ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

네트워크 디스크를 공유하여 SQL 서버의 대용량 스토리지 확보 방법 - Amazon EC2 Windows 환경에서 -☆

A method of Securing Mass Storage for SQL Server by Sharing Network Disks - on the Amazon EC2 Windows Environments -

강 성 욱[1] 최 종 선[1] 최 재 영[1*]
Sungwook Kang Jungsun Choi Jaeyoung Choi

요 약

클라우드 컴퓨팅 환경에서 사용자는 중앙처리장치, 메모리, 네트워크, 저장 장치 등과 같은 인프라를 IaaS(Infrastructure as a Service) 서비스로 제공받을 수 있다. 저장 장치의 경우 서비스 제공자가 제공하는 물리적인 자원들의 인스턴스가 제공하는 저장 용량이 제한되어 있으므로, SQL 서버가 사용할 수 있는 최대 저장 용량을 지원할 수 없다. 본 논문에서는 한정된 용량을 가진 인스턴스들의 네트워크 디스크를 공유하여 SQL 서버에서 사용할 수 있는 대용량의 저장소를 확보하기 위한 방법을 제안한다. 실험을 통해 아마존 EC2 윈도우즈 환경에서 아마존 EBS 볼륨을 사용하는 하나의 인스턴스가 사용할 수 있는 최대 저장 용량을 초과하는 대용량 저장 공간을 확보할 수 있으며, SQL Server를 운영하는 아마존 클라우드 환경에서 디스크 용량 및 성능을 증가시켜 전체적인 SQL Server의 성능을 향상시킬 수 있음을 확인하였다.

☞ 주제어 : 클라우드 저장장치, SQL 서버, 네트워크 디스크 공유, 아마존 EC2, 아마존 EBS

ABSTRACT

Users are provided infrastructure such as CPU, memory, network, and storage as IaaS (Infrastructure as a Service) service on cloud computing environments. However storage instances cannot support the maximum storage capacity that SQL servers can use, because the capacity of instances provided by service providers is usually limited. In this paper, we propose a method of securing mass storage capacity for SQL servers by sharing network disks with limited storage capacity. We confirmed through experiments that it is possible to secure mass storage capacity, which exceeds the maximum storage capacity provided by an instance with Amazon EBS on Amazon EC2 Windows environments, and it is possible to improve the overall performance of the SQL servers by increasing the disk capacity and performance.

☞ keyword : Cloud Storage, SQL Server, Network disk sharing, Amazon EC2, Amazon EBS

1. 서 론

클라우드 컴퓨팅 환경에서 IaaS와 같은 서비스는 사용한 만큼만 과금하는 특징 때문에 초기 도입 비용이 저렴하며, 사용자는 사용량에 따라 유연하게 하드웨어를 확장할 수 있다. 클라우드 서비스는 단일 인프라에서 발생할 수 있는 장애(Single Point of Failure, SPOF)에 대비하여 고가용성(High Availability, HA)을 추구하고, 이를 바탕으로 보안 및 안정성을 극대화시킬 수 있다. 또한 클라우드 서비스는 내부적으로 데이터를 중복 저장하여 물리적인 장애에도 유연하게 대처할 수 있고, 시스템들은 매우 빠른 네트워크로 연결되어 초고속 데이터 통신이 가능하도록 설계되어 있다 [1][9][11]. 이와 같은 특징들은 클라우드 컴퓨팅 환경에서 대용량의 Scale-Out 환경을 유연하게 구성할 수 있도록 한다.

그러나 서비스 제공사에서 지원하는 인스턴스의 최대 스토리지 용량은 SQL 서버에서 사용할 수 있는 최대 스토리지 용량보다 작기 때문에, SQL 서버에서 사용할 수 있는 최대 스토리지 용량을 지원하지 못하는 문제가 발생한다. 본 논문은 클라우드 컴퓨팅 환경의 초고속 네트

[1] School of Computer Science and Engineering, Soongsil University, Seoul, 06978, Korea
[*] Corresponding author (choi@ssu.ac.kr)
[Received 12 August 2015, Reviewed 10 September 2015(R2 4 November 20115), Accepted 12 February 2016]
☆ 본 논문은 강성욱(2015년)의 석사학위 논문을 수정 및 보완한 논문입니다.
☆ 본 연구는 2015년도 정부(미래창조과학부)의 재원으로 한국연구재단의 차세대정보컴퓨팅기술개발사업의 지원(NRF-2015M3C4A7065662)으로 수행되었습니다.

http://dx.doi.org/10.7472/jksii.2016.17.2.01

워크 및 SQL 서버의 파일 그룹이 가진 특성을 활용하여 SQL 서버에서 사용할 수 있는 대용량의 스토리지를 확보하기 위한 방법을 제안한다.

