

# 웹 데이터베이스 서버 성능개선을 위한 램의 활용

이원조 · 이단영\* · 고재진\*\*

울산대학교 대학원 컴퓨터정보통신공학과

wjlee@mail.ulsan-c.ac.kr

## Using of RAM for Performance Tuning in Web Database Server

Won-Jo Lee · Dan-Young Lee\* · Jae-Jin Koh\*\*

School of Computer Engineering and Information Technology,

University of Ulsan

### 요 약

인터넷 E-Business를 위한 ASP를 이용한 웹사이트 구축이 급속도로 증가하는 추세로 대화형 서비스를 제공하기 위해 데이터베이스인 MS SQL Server의 적용이 많아졌다. 따라서 본 연구에서는 웹사이트 구축시 데이터베이스의 성능개선을 위한 임시테이블의 램 활용기법에 대한 효율적인 모델을 제안하고 성능을 평가하였다. 이와 같은 결과로 웹서버의 성능개선에 임시테이블의 램 활용에 대한 기법이 유익하게 활용될 것이다.

## 1. 서 론

최근 인터넷의 이용자가 폭발적으로 증가하여 개인이나 기업들의 웹 사이트 구축이 활발하게 이루어지고 있으며, 이로 인하여 네트워크와 서버의 부하가 가중되어 성능개선을 위한 여러 가지 기법이 적용되고 있다. 이러한 기법들을 보면 분산시스템, 클러스터링, 로드 밸런싱, 캐싱서버 등의 기법들이 있다. 그러나 본 연구에서는 마이크로소프트사의 관계형 데이터베이스인 SQL Server6.5의 Tempdb의 램 영역사용에 대한 여러 가지 기술자료들이 있으나, 데이터베이스 적용에 대한 유용성 평가가 없어 실무에 많이 사용되고 있지 않고 있다. 따라서 아직 많은 서버들이 SQL Server6.5를 사용하고 있는 현실을 고려할 때 옵션의 변경이나 적은 투자로 서버의 성능을 개선할 수 있는 기법을 제안 하고자 한다.[9][10]

## 2. 관련연구

### 2.1 서버 부하의 특성

정보시스템에서 부하의 발생요소는 크게 O/S의 자원할당 제어기능과 CPU 그리고 메모리, 디스크, 네트워크 그리고 응용프로그램 등으로 설명될 수 있다. 여기서 부하를 줄이는 방법과 부하에 영향을 주는 자원의 확장으로 성능을 개선할 수 있다.

부하방지를 위한 시스템의 분산은 크게 두 가지 측면으로

보는데 하나는 네트워크의 부하를 분산하는 일과 수행하는 업무를 분산하는 일이다. 정보시스템에서 부하 분산시 작업의 특성을 잘 고려해야 한다. 작은 규모로 분산할 경우에는 관리적인 측면의 비효율성을 가져온다. 서버시스템 내에서의 부하요소는 O/S, CPU, RAM, Disk, BUS, I/O Controller, Disk Controller, Network등을 이해하고 점검해야 한다. 그리고 CPU, RAM, I/O Controller는 시스템 버스에 종속되어 있으므로 시스템 버스의 구조는 시스템의 성능에 중요한 영향을 미친다. 따라서 연산이 많은 시스템의 경우에는 이 시스템 버스에서 부하가 집중적으로 발생하게 된다.[1][6][8]

### 2.2 Load Balancing

Load Balancing은 웹사이트의 성능을 개선하여 사용자들에게 최적의 성능을 제공하기 위한 가용성과 안정성을 추구한다. 그간 웹사이트의 성능을 개선하기 위해서 시스템이나 네트워크를 물리적으로 증대시키는 방법이 주로 사용되어 왔으나 부하의 증가에 따른 대응에 막대한 비용이 발생하여 부하관리에 대한 기법들을 적용하는 것이다. 이러한 부하를 효율적으로 분산시키는 기법이 Load Balancing이다. Load Balancing Tool들을 보면 소프트웨어 기반의 제품, 라우터 기반의 제품, 로드 밸런싱 지원 스위칭 HUB등으로 나눌 수 있다.[11]

2.3. ASP(Active Server Pages)

ASP는 하나 이상의 스크립트 (작은 내장 프로그램)를 담은 HTML 페이지로 사용자에게 보내지기 전에 마이크로소프트 웹 서버에서 처리된다. ASP는 사용자에게 웹 이지를 만들어 제공하기 위해 서버에서 관련 프로그램이 모두 수행되는 SSI나 CGI 애플리케이션과 꽤 유사하다. 대개, 서버의 웹 이지 스크립트는 사용자의 요구에 따라 데이터베이스에 접속한 다음, 그 결과를 토대로 하여 요구자에게 보내기 전에 순간적으로 웹 페이지를 만들거나 특화하게 된다.[5][9][10]

