로그(Active Log), 보관 로그(Archive Log)를 조정하여야 한다.
세 번째는 DB가 확인처리나 롤백(Rollback) 처리를 하고 나서 해당 스레드의 활성화 여부 설정옵션을 성능과 자원의 부하를 고려하여 조정하여야 한다.
네 번째는 WLM(Work Load Manager) 적용 검증으로
WLM은 목표 성능을 맞추도록 자원을 각 워크로드(Workload)에 자동으로 분배하는 하는 것으로 이를 조정함으로써 성능 향상이 가능하다.

3.3 데이터베이스 성능 요소

데이터 베이스의 성능도 4가지로 나누어서 하는데
첫 번째는 DB 설계 및 조정 작업시 DBMS의 특성을 고려하여 데이터의 저장소, 데이터 베이스, 테이블 공간(Table Space), 테이블의 운영관리 기준을 준수함으로써 성능 향상을 얻을 수 있다
두 번째는 유니코드(Unicode) 적용을 테스트하여Host-EBCDIC, 서버-ASCII 코드의 서로 다른 코드의 사용 으로 인한 데이터 유실이나 코드전환에 따른 부하를 줄일 수 있는지 검증한다.
세 번째는 테이블 검색 시 인덱스의 사용은 성능에 커다란 영향을 미친다. DB2에서는 Optimizer가 인덱스의 사용을 결정하지만, 처리형태를 감안하여 인덱스의 사용을 유도하거나 사용하는 테이블의 검색유형에 맞게 인덱스를 추가 하여야 한다.
네 번째는 가장 기본적인 성능 요소인 접근경로를 결정하기 위해DB2에서는 Optimizer가 수행하는데 DB2 카타로그 통계(Catalog Statistics)와 SQL문장을 기초로 비용을 산정하여 가장 비용이 낮은 접근 경로를 선택한다. 이를 위해 데이터 저장 상태를 조정하여야 한다.

3.4 어플리케이션 성능 요소

어플리케이션 성능은 세가지 영역으로 조정하는데
첫 번째는 SQL조정으로 접근 경로(Access Path)를 도출하여 조정하는 것과 다른 하나는 적절한 조인 (Join)방법의 선택으로 수행한다.
두 번째는 JDBC 어플리케이션에서 사용하는 데이터 타입을 검증해 본다.
세 번째는 네트워크 트래픽(Traffic)은 멀티티어 시스템 환경에서 특히 중요하며, 가능한 트래픽을 감소시키는 방향으로 SQL 통합관점에서 어플리케이션 설계를 검증한다.

3.5 웹서버 성능 요소

KEEPALIVE는 웹서버에서 WAS서버로의 연결을 계속 유지하도록 하는 설정 옵션이며, 이 설정을 통해대기 시간을 줄일 수 있으므로 적용여부를 결정한다.

3.6 WAS서버 성능 요소

3.6.1 Dynamic Statement Cache 적용 여부 검증
WAS에서 SQL요청시 DB2측에서 보면, 동적 SQL수행시 접근경로를 결정하는 준비 과정을 매번 반복 수행하므로 CPU 부하를 높인다.
한번 사용된 접근 경로와 최적화 결과를 저장하고 재사용 가능하게 함으로써 시스템 성능을 향상 시킬 수 있다.
3.6.2 Auto commit 적용 여부 검증
일반적으로 Auto commit이 기본으로 설정되어 매 SQL마다 자동으로 확인을 수행하여 불필요한 부하가 발생한다. 따라서 Auto commit설정 조정을 통해 성능향상이 가능하다.

3.6.3 Keep alive적용 여부 검증
비 활성화 상태의 시간설정 옵션에서Connection Interface의 재사용과 관련되어 성능에 중요한 영향을 주는 요소이므로 Keep alive 설정 조정을 통해 성능 향상이 가능하다.

3.7 JDBC 드라이버 성능 요소

3.7.1 JDBC 드라이버 타입 선정
JDBC는 Java 어플리케이션이 관계형 데이터 베이스에 접근에 사용하는API(Application Programming Interface)이다. 적절한 JDBC 드라이버 타입을 적용함으로써 성능향상을 기대할 수 있다.

4. 성능 향상을 위한 조정 CASE STUDY

4.1 성능 향상 조정 방법

성능 향상을 위한 요소들을 각각 분석과 테스트를
해야 하나 현실적으로 어려운 점이 있어 종합적으로
결합하여 조정 작업을 실시하였다.

4.2 성능 향상을 위한 조정

4.2.1 성능 모니터링 분석에 의한 조정
1) Lock/Latch Time 분석
SQL로직 분석결과 채번Logic에서 Lock이 많이 발생되고 있어 테이블에 IDENTITY 컬럼을 추가하고 프로그램에서는 해당 IDENTITY컬럼을 사용하는identity_val_local() 함수를 적용하여 Lock/Latch Time을 줄였다
2)갱신 확인 처리 시간 분석에 의한 조정
Auto commit옵션이 기본으로 설정되어 SQL마다불필요한 부하가 발생하던 것을 해제하여 트랜잭션당 갱신확인 횟수를 줄였다.
4.2.2 DB2 서브시스템 성능 요소 조정
1)풀 조정
버퍼 풀(Buffer Pool), EDM풀(Environmental Descriptor Manager Pool), RID 풀(Record Identifier Pool), 정렬 풀(SORT Pool)을 시스템 자원에 맞게 조정하였다.
2)LOG 조정
출력 버퍼 크기, Active Log, Archive Log를 조정하였다.
3)WLM 적용 검증 및 조정
DB2 Work Load가 최상의 성능을 낼 수 있도록 주소공간 목표 성능을 VTAM>IRLM>DBM1>DRDA순서로 조정하였다.