클라우드를 서비스하는 업체에 따라 자사의 플랫폼에 대한 특징 및 서비스 내용에 차이가 있으므로, 본 논문에서는 IaaS 플랫폼 중 하나인 Amazon EC2를 사용하였다. 인스턴스의 최대 스토리지 용량을 확보하는 것과 같이 비용이 매우 크게 요구되는 테스트에서는 실제 테스트 환경을 구성할 수 없기 때문에 동일한 개념을 적용한 소규모 환경으로 실험하였다. 아마존 클라우드 환경에서 EBS 디스크를 제안하는 방법으로 네트워크 디스크 공유하여 확장하였을 때, 확장된 DBMS 저장소의 성능을 측정하여, 기존의 클라우드 환경에서 하나의 인스턴스가 제공하는 제한된 물리적인 스토리지 용량의 한계를 극복할 수 있음을 보였다.

본 논문의 구성은 다음과 같다. 1장인 서론에서는 연구 배경 및 목적을 제시하고, 2장에서는 Amazon EBS 볼륨 스토리지 특성 및 SQL 서버 특성을 설명한다. 3장에서는 클라우드 환경에서 SQL 서버가 사용할 수 있는 스토리지 확보 방법을 제시하고, 4장에서는 제안한 방법을 실험하고 분석한다. 5장에서는 본 논문의 결과를 다룬다.

2. 관련연구

2.1 Amazon EBS 볼륨 스토리지 특성

인스턴스에서 지원할 수 있는 최대 볼륨 수는 운영체제에 따라 다르다. 하나의 인스턴스에서 사용할 수 있는 물리적인 최대 볼륨수가 고정되어 있기 때문에 인스턴스에서 사용할 수 있는 스토리지 최대 용량은 (최대 볼륨 수) x (단일 볼륨 용량)으로 계산할 수 있다[2, 3].

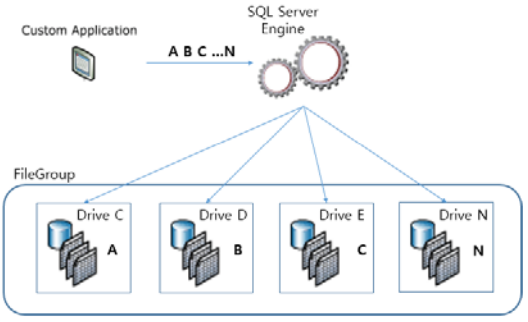
(표 1) 아마존 인스턴스 최대 볼륨
(Table 1) Maximum Volume of Amazon Instance

Windows		Linux
Driver	Volume limit	최대 40개 이상의 볼륨 연결을 지원하지만, 40개 이상인 경우는 보장되지 않으며 부팅 오류가 발생할 수 있음
AWS PV	26	
Citrix PV	26	
Red Hat PV	17	

네트워크 디스크 공유 방법을 통해 최대 볼륨으로 구성된 여러 인스턴스들의 디스크를 SQL Server 파일 그룹에서 각 인스턴스의 공유된 디스크로 연결할 경우, 연결된 디스크를 인스턴스의 물리적인 최대 볼륨 용량을 초과하여 SQL 서버의 데이터 파일 저장소로 사용할 수 있다.

2.2 SQL 서버 특징

SQL 서버의 파일 그룹은, 다수의 데이터 파일이 동일한 파일 그룹에 있는 경우, 자체적으로 데이터를 분산하여 RAID 0과 같은 효과를 낼 수 있는 특징을 갖는다[4]. 그림 1에서 볼 수 있듯이, 데이터베이스 엔진은 SQL 서버의 데이터 파일이 동일한 파일 그룹에서 여러 개의 데이터 파일로 분산되어 있을 때 사용자로부터 입력된 데이터를 분산하여 처리한다. 이때 I/O 분산을 통한 성능을 향상시키기 위해 데이터 파일을 저장하는 각각의 디스크는 물리적으로 분리되어 있어야 한다.



(그림 1) SQL 서버 파일 그룹 특성
(Figure 1) Property of SQL Server File Group

SQL Server 2012에서 데이터베이스가 사용할 수 있는 최대 디스크의 용량은 표 2와 같이 524,272 TB이며, 단일 데이터 파일 사이즈는 최대 16 TB이다. 데이터 파일은 최대 32,767개까지 구성할 수 있다. 윈도우즈에서 인식할 수 있는 최대 용량은 18 EB[1]이며, 가상화된 윈도우는 최대 64 TB까지 사용이 가능하다[5].

