1. What are the major requirements in managing large data?

이용 가능한 정보의 양이 넘쳐나고 있다. 그리고 데이터의 가치가 광범위하게 인정되고 있다. 그 거대하고 복잡한 데이터들을 얻기 위해서 사람들은 간단하게 데이터를 관리하고 유용한 정보를 추출해 낼 수 있는 도구를 필요로 한다. 많은 양의 데이터를 관리하기 위해서는 큰 데이터들을 빠르게 처리할 수 있어야 하고 많은 양의 데이터가 있으면 많은 수의 유저들이 데이터에 접근하게 되는데 동시에 접근 할 때 일관된 정보를 제공할 수 있어야 한다. 또한 어떤 부분의 데이터들은 다른 사람들이 접근하지 못하도록 제한되어야만 한다. 이런 점들을 실행하기 위해서는 file에 데이터를 저장하고 특수한 용도의 코드를 써서 데이터를 관리할 수도 있지만 이 방법은 문제점이 많다. 그러나 DBMS라고 부르는 database management system은 거대한 데이터를 활용하고 관리 하는 데에 몇 가지 중요한 이점을 가지고 있다.

1. What are the limitations of file system to support those requirements in database management?

우리는 operate system file 에 데이터를 저장해 관리하는 것을 시도할 수 있다. 하지만 이 접근은 많은 문제점을 가지고 있다. 많은 양의 데이터는 보통 디스크나 테이프 같은 저장 장치에 저장한다. 그리고 그 저장 장치에서 메인 메모리로 프로세싱에 필요한 관련 있는 파트를 가져온다. 하지만 32비트 컴퓨터 시스템에서 데이터를 최대 4기가 바이트씩 조회할 수 있으므로 데이터 접근 시에 많은 시간이 소요된다. 따라서 필요한 데이터로 한번에 접근해서 참조할 수 있도록 하는 기술이 필요하다.

그리고 파일 시스템에서는 유저들이 요청하는 데이터를 처리하는 스페셜한 프로그램을 작성해야 하는데 이 프로그램이 매우 복잡하다 찾아야 하는 데이터의 양이 매우 크기 때문에.

또, data concurrency 문제도 있다. 다양한 사용자가 동시에 데이터에 접근을 하게 되는데 이 때 data concurrency가 지켜지지 않으면 어떠한 데이터의 수정 중에 발생한 invalid한 데이터를 사용하게 되는 문제가 생긴다.

보안과 관련한 부분에서도 문제점이 있다. Operating System 은 패스워드 메커니즘만을 제공한다. 이것은 데이터의 안전성과 사용자들의 접근권한을 관리하는 부분에서 충분히 유연하지 않다. This is not sufficiently flexible to enforce security policies in which different users have permission to access different subsets of the data.

DBMS의 필요성을 이해하기 위해서 시나리오 하나를 고려해보자 : 어떤 회사는 고용인, 상품 등

등에 대한500GB^3의 큰 데이터를 가지고 있다. 이 데이터는 고용인들에 대해 동시에 접근된다. 데이터에 대한 질문은 빠르게 대답해야만 하고 다른 유저들에 대해 일관되게 응용되도록 바뀌어야하고 확실한 파트의 데이터는 제한되어야만 한다.

우리는 시도할 수 있다 os file에 데이터를 저장해 관리하는 것을. 이 접근은 많은 문제점을 가지

고 있다.

우리는 아마도 500기가의 데이터를 메인 메모리에 모두 저장하지 않을 것이다. 우리는 디스크나 테이프 같은 저장 장치에 데이터를 저장해야만 한다. 그리고 메인 메모리로 프로세싱에 필요한 관련있는 파트를 가져와야한다.

우리가 500기가의 메인 메모리를 가지고 있다 하더라도 32비트 컴퓨터 시스템에서 우리는 4기가 이상의 데이터를 바로 조회할 수 없다. 우리는 모든 데이터를 확인하는 방법을 프로그래밍해야만 한다.

우리는 유저가 데이터에 대해 물어보는 모든 질문에 답을 하는 특별한 프로그램을 써야한다. 이 프로그램들은 복잡할 것이다 찾아야하는 데이터의 양이 크기 때문에.

우리는 보장해야한다 데이터가 일관된 상태로 저장되도록 만약 시스템 캐쉬가 변화를 만드는 동안에도.

