클라우드 데이터베이스를 위한 분리수행이 가능한 트랜잭션 처리 기법

Decoupled Transaction Processing System for Cloud Database

최 도 진, 송 석 일
한국교통대학교 컴퓨터공학과

Choi do-jin, Song seok-il
Department of Computer Engineering, Korea National University of Transportation

요약

이 논문에서는 클라우드 데이터베이스를 위한 분리수행이 가능한 트랜잭션 처리시스템을 제안한다. 제안하는 시스템은 기존의 클라우드 데이터베이스의 수정을 최소화 하면서 트랜잭션 처리가 가능하도록 할수 있다. 질의를 클라우드 데이터베이스의 CRUD (Create, Read, Update, Delete) 연산의 조합으로 변환하여 수행시킴으로써 동시성제어와 장애복구에 관한 기능을 제공한다.

I. 서론

기존의 클라우드 데이터베이스는 일관성 보다는 확장성과 가용성에 초점이 맞추어져있어 데이터 일관성에 관한 문제는 응용 개발자가 고려해야 할 문제로 남겨진다. 하지만, 현재 온라인 게임이나 협업 편집, 소셜 네트워크와 같은 응용에서는 상대적으로 높은 데이터 일관성을 요구하고 있다[1]. 일부 클라우드 데이터베이스에서는 데이터베이스의 일부 파티션에서만 트랜잭션이 수행되도록 제한하는 접근법을 이용하여 높은 일관성을 유지하기 위한 트랜잭션 기능을 지원하기도 한다[1-7]. 이 방법들은 대부분 낙관적 동시성 제어기법을 잠금 기법과 같이 사용하여 트랜잭션 응답시간을 저하시킬 수 있다.

[3, 7]에서 제안된 기법들은, 데이터베이스 커널을 트랜잭션 컴포넌트(TC)와 데이터 컴포넌트(DC)로 구분한다. TC는 동시성제어와 장애복구 기능을 제공하고, DC는 캐시 및 접근 관리방법(access methods) 관리를 담당한다. TC는 데이터의 물리적 저장 구조를 알지 못하고, DC는 트랜잭션 처리 과정을 알지 못 한 채 수행된다. 이를 통해 클라우드 데이터베이스의 확장성과 처리율의 향상을 얻는다.

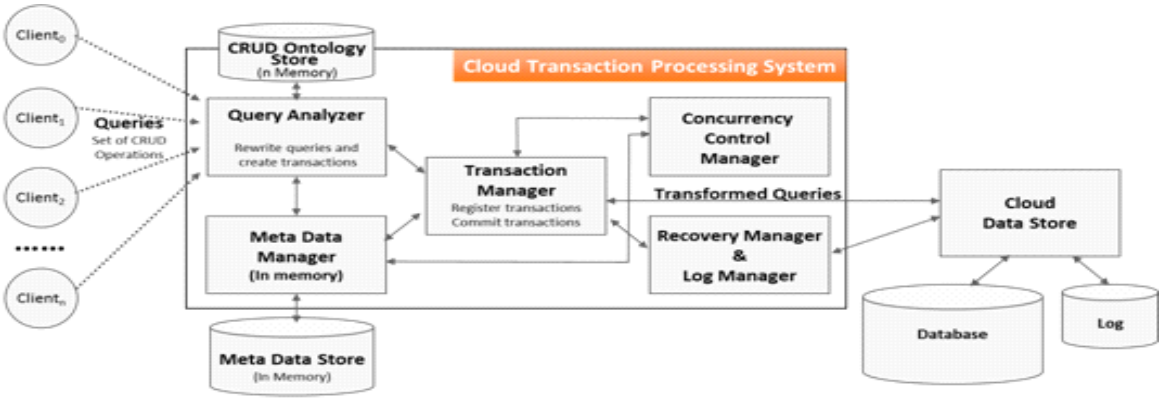
[3, 7]에는 TC가 테이블의 키에 대한 정보를 가지고 있지 않기 때문에 특정 키에 대한 잠금을 수행 할 수 없다. 이를 보완하기 위해 파티션 잠금 기법을 제안하고 있다. 파티션 잠금 은 물리적 리소스(페이지, 레코드) 대신에 테이블을 키의 범위로 나누어 다수의 파티션을 생성하고 각 파티션을 잠금 단위로 사용한다. 만약 키 범위가 동일하고 정적인 경우에는 키가 한쪽으로 쏠리는(Skewed) 문제를 처리하기가 어렵다. 키가 많이 몰리는 파티션에 대한 잠금은 트랜잭션의 처리율의 저하와 낮은 응답시간을 발생시킬 수 있다.