(표 2) SQL Server 2012의 최대 지원 파일 크기
(Table 2) Maximum File Size of SQL Server 2012

	Max Size (x86)	Max Size (x64)
Database Size	524,272 TB	524,272 TB
Data File Size	16 TB	16 TB
Log File Size	2 TB	2 TB
Data File Number	32,767	32,767

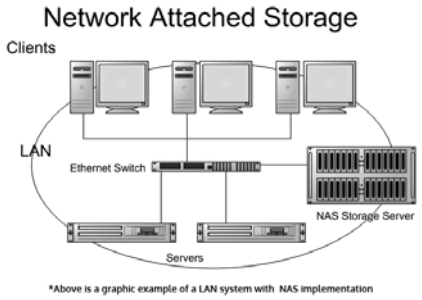
[1] EB = Exabyte (10^{18})

2.3 HDFS 시스템과 NAS

클러스터 시스템은 여러 개의 컴퓨터들을 네트워크로 묶어 병렬처리 시스템의 노드로 사용함으로써 유휴 자원들을 유용하게 사용하는 것이다. 과거에는 네트워크의 속도가 느려서 통신에 대한 높은 오버헤드를 부담하였으나, 최근에는 네트워크가 고속화되고 간략화된 프로토콜이 출현함에 따라 성능 및 오버헤드 부분의 많은 문제점이 해결되었다. 클러스터 시스템에서는 I/O 연산이 전체 시스템의 병목을 방지하기 위해 데이터를 분산하여 저장할 필요가 있으며, 이러한 서비스를 지원하는 단일 I/O 공간 SIOS[2]를 요구하게 된다.

클러스터 환경에서 대표적인 데이터 분산 처리 시스템은 HDFS[3]이다. HDFS는 네트워크에 네임노드와 데이터노드를 생성하여 데이터를 분산하여 처리한다[12]. HDFS의 특징은 데이터 노드가 공유되며 복제된다. 데이터간의 커플링이 없어 특정 노드가 손실되어도 데이터는 유지된다. 하지만 RDBMS에서 제공하는 JOIN이나 데이터 무결성에 대한 것은 보장되지 않는다.

NAS는 컴퓨터 네트워크에 연결된 파일 수준의 컴퓨터 기억장치이며 클러스터 NAS는 여러 대의 서버에서 동시에 실행되는 분산 파일 시스템을 이용한 NAS이다. 클러스터 NAS의 특징은 데이터와 메타데이터를 스트라이핑하여 분산시킬 수 있다.



본 논문에서는 HDFS 특성과 유사한 효과를 나타내기 위해 SQL Server를 이용하여 데이터 분산 및 스트라이핑 기법을 구현하였으며 RDBMS의 특징인 JOIN 및 무결성을 보장하였다. NAS의 특성을 구현하기 위해 클라우드 환경에서 Windows 폴더 공유기법을 사용하여 여러 시스템의 디스크를 공유하여 성능 및 용량을 확장할 수 있도록 하였다. 이 방법을 사용함으로써 기존의 클라우드에서 제공하는 저장소의 한계를 초과하여 SQL Server에서 사용할 수 있다.

[2] SIOS : Single I/O Space

[3] HDFS : Hadoop Distribute File System

3. 제안하는 대용량 스토리지 확장 방법

3.1 시스템 구성

제안하는 시스템은 IaaS 플랫폼 형태의 Amazon EC2 서비스에서 구성하였다. 표 3은 대용량 스트로지 확장 방법을 적용하기 위한 시스템의 HW 및 SW의 명세를 나타낸다[6, 7].

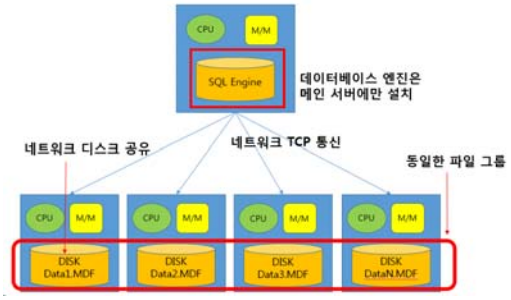
(표 3) 제안하는 시스템의 하드웨어 및 소트트웨어 명세
(Table 3) Hardware and Software specification of the Proposed System

	Installed with SQL Server	Server with Shared Disk
Model	c4.8xlarge	m3.xlarge
CPU	36 Core	4 Core
Memory	60 GB	15 GB
Disk	Amazon EBS Volumn (100 GB, 300 IOPS)	Amazon EBS Volumn (100 GB, 300 IOPS)
Network	10 Gbit	1 Gbit
OS	Windows Server 2012 R2	Windows Server 2012 R2
DB	SQL Server 2012 SP2	