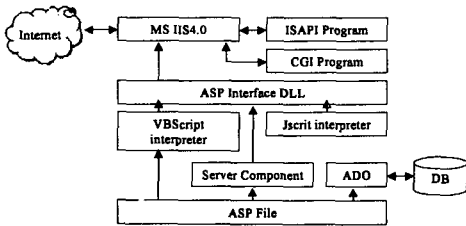


그림 1. ASP, ISAPI, CGI 연동 Configuration

2.4 임시 테이블

임시 테이블은 사용자가 제거하지 않아도 SQL서버가 트랜잭션이 끝나면 자동적으로 삭제된다. 이 테이블은 Tempdb 데이터베이스에 생성된다. 이러한 임시 테이블에는 2가지 종류가 있는데 전역 임시 테이블은 ##로 테이블 명이 시작되고 연결된 모든 사람에게 보이고 다수의 사용자에게 공유되며, 임시적으로 쓰지 않은 데이터에 적합하다. 그리고 로컬 임시 테이블은 #로 테이블 명이 시작되고 현재 연결상태에서만 보인다. 이러한 임시 테이블은 스토어드 프로시저 안에서 쓰인 임시 테이블은 프로시저의 수행이 종료되면 자동으로 소멸되는 성격을 갖고 있다.[1][2]

3. 제안모델

3.1 Tempdb를 RAM에 올리는 방법

마이크로소프트의 SQL Server6.5에서 Tempdb는 기본적으로 Master Disk Device에 2MB로 생성된다. 이것은 설치 완료 후 다른 디바이스로 확장할 수 있다.(alter database command), 이 옵션은 작업 테이블, 임시 테이블, 정렬 연산과 관련된 페이지 접근 연산의 속도를 증가시키는 옵션이다. 하지만 주의해서 사용해야 하는데 그 이유는 SQL서버가 일반적으로 캐시를 사용자 보다 더 잘 관리하므로 용량이 큰 메모리를 가진 서버에서 사용되어야 효과적이다. 그러면 어떻게 하면 다량의 데이터를 효율적으로 처리하여 실행속도를 향상시키느냐 하는 것이다. Tempdb를 램에 올리는 방법은 다음과 같다.[3][4]

- MS SQL Server6.5의 Enterprise Manager를 실행하여 Server Configuration/Options을 선택하고 Configuration을 선택하고 "Tempdb in ram(MB)"가 "Current" 0MB인데

"Maximum" 2044MB로 설정되어 있다.

- 전항과 동일한 방법으로 "Tempdb in ram(MB)"의 "Current"에 20MB를 할당하고 SQL Server6.5의 서비스를 종료한 후 재 기동한다.

- 이제 Tempdb의 정보를 확인해 보면 확장되었음을 알 수 있다.

3.2 효율적인 Tempdb의 램 설정 기법

우선 기존 서버 RAM의 용량을 확장하지 않고 Tempdb를 RAM에 올리는 것에는 주의를 요한다. 왜냐하면 SQL Server6.5는 자체적으로 캐시에 Tempdb의 테이블을 저장하고 있기 때문에 데이터가 중복되어 성능이 저하될 수 있다. 따라서 기존의 시스템이 64MB의 RAM을 내장하고 있을 경우에는

Available Memory (24MB)	Application Load(10MB)
Assign tempdb(20MB)	Application Load(10MB)

그림 2. 기존 서버의 메모리 할당(20MB)

그림2와 같은 상태가 되어 SQL 서버 캐싱시 중복으로 인하여 성능이 저하된다.

Available Memory (44MB)	Application Load(10MB)
	Application Load(10MB)
Extended Memory tempdb(20MB)	

그림 3. 기존 서버의 RAM을 확장

따라서 그림3과 같이 기존 서버의 RAM을 20MB 이상으로 확장하고 Tempdb를 확장된 RAM 영역을 사용하게 될 경우에 실행 속도를 향상시킬 수 있으며, 다음과 같은 특정 상황에서 사용되어야 성능의 향상을 기대할 수 있다. 예를 들면 다음과 같다.[4][5]

- 데이터베이스가 MS SQL Server6.5를 사용할 경우
- 여러 개의 응용프로그램들이 Tempdb를 동시에 사용할 경우
- 처리할 데이터량이 Tempdb의 메모리 구성에 적합할