4.2.3 데이터베이스 성능 요소 조정

1)물리 DB 설계 검증 및 조정

Unix DB에서 사용되는 VARCHAR는 CHAR을 사용하여 I/O성능을 향상 시키고 로우 마지막에 정의하였다. FreeSpace도 조정하여 다량의 삽입 SQL로 처리 하였다.

2)유니코드 적용 여부 검증 및 조정

웹 어플리케이션에서 코드전환 부하를 줄이기 위해 DB2 데이터를 유니코드로 저장하였다.

3)인덱스 조정

테이블 공간검색이 발생하는 경우 발생 원인을 분석하여 해결방안에 따라 처리 하였다.

4.2.4 어플리케이션 성능요소 조정

1)Java 데이터 타입 매칭 조정

DB2 Table에서 사용하는 데이터 타입과 일치시킴

2) SQL 통합 여부 검증 및 조정

SQL 구성로직을 분석하여 통합 가능한 SQL을 통합시켜 SQL 호출 수를 줄임

4.2.5 WAS 서버 성능 요소 조정

1) Dynamic Statement Cache 적용 여부 검증 및 조정

WAS에서 DB2 z/Series캐쉬(Cache)에 접근경로와최적화 결과를 저장하고 재사용 하는데 사용되는 옵션을 양쪽 서버에 모두 적용하여 SQL 호출수가 줄어 들었다.

4.2.6 JDBC 드라이버 성능 요소 조정

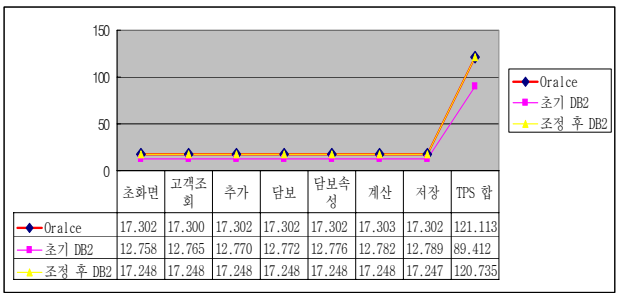
1) JDBC 드라이버 타입 조정

DB2 for z/OS는 Local Driver Type에 대해 JDBC 타입2와 JDBC타입4를 지원한다. 표 2는 두 가지 타입의 테스트 결과를 보여 주고 있으며, 결과에 따라 JDBC 타입4를 적용하였다.[12]

Driver	WAS Instance	Pool	THD	JVM Mem	Start time	End time	WAS CPU	User			TPS
								Start	Incre	Max	
Type4	2	50	50	1G	01:22	01:32	92	1	1	100	**19**
Type2	2	50	50	1G	02:11	02:22	91	1	1	100	**10**

표 2 JDBC 타입4와 타입2의 테스트 결과

4.3 최종 성능 개선 결과



	초화면	고객조회	추가	담보	담보속성	계산	저장	TPS 합
Oralce	17.302	17.300	17.302	17.302	17.302	17.303	17.302	121.113
초기 DB2	12.758	12.765	12.770	12.772	12.776	12.782	12.789	89.412
조정 후 DB2	17.248	17.248	17.248	17.248	17.248	17.248	17.247	120.735

2차에 걸쳐 성능 조정 요소를 반영한 결과 TPS가 초기 89.4에서 120.7로 향상되었으며 그 중 성능향상효과가 큰 주요 요소는 SQL 통합, 채번 Logic에서 IDENTITY Column 적용, 유니코드 적용 부분이다.

5. 결론 및 향후 연구 과제

본 논문에서는 J2EE 환경에서 IBM DB2데이터베이스를 적용한 시스템에서 발생할 수 있는 성능 저하의 문제 원인을 찾기 위한 접근방법을 제시하고 이를 사용한 성능 조정 성공사례를 제시하였다. 그러나 사례 연구를 위해 구축된 시스템이 배치처리 시스템과 전사 시스템을 대상으로 검증하지 못했고 스트레스 테스트 시 실제 사용자 수를 감안하지 못하고 테스트 한 것과 WAS서버와 DB서버의 Location이 달라서 발생할 수 있는 네트워크 딜레이를 실제 감안하지 못한 것이 한계점이라고 할 수 있다.

향후 연구에서는 본 논문의 미흡한 부분인 전사 업무 및 배치처리 시스템 관점에서 구체적으로 규명되어져야 할 것이다.

참고문헌

[1] Michael Nash, “JAVA Frameworks and Components”, CAMBRIDGE, 2003

[2] PLATINUM Technology, DB2 SQL Performance and Tuning

[3] IBM, Mercury Interactive

[4] IBM Framework for e-business

[5] IBM, LoadRunner Vuser 스크립터 만들기(V 7.8)

[6] DB2 for z/OS and OS/390: Ready for Java

[7] IBM, DB2 Administration Guide Vol 1

[8] DB2 Application Programming Guide and Reference for Java

[9] DB2 for z/OS Stored Procedures : Through the CALL and Beyond

[10] BEA, JDBC Data Sources

[11] 박기수, DB2를 이용한 OLTP에서 시스템 성능확보 및 유지방안 연구, 1998

[12] BEA, Weblogic Type 4 JDBC Drivers