표 3에서 보듯이 Amazon EC2의 Windows Server 2012 R2 환경에서 실험하였다. 시스템은 10 Gbit의 네트워크를 지원하는 c4.8xlarge와 1 Gbit의 네트워을 지원하는 m3.xlarge의 2가지 인스턴스를 이용하였다. SQL 서버가 구동될 서버는 CPU, 디스크, 메모리, 네트워크 등의 하드웨어로 인한 병목이 최대한 발생하지 않도록 구성한다. 네트워크 공유 디스크로 제공될 서버는 상대적으로 낮은 성능의 서버를 준비한다.

Amazon EBS 볼륨은 1 GB당 3 IOPS를 제공한다. 각 인스턴스당 Amazon EBS 볼륨은 동일한 크기 및 성능으로 생성한다. 생성된 드라이브는 네트워크 디스크 공유를 설정하여 TCP/IP로 접속할 수 있도록 한다. 본 논문에서는 총 5대의 시스템으로 실험을 진행하기 때문에

Amazon EBS 볼륨을 생성할 때 볼륨의 크기에 따라 제공되는 성능이 다르므로 반드시 동일한 Amazon EBS 사양을 준비한다.



(그림 2) 제안하는 시스템 구성
(Figure 2) Configuration of the Proposed System

그림 2는 제안하는 시스템의 구성으로 SQL 서버는 여러 인스턴스 중 하나의 인스턴스에만 설치한다. 나머지 다른 인스턴스들은 네트워크로 디스크를 공유할 수 있는 용도로 사용된다. 공유된 네트워크 디스크에 데이터 파일을 생성할 때 반드시 데이터 파일은 동일한 파일 그룹을 사용해야 한다.

3.2 SQL 서버 메모리 설정

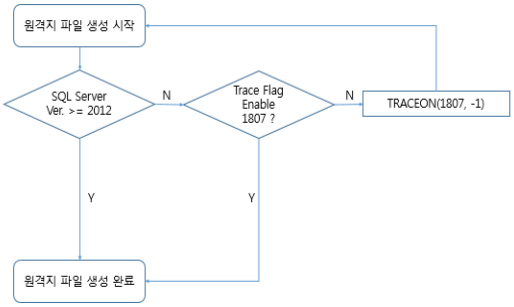
SQL Server가 구동되는 서버의 물리적인 여유 메모리가 테스트에 사용되는 데이터의 크기보다 작은 경우, 메모리에서 병목 현상이 발생하기 때문에 디스크의 읽기 성능을 정확히 측정할 수 없다. 이러한 이유로 SQL 서버에서 사용할 수 있는 메모리를 여유있게 할당한다. 테스트 환경에서는 SQL 서버의 가용메모리를 20 GB로 설정하였다[8].

```
SP_CONFIGURE 'SHOW ADVANCED OPTIONS', 1
GO
RECONFIGURE
GO
SP_CONFIGURE 'MAX SERVER MEMORY (MB)',
'20480'
GO
RECONFIGURE
GO
```

(그림 3) SQL 서버 메모리 설정
(Figure 3) Setting of SQL Server Memory

3.3 SQL 서버 설정

공유된 네트워크 디스크에 데이터 파일을 생성하기 위해서는 SQL Server Trace Flag 1807이 활성화되어 있어야 한다[10]. 그림 4와 같이 SQL Server 2012 이전 버전까지는 강제적으로 1807을 활성화시켜야 하며, SQL Server 2012부터는 기본적으로 네트워크 위치의 데이터 파일 생성을 지원한다.



(그림 4) 원격 데이터 파일 생성 프로세스
(Figure 4) Remote Creation Process of Data File

네트워크 위치에 데이터 파일 생성이 실패할 경우를 대비하여 네트워크 디스크를 사용하기 위해 그림 5와 같이 Trace Flag 1807을 활성화시킨다.