[1]에서는 잠금 기반 동시성 제어기법 대신 SI (Snapshot Isolation) 기법을 통하여 [3, 7]의 문제를 해결하고 있다. 또한, 동적 파티션 잠금 기법을 통해 키가 한쪽으로 쏠리는 문제를 해결 하였다. 이 방법에서는 각 파티션의 레코드 접근 수를 모니터링하고, 이를 통해 임계치보다 높은 파티션에 대하여 재 파티션으로 트랜잭션 처리율을 높인다.

[1, 3, 7]에서 제안된 기법은 기존의 클라우드 데이터베이스에는 적용하기 힘들다. 이미 클라우드 서비스 및 응용이 기존의 클라우드 데이터베이스를 기반으로 구축되어 있다. 따라서, 트랜잭션 처리를 위한 새로운 접근방법과 기술을 클라우드 데이터베이스에 적용하기 위해서는 기존의 클라우드 데이터베이스를 변경하지 않거나 최소화 하여야 한다.

본 논문에서는, 기존 클라우드 데이터베이스를 위한 새로운 트랜잭션 처리 시스템을 제안한다. 제안하는 클라우드 트랜잭션 처리 시스템은 클라우드 데이터베이스와 느슨하게 결합된다(loosely coupled). 따라서, 기존의 클라우드 데이터베이스를 크게 수정하지 않고 트랜잭션 처리가 가능하게 한다. 제안하는 기법(CTPS : Cloud Transaction Processing System)은 [1, 3. 7]의 TC와 유사하다. 그리고 기존의 클라우드 데이터베이스는 [1, 3, 7]의 DC와 유사한 역할을 수행한다. CTPS는 CRUD(create, read, update, delete) 질의를 분석하여 트랜잭션 형태의 질의로 변환한다. 또한, CTPS는 [1, 3, 7]와 유사한 동시성제어와 장애복구 방법을 사용한다.

본 논문은 다음과 같이 구성된다. 1장에서는 현재 클라우드 데이터베이스에서의 트랜잭션 처리에 관한 현황을 설명하고, 2장에서는 본 논문에서 제안하는 클라우드 트랜잭션 처리 시스템에 대하여 설명한다. 마지막으로 3장에서는 결론을 제시한다.



▶▶ 그림 1. 클라우드 트랜잭션 처리 시스템의 아키텍처

Ⅱ. 제안하는 클라우드 트랜잭션 처리 시스템

본 논문에서는, 기존의 클라우드 데이터베이스의 별다른 수정 없이 트랜잭션 처리가 가능한 클라우트 트랜잭션 처리시스템을 제안한다. 제안하는 시스템의 전체적인 아키텍처는 그림 1과 같다. CTPS는 질의 분석기, 메타 데이터 관리자, 트랜잭션 관리자, 동시성 제어 관리자, 복구 관리자, 로그 관리자로 구성되어 있다. 각 클라이언트로부터 클라우드 데이터베이스에 질의가 들어오면 질의 분석기는 먼저 질의를 분석한다. 질의는 CRUD 명령어로 구성된다고 가정한다. 질의 분석기는 CRUD 온톨로지를 이용해 질의를 트랜잭션 형태로 변환한다. 트랜잭션 형태의 질의는 트랜잭션 제어 문과 잠금 및 로그 명령어를 포함한다. 트랜잭션 제어문은 begin, abort, transaction, rollback으로 구성된다.