경우

- SQL서버의 RAM이 128MB 이상일 경우
- SQL서버가 캐시를 사용하는 비율이 낮을 경우

#### 4. 성능평가

##### 4.1. 실험에 사용된 시스템 사양

- Hardware System : CPU Pentium 350, RAM 64MB
- Operating System : Microsoft Windows NT Server4.0
- Database : Microsoft SQL Server6.5

##### 4.2 실험방법

실험의 결과는 시스템의 사양과 운영체제 버전의 차이, 테이블의 구조, 데이터의 량에 따라 다를 수 있다. 먼저 아래의 Script 문을 ISQL/W에서 실행하면 다음과 같은 적용 전 수행결과를 얻을 수 있다.

```
set statistics time on
SELECT * FROM sales
- 수행결과(적용 전)
SQL Server Parse and Compile Time:
cpu time = 60 ms.
```

이제 Tempdb in RAM을 20MB로 설정하고 SQL Server를 Restart 한 후 아래의 Script 문을 ISQL/W에서 실행하면 다음과 같은 적용 후 수행결과를 얻을 수 있다.

```
set statistics time on
SELECT * FROM sales
- 수행결과(적용 후)
SQL Server Parse and Compile Time:
cpu time = 28 ms.
```

##### 4.3 실험결과

실험결과는 서버의 상태에 따라 다를 수 있으나, 본 실험에서는 그림4과 같이 "Tempdb in RAM" 적용시 약 2배의 성능 향상을 보였다.

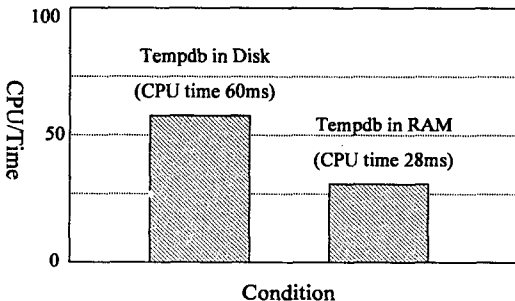


그림 4. 성능비교 결과(Tempdb in RAM)

#### 4.4. 실험 시 주의사항

이러한 데이터베이스의 성능 향상기법은 어느 경우에 서나 적용할 수 있는 것이 아니며, 적용시 반드시 컴퓨터 하드웨어 램이 충분하거나 추가하여 적절히 활용해야 한다. 만약 "Tempdb in RAM"에 컴퓨터의 하드웨어 램보다 많은 램 크기를 설정하면 SQL Server6.5가 기동되지 않는다. 이럴 경우에는 MS SQL Server6.5 Setup에서 "Rebuild Master Database"를 수행하면 Master Database가 재구성되어 Tempdb in RAM 설정이 복원된다.

#### 5. 결 론

본 논문에서는 과거에 제안되었던 MS SQL Server6.5의 "Tempdb in RAM" 기법의 적용에 대한 논란이 있어왔다. 그러나 아직 SQL Server6.5로 기 구축된 많은 웹 서버들이 사용되고 있으며, 성능개선의 시점에 놓여있다. 실험의 결과와 같이 서버 상태에 따른 효율 정도는 상이하나 간단하고 경제적으로 서버의 성능을 2배정도 개선할 수 있는 기법으로 평가됩니다. 따라서 향후 과제는 다양한 서버 환경에서 실험을 반복하여 최적한 서버의 성능을 향상시킬 수 있는 연구가 필요하다.

#### 6. 참고문헌

- [1] 이재훈, SQL 서버6.5 언리쉬드, 도서출판대림, 1977
- [2] 정원혁, Microsoft SQL Server7.0, 도서출판대림, 1999
- [3] 월간 마이크로소프트, Microsoft Developer Journal 3, 1998
- [4] 월간 마이크로소프트, Microsoft Developer Journal 12, 1997
- [5] Terms Korea and whatis.com Inc.
- [6] 문진용, 인터넷 케현에서의 대체 알고리즘의 설계 및 성능 평가, 한국정보과학회, 2000
- [7] 정훈진, 클러스터 시스템에서의 동적 여분 부하 균등화, 한국정보과학회, 2000
- [8] 김남기, 데이터 기반 레이블 스위칭을 위한 캐쉬 대체 기법, 한국정보과학회, 2000
- [9] 깨비메일정보, ASP로 웹 서버 구축하기, 기진연구소, 2000
- [10] 오해석, ASP를 이용한 웹 사이트 구축, 홍릉과학, 2000
- [11] 이원조, 캐싱서버를 이용한 네트워크 최적화에 관한 연구, 울산과학대학, 제27권, 2000.