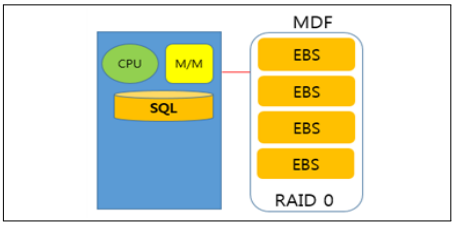
```
BEGIN TRY
        DROP TABLE #TRACESTATUS
END TRY BEGIN CATCH END CATCH

CREATE TABLE #TRACESTATUS (
        [TRACEFLAG] INT,
        [STATUS] INT,
        [GLOBAL] INT,
        [SESSION] INT
)

INSERT #TRACESTATUS
EXEC ('DBCC TRACESTATUS')

IF EXISTS (SELECT * FROM #TRACESTATUS WHERE
TRACEFLAG = 1807)
BEGIN
        RETURN
END
ELSE
BEGIN
        DBCC TRACEON(1807, -1)
END
```

(그림 5) Trace Flag 1807 설정
(Figure 5) Setting of Trace Flag 1807

3.4 워크로드 실험 구성

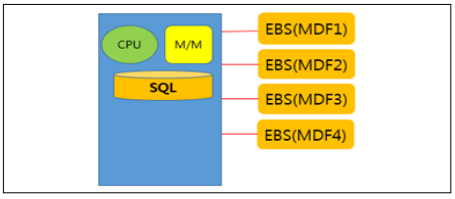
워크로드 실험의 진행을 위해 그림 6과 같이 세 가지 방법으로 실험 환경을 구성하였다.

그림 6은 3가지 방법으로 구성된 실험 환경을 보여주고 있다. 그림 6 (a)의 실험 1은 인스턴스에 Amazon EBS 볼륨을 직접 연결한 RAID 0으로 구성하여 성능을 측정한다. 그림 6 (b)의 실험 2는 인스턴스에 연결된 Amazon EBS 볼륨 각각의 디스크에 데이터 파일을 분산하여 성능을 측정한다. 그림 6 (c)의 실험 3은 다른 인스턴스에 연결된 각각의 Amazon EBS 볼륨을 네트워크 디스크 공유로 연결하여 성능을 측정한다. 각 실험에서 분산된 데이터 파일은 반드시 동일한 파일 그룹에 속해있어야 한다.



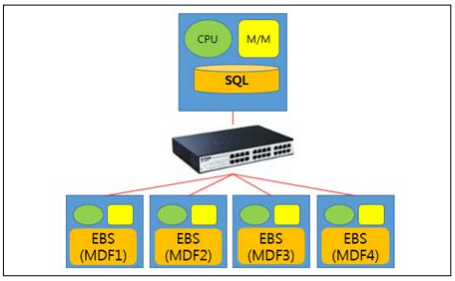
(a) 실험 1 - RAID 0으로 직접 연결

(a) Experiment 1 - Connect directly with RAID 0



(b) 실험 2 - 각 디스크를 직접 연결

(b) Experiment 2 - Connect each disk direclty



(c) 실험 3 - 네트워크 디스크 공유로 연결

(c) Experiment 3 - Connect by sharing network disk

(그림 6) 실험 환경 구성

(Figure 6) Configuration of Test Environments

4. 실험 및 성능평가

실험 방법은 SQL 서버 메모리에 데이터가 적재되기까지의 시간을 측정하여 결과 값으로 사용한다. 결과의 신뢰성을 높이기 위해 각 실험 방법에 따라 5번의 테스트를 진행하였으며, 이 결과들의 평균값을 산출하여 최종 결과를 도출한다. 측정되는 성능 값은 순수 스토리지의 처리 능력이 아닌 SQL 서버에서 처리하는 스토리지의 성능이다. 성능 측정에 사용된 단위는 실제 데이터 읽기를 완료한 시간을 초로 계산하였다. 로그파일은 분산되어 있어도 병렬처리의 효과를 나타내지 못하기 때문에, 모든 테스트에서 가장 빠른 인스턴스 스토어에 위치하여 로그 파일의 병목을 최소화할 수 있도록 구성한다.

```
IF EXISTS (SELECT * FROM SYSOBJECTS WHERE NAME =
'TEST')
      DROP TABLE TEST
GO

IF EXISTS (SELECT * FROM SYSOBJECTS WHERE NAME =
'TEST')
      DROP TABLE TEST
GO

CREATE TABLE TEST (
X INT NOT NULL,
Y CHAR(896) NOT NULL DEFAULT (''),
Z CHAR(120) NOT NULL DEFAULT('')
)
GO

INSERT TEST (X)
SELECT R FROM (
      SELECT
      ROW_NUMBER() OVER (ORDER BY (SELECT 1)) R
      FROM
      MASTER..SPT_VALUES A, MASTER..SPT_VALUES B
) P
WHERE R <= 4000000
GO
```