OS는 보안에서 패스워드 메커니즘만을 제공한다. 이것은 충분히 유연하지 않다 다른 유저들이 데이터의 다른 부분집합들을 접근하는 것의 허가를 가지고 있는 보안 정책을 강화하는 것에

DBMS는 작업을 선행하기 쉽게 만들어진 소프트웨어이다. OS파일보다 DBMS에 데이터를 저장하는 것이 더, 우리는 DBMS의 데이터를 다루는 팔팔하고 효율적인 방식의 특징들을 사용할 수 있다. 데이터의 양과 수백 기가바이트로 자라나는 유저들의 수, 그리고 수천의 유저들은 현재 기업의 테이터베이스이다. DBMS의 도움은 필수적이 되었다.

1. How your database architecture and concept can overcome those limitations?

DBMS를 사용하면 데이터 관리에 많은 이점이 있다.

1. 데이터 독립성 : application 프로그램들은 이상적으로 저장소와 데이터의 모습이 자세하게 노출되지 않아야 한다. DBMS는 데이터의 세세한 부분을 숨기고 추상적인 모습을 제공한다.
2. 효율적인 데이터 접근 : DBMS는 데이터를 효율적으로 되찾아오고 저장하기 위해 세련된 기술의 다양성을 활용한다. 이 특징은 특히 중요하다 만약 데이터가 외부 저장소에 저장되어 있다면.
3. 데이터 진실성과 보안 : 만약 데이터가 항상 DBMS를 통해 접근된다면, DBMS는 강화할 수 있다 제약의 완전성을. 예를 들면, 고용인들에게 월급 정보를 주기 전에 DBMS는 회사 예산이 초과되지 않았는지 확인할 수 있다. 또한 이것은 어떤 데이터가 서로 다른 유저들에게 보이는지 지배하는 접근 통제를 강화시킨다.
4. 데이터 관리 : 몇몇의 유저들이 데이터를 공유할 때, 데이터 관리의 중앙집권화는 중요한 향상을 제공할 수 있다. 관리되는 데이터의 본질을 이해하고 다른 그룹의 유저들이 데이터를 어떻게 사용하는지 이해한 경험 많은 전문가들은 중복을 최소화하고 검색 저장을 효율적으로 하기 위해 데이터 저장을 미세 조정할 수 있도록 데이터 표현을 구성할 책임이 있다.
5. 동시 접근과 사고 회복 : DBMS는 유저들이 오직 한 명이 그 시간에 접근한다고 생각할 수 있도록 데이터 동시 접근을 조정한다. 더 멀리에, DBMS는 시스템 실패의 영향으로부터 유저들을 보호한다.
6. Application 개발 시간의 감소 : 분명하게, DBMS는 DBMS의 많은 application접근 데이터를 공통으로 하는 중요한 기능을 지지한다. 이것은, 데이터를 위한 높은 레벨의 인터페이스와 함께 빠른 application 개발을 가능하게 한다. 또한 DBMS application들은 비슷한 독립형의 application보다 더 팔팔할 가능성이 있다. 왜냐하면 많은 중요한 작업들은 DBMS에 의해 다루어 진다. (그리고 application에서 디버깅이나 테스트를 해서는 안된다.)

이 이점들을 보면 DBMS를 사용하지 않을 이유가 있는가? 때때로 있다. DBMS는 복잡한 소프트웨어이다, 확실하게 workload의 종류에 최적화되어있다(예를 들면, 복잡한 쿼리의 답을 한다던가 많은 동시 요청을 다룬다던가 하는), 그리고 이것의 퍼포먼스는 충분하지 않을지도 모른다 전문적인 application을 위해서는. .

1. Why data independence and query language are so important? Give one or two examples how the relational data model could remove the physical dependence in its data structure and query processing.
2. What is transaction? And, what is concurrency control? And, recovery? Why are they so important in database?

|  |  |  |
| --- | --- | --- |
|  | **File System** | **DBMS** |
| Data access | It is slow because it has to scan several times. | It is fast because you need to access and reference the data you need. |
| Data concurrency | Implementation is very complex. | Maintain data independence |
| Security | Operating systems provide only a password mechanism for security. So it is not suitable. | It protects the security by checking in advance before the user accesses the data. |
| Recovery | weak. | Protect users from the effects of system failure. |