예를 들어, 클라이언트가 "read A; A=A+1; writeA;" 질의를 실행하면 질의 분석기는 이를 "begin transaction; r-lock (A); read A; A=A+1; w-lock (A); write A+timestamp; write A; return A to CTPS; log (A); If fail, rollback; commit;" 형태로 변환한다. CTPS는 낙관적 동시성제어 기법을 사용하므로 트랜잭션에는 타임스탬프와 잠금 명령어가 포함된다.

Deuteronomy[7]와 같이 제안하는 시스템은 클라우드 데이터베이스와 느슨하게 결합한다. 잠금 기법 또한 [7]에서 제안한 키 범위 기법을 채택한다. 키에 관한 정보는 메타 데이터 스토어에 저장되며 이는 빈번히 접근 되므로 메타데이터 스토어는 인 메모리 스토어가 빈번하게 접근 될 경우 지연이 발생하는 문제를 해경하기 위해 인 메모리 방식의 저장기법을 사용한다. 변환 된 트랜잭션은 트랜잭션 관리자에게 전달되고, 각 트랜잭션은 메타 데이터 괸리자를 통해 메타 데이터에 등록된다.

변환된 CRUD 명령어는 순차적으로 클라우드 데이터베이스에 전달되어 수행된다. CRUD 명령어를 보내기 전에 CTPS는 각 트랜잭션에 잠금의 필요 여부를 보고, 잠금이 필요하다면 트랜잭션 관리자는 동시성 제어 관리자에게 잠금 명령어를 전달한다. 질의가 수행된 후 에는 클라우드 데이터베이스에서 응답을 받게 되고, 로그 관리자는 반환된 결과를 분석하여 로그에 기록을 한다. 기록된 로그는 다시 클라우드 데이터베이스에 저장이 된다. 만약 클라우드데이터베이스에 실패 응답이 반환되면 트랜잭션 관리자는 트랜잭션을 abort 시킨 후 abort 로그를 생성할 것이다. 그렇지 않으면 트랜잭션은 commit 된다.

Ⅲ. 결론

본 논문에서는 기존의 클라우드 데이터베이스를 크게 수정하지 않고 트랜잭션 처리가 가능한 시스템을 제안하였다. CRUD 온톨로지를 이용하여 기존 클라우드 데이터베이스의 CRUD 명령어를 분석하여 트랜잭션 형태로 변환하여 동시성제어를 제공하였다. 또한, 로그를 통하여 장애복구에 대한 기능을 제공하였다.

▌참 고 문 헌▐

[1] T. Kim, and S. Song "Dynamic Partition Lock Method to Reduce Transaction Abort Rates in Cloud Data Management Systems", Cluster Computing Journal, accepted.

[2] M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska, "Building a database on S3." In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, 2008, pp. 251-264.

[3] D. Lommet, A. Fekete, G. Weikum, and M. Zwilling, "Unbundling transaction services in the cloud", In Proceedings of CIDR, 2010.

[4] S. Das, D. Agrawal, and A. E. Abbadi, "ElasTraS: An Elastic Transactional Data Store in the Cloud", In Proceedings of USENIX HotCloud Workshop, June 2009.

[5] Z. Wei, G. Pierre, and C.-H. Chi, "Scalable Transactions for Web Applications in the Cloud", In Proceedings of the Euro-Par Conference on Parallel Processing, 2009.

[6] J. Baker, C. Bond, J. Corbett, J. J. Furman, A. Khorlin, J. Larson, J. Léon, Y. Li, A. Lloyd, and V. Yushprakh, "Megastore: Providing Scalable, Highly Available Storage for Interactive Services", In Proceedings of CIDR, 2011, pp. 223-234.

[7] J. J. Levandoski, D. B. Lomet, M. F. Mokbel, and K. Zhao, "Deuteronomy: Transaction support for cloud data", In Proceedings of CIDR, 2011, pp. 123-133.