(그림 7) 실험 데이터 생성

(Figure 7) Data Creation for Tests

그림 7은 실험에 사용할 테이블 및 데이터를 생성하기 위한 SQL을 나타낸다. 테이블 정의 및 데이터 환경은 모든 실험에 공통으로 적용된다. 한 ROW당 1,020 byte의 크기를 가진 데이터 4,000,000건을 추가하여 약 4 GB의 데이터를 생성한다.

```
DECLARE @X INT, @Y CHAR(896), @Z CHAR(120)
SELECT @X=X, @Y=Y, @Z=Z FROM TEST
```

(그림 8) 성능 측정 코드
(Figure 8) Program Code for Performance Measurement

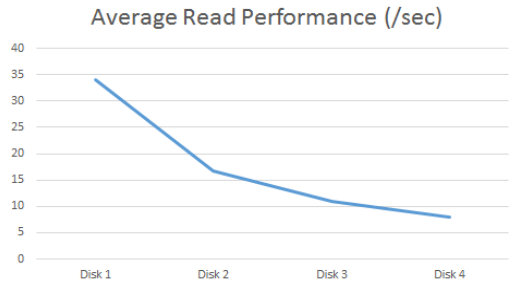
그림 8은 성능 측정시 결과에 대한 신뢰성 및 결과 왜곡을 방지하기 위해 읽은 데이터에 대해서 UI 또는 파일 출력이 아닌 메모리에 적재하여 불필요한 리소스 사용 및 병목을 제거하기 위한 프로그래밍 코드이다. DECLARE 명령어는 변수를 선언하는 부분이고, @X, @Y, @Z는 변수명이며 그 뒤에 데이터 타입을 정의한다. SELECT 구문은 데이터를 조회하는 명령이며 조회 데이터 결과는 선언부에서 정의한 변수에 저장된다.

그림 9 (a)에 있는 실험 1의 결과를 살펴보면 디스크 1개를 사용할 때보다 다수의 디스크를 RAID 0으로 구성하였을 때 사용하는 디스크의 갯수에 비례해서 성능이 향상되었다. 이는 각 디스크의 제한된 성능을 RAID 0을 구성함으로써 여러 디스크 성능을 합하여 사용하는 효과를 나타낸다.

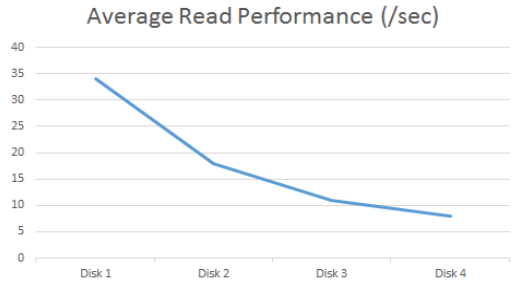
그림 9 (b)에 있는 실험 2의 결과는 실험 1과 유사한 결과를 나타내었다. 실험 2의 결과로 SQL 서버에서는 동일한 파일 그룹에 데이터파일을 분산하였을 때 RAID 0과 같은 성능을 나타낸다는 것을 확인하였다[4].

그림 9 (c)에 있는 실험 3의 결과는 성능 향상의 증가율은 실험 1 및 실험 2와 유사한 결과를 나타내었다. 하지만 실제 읽기에 대한 시간은 실험 1 및 실험 2 보다 2배 이상 오래 걸렸다. 실험 3에서는 데이터 파일의 위치가 네트워크로 공유된 디스크 폴더에 존재하고, 이 경우에는 디스크의 성능뿐만 아니라 네트워크의 상태가 실험에 영향을 미치게 된다. 따라서 실제 사용할 수 있는 네트워크의 대역폭이 제한되기 때문에, 물리적인 디스크의 최대 처리량을 사용할 수 없었다[7].

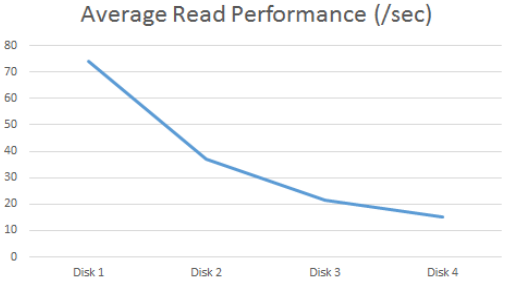
그림 10는 각 인스턴스의 물리적인 디스크의 사용량을 Byte로 측정한 값이다. 네트워크는 초당 전송하는 패킷의 양을 Mbps로 측정하였다. 그림 10에서 보여지는 그래프 세로축의 비율이 다른 이유는, 성능 모니터의 Y축이 자동적으로 스케일링되어 나타나고 있어서 순간적인 네트워크의 스파이크 현상으로 축 기준이 변화되기 때문이다. 정확한 수치는 박스로 표시된 부분으로 확인할 수 있다.



(a) 실험 1. 평균 읽기 성능
(a) Experiment 1. performance of average read

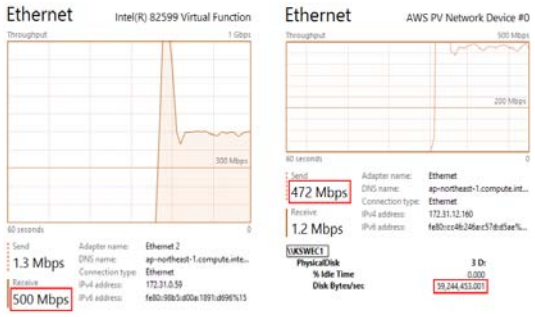


(b) 실험 2. 평균 읽기 성능
(b) Experiment 2. performance of average read

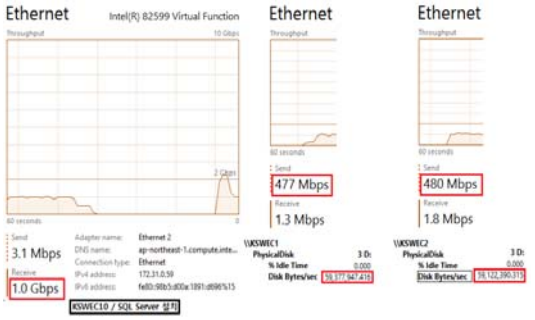


(c) 실험 3. 평균 읽기 성능
(c) Experiment 3. performance of average read

(그림 9) 읽기 성능 결과
(Figure 9) Performance of Read



(a) 공유 디스크 1개를 사용하였을 때
(a) using 1 Shared Disk



(b) 공유 디스크 2개를 사용하였을 때
(b) using 2 Shared Disks

(c) 공유 디스크 4개를 사용하였을 때
(c) using 4 Shared Disks

(그림 10) 네트워크 사용량 및 디스크 데이터 처리량
(Figure 10) Network Throughput and Processed Data Throughput

실험에서 SQL 서버가 설치된 인스턴스의 네트워크 대역폭은 10 Gbit이며 네트워크 디스크를 제공하는 있는 인스턴스의 네트워크 대역폭은 1 Gbit이다. 네트워크 디스크를 1개만 연결하였을 때 디스크 성능을 측정한 결과를 보면 초당 디스크에서 사용하는 데이터 양은 인스턴스에 직접 연결된 단일 볼륨보다 처리량이 낮은 것을 확인할 수 있다. 또한 네트워크로 공유된 디스크의 개수가 증가할수록 전체적인 처리 성능이 빨라지는 것을 확인할 수 있다. SQL 서버에서 각각 공유된 디스크로 데이터를 분산하여 처리하였을 때 공유된 디스크의 인스턴스에서 제공하는 각각의 대역폭은 1 Gbit이지만 SQL 서버에서 사용하고 있는 대역폭은 공유된 디스크의 전체 대역폭의 총합만큼 사용할 수 있음을 확인하였다. 즉 연결된 서버들 간의 네트워크 속도만 보장된다면 빠른 처리성능을 기대할 수 있다.

5. 결 론

본 논문은 클라우드 환경에서 인스턴스가 가지고 있는 물리적인 스토리지 제약을 SQL 서버의 특성을 활용하여 제약 사항을 초과하는 스토리지 구현 방법에 대해서 제안하였다. Amazon EC2 클라우드의 c4.8xlarge 인스턴스에서 SQL 서버가 자신의 인스턴스에 직접 연결한 Amazon EBS 볼륨을 RAID 0으로 구성하였을 때 구성된 디스크의 개수에 따라 성능이 비례해서 증가하는 것을 확인할 수 있었다. SQL 서버의 동일한 파일 그룹에서 각각의 단일 Amazon EBS 볼륨에서 데이터 파일을 구성하였을 때 데이터 파일의 개수에 따라 성능이 비례해서 증가하는 것을 확인할 수 있었다. 네트워크로 공유된 디스크를 사용한 경우에도 동일한 그룹으로 데이터 파일을 설정하였을 때에는 성능이 비례해서 증가하는 것을 확인할 수 있었지만 공유된 디스크를 사용하는 경우에는 인스턴스에 직접 연결된 디스크보다 낮은 성능이 측정되었다. 이유는 제공되는 인스턴스의 종류에 따라 네트워크 대역폭의 뺀 제약사항으로 인해 성능제약이 발생한 것으로 확인되었다.

일반적으로 클라우드 시스템은 인스턴스간의 연결이 내부적으로 초고속 네트워크로 구성되어 있다. 초고속 네트워크는 물리적인 단일 디스크의 데이터 전송 대역폭보다 더 많은 데이터를 전송할 수 있다. 이러한 특징을 이용하여 분산되어 있는 네트워크 디스크를 사용할 경우 각 인스턴스가 제공하는 물리적인 성능을 최대한 사용할

수 있어 더 빠른 데이터 처리 성능을 구현할 수 있다.

이 연구를 통해서 인스턴스에서 제공하는 단일 디스크 처리량 및 네트워크 대역폭에 대한 제약사항이 없다면 SQL 서버에서는 네트워크 디스크 공유를 사용하여 하나의 인스턴스에서 제공하는 물리적인 디스크 용량 제약을 초과하는 디스크 용량뿐만 아니라 성능까지 비례해서 고용량 고성능의 스토리지를 확보할 수 있다는 것을 보일 수 있었다.

본 논문에서는 클라우드의 IaaS 환경과 SQL 서버의 특성을 이용하여 대용량 스토리지 확보에 대한 방법에 대해서 살펴보았다. 클라우드의 IaaS 서비스에 SQL 서버를 설치하여 분산된 데이터 파일을 사용한다면 시스템을 중단시키지 않고 디스크 용량의 확장 및 축소가 가능하며, 인스턴스의 물리적인 디스크 제약을 초과하여 사용할 수 있으므로 높은 확장성을 유지하면서 상황에 따라 유연하게 디스크 용량을 조절함으로써 저렴한 비용으로 SQL 서버의 서비스를 제공할 수 있는 이점이 있다.

참 고 문 헌 (Reference)

[1] "Amazon Elastic Compute Cloud(Amazon EC2)," https://aws.amazon.com/ec2/

[2] "Amazon Elastic Compute Cloud Storage," http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Storage.html

[3] O.G. Min, H.Y. Kim, and G.H. Nam, "Trends in Technology of Cloud Computing (in Korean)," Electronics and Telecommunications Trends, Vol. 24, No. 4, Aug. 2009.

[4] "Amazon Instance Vloume Limits," http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/volume_limits.html

[5] "Amazon EBS Volume Types," http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html

[6] "SQL Server Database Files and Filegroups," https://msdn.microsoft.com/en-us/library/ms189563.aspx

[7] "Maximum Capacity Specifications for SQL Server," https://technet.microsoft.com/en-us/library/ms143432(v=sql.110).aspx

[8] HDFS, "http://hadoop.apache.org/core/docs."

[9] "Amazon EC2 Instance," http://aws.amazon.com/ec2/instance-types/?nc2=h_ls

[10] "Amazon EC2 Instance Configuration," http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-ec2-config.html

[11] "SQL Server Memory Server Configuration Options," https://msdn.microsoft.com/en-us/library/ms178067(v=sql.120).aspx

[12] "Description of support for network database files in SQL Server," https://support.microsoft.com/en-us/kb/304261.

◐ 저 자 소 개 ◑

강 성 욱 (Sungwook Kang)
2005년 동명대학교 모바일학과 졸업(학사)
2015년 숭실대학교 정보과학대학원 소프트웨어공학과 졸업(석사)
2009년 ~ 현재 NEXON
관심분야 : 데이터베이스, 분산처리 시스템, 데이터마이닝
E-mail : jevida@naver.com

최 종 선 (Jungsun Choi)
2000년 숭실대학교 컴퓨터학부 (학사)
2002년 숭실대학교 컴퓨터학과 (석사)
2010년 숭실대학교 컴퓨터학과 (박사)
2011년~2012년 숭실대학교 지능형로봇연구소 연구원
2012년~2013년 서일대학교 인터넷정보과 전임교원
2013년~현 재 숭실대학교 컴퓨터학부 조교수
관심분야 : 시스템소프트웨어, 병렬/분산처리, 지능형로봇
e-mail : jongsun.choi@ssu.ac.kr

최 재 영 (Jaeyoung Choi)
1984년: 서울대학교 제어계측공학과 (학사)
1986년: 미국 남가주대학교 전기공학과(컴퓨터공학) (석사)
1991년: 미국 코넬대학교 전기공학부(컴퓨터공학) (박사)
1992년~1994년: 미국 국립 오크리지연구소 연구원
1994년~1995년: 미국 테네시 주립대학교 연구교수
1995년~현재: 숭실대학교 컴퓨터학부 교수
관심분야: 시스템소프트웨어, 클라우드 컴퓨팅, 고성능컴퓨팅(HPC)
E-mail : choi@ssu.ac.kr