DB 0310

1. 클라썸 질문 활용해라

=> 이유 : 다양한 사람들의 접근 법과 견해의 차이가 있기 때문이다.

2. 영어 원서 읽는 연습해라

=> 5년차 개발자가 되면 진두지휘해야하는 상황이 오는데, 영어 원서 읽을 줄 몰라서 신 기술을 도입하지 못할 것이기 때문이다.

3. 쿼리문은 주로 사용자가 사용한다.

=> DB 개발자, 관리자는 쿼리문은 기본적으로 알아야하며, DB 환경을 만드는 사람이기에 쿼리문을 주로 사용하진 않는다.

데이터베이스 시스템 특징

1. Real-time accessibility(실시간 접근)

2. Continuous evolution(연속적인 변화) => 끊임없이

3. Concurrent sharing(동시 공유) => 동시 접근가능해야 한다.

4. Content reference(내용 참조) => 사용자는 물리적 위치를 신경쓰지 않는다.

DBMS 구성

DB 구성 요소 -> DB 접근 -> 질의 처리기 -> 저장 장치 관리자 -> 디스크 저장소

3단계 스키마 구조

1. External Schema(외부 스키마) => 뷰

2. Conceptual Schema(개념 스키마) => 개념 단계

3. Internal Schema(내부 스키마) => 물리적 단계

RDBMS(관계 데이터 모델)

행(row = 튜플(tupple)), 열(column = 속성(attribute))의 형태로 저장하는 테이블(=릴레이션(relation)) 구조

데이터 무결성, 참조 무결성을 보장하는데 매우 효과적

무결성이란, 결점이 없다는 뜻이다.

DBRanking

개발자들이 사용하는 DB 순위가 적힌 홈페이지이다.(가장 최신버전)

꼭 RDBMS만 사용하는 것이 아닌, 다양한 모델(Relational, Document, Key-value 등)들이 있다.

Redis => 필요한 데이터 -> 메모리(휘발성)저장, 로그(데이터 기록) -> 디스크(비휘발성)저장

Schema => 명제, Pk, fk 제약조건

릴레이션 특징

행, 열 순서와 상관 없다.

튜플에 다중 값이 있는 경우 => 컬럼 분해

스키마 다이어그램(Schema Diagram) => DB 설계 1단계

무결성 제약조건

중복 제거

1. 수퍼 키(super key)

2. 후보 키(candidate key)

3. 기본 키(primary key; PK)

4. 외래 키(foreign key; FK)

기본 키를 참조하는 속성

외래 키를 가진 릴레이션 = 참조하는 릴레이션(referencing relation)

키의 포함 관계

Attribute > Super keys > Candidate keys > Primary keys

무결성 제약조건 종류

1. 도메인 무결성 제약조건(domain integrity constraint)

튜플들은 각 속성 도메인에 포함되는 값만 가져야 함

2. 기본 키 제약조건(primary key integrity constraint)

NULL값을 가지면 안됨, 중복되면 안됨

3. 외래 키 제약조건(foreign key integrity constraint)

참조하는 릴레이션이 존재하는 경우우, 참조하는 릴레이션 인스턴스가 삭제되기 전에, 참조되는 릴레이션의 튜플은 삭제될 수 없음음

SQL

1970년대 IBM 연구소에서 개발된 SEQUEL이라는 언어로 시작되었으며, 이후 1986년에 미국 국가표준 협회(ANSI)에 의해 DB 표준언어로 승인됨됨

SQL 주요 기능

1. 데이터 정의어(Data Definition Language; DDL)

-CREATE(생성), ALTER(변경), DROP(삭제)

2. 데이터 조작어(Data Manipulation Language; DML)

-SELECT(조회), INSERT(삽입), UPDATE(수정), DELETE(삭제)

3. 데이터 제어어(Data Control Language; DCL)

-GRANT(권한부여), REVOKE(취소), COMMIT(반영), ROLLBACK(복귀)

SELECT 절

질의 최적화, 대소문자를 구분하지 않음

WHERE 절

조건을 지정함, 얼마나 잘 활용하는지 중요

Cartesian Product

두 테이블 간의 모든 가능한 조합을 만들어내는 연산

Nested Subqueries(중첩 서브쿼리)

하나의 query가 다른 query안에 포함되어 있는 구조

-스칼라 서브쿼리(scalar subquery) : 단일 값 반환 => 집계 함수(count, average, sum)

-테이블 서브쿼리(table subquery) : 여러 행과 열을 반환

데이터 모델링

DB 프로그램 안에서 테이블 설계를 어떻게 할 것인지?

주의) 테이블 수정시 테이블을 삭제하고 다시 생성해야하므로 신중히 생성해야한다.

테이블 컬럼 추가시 -> 테이블 데이터 전체 삭제 -> 컬럼 추가 -> 데이터 재삽입입

파트별 권한 여부도 신경써야함.

프로젝트 진행 시 순서

기획 -> DB 설계 -> 프론트/백엔드

데이터베이스 생명주기

1. 요구사항 수집 및 분석

사용자에게 요구사항 수집하고 난 뒤에 사용자에게 '이렇게 설계하면 되는거죠?'라고 확인하는것은 하수중에 제일 하수이다.

(자기 분야는 책임을 갖고 일하기!)

2. 설계

3. 구현

4. 운영

5. 감시 및 개선

유지보수까지 생각하고 만드는 사람이 진정한 고수이다.

아마추어와 프로의 차이

아마추어 : 자신의 능력을 최대한 부각해서 만들줄 아는 사람(기간 내에 해결 X)

프로 : 자신의 능령을 제한된 범위에서 만들줄 아는 사람(기간 내에 해결 O)

DB 설계 시 1개의 DB만 사용하지 않는다

ex) 카카오톡 나와의 대화 DB, 사용자 대화 DB 분리 운영

Radis -> 메모리 저장 DB, 로그 저장 DB 분리 운영

Relation -> 이미지 저장 불가능, 이미지 경로나 이름만 저장 가능

데이터 모델링 과정

1. 요구사항 수집 및 분석

백엔드 개발자는 컴퓨터 저장 장치 비용 정도는 알아둬야 한다

이유 : 서버를 On-Premise(물리적 서버 구축)방식을 사용하여 만들어야 할 수도 있기 때문이다.

규모에 따라 DB를 서버에서 관리할 지? 개인 컴퓨터에서 관리할 지? 판단해야한다.

2. Conceptual Modeling(ERD 작성성)

필요한 구성요소, 구성요소 기록이 가능유무 등을 사용자와 협의 해야한다.

3. Logical Modeling

테이블의 기본키(PK), 외래키(FK) 등을 판단한다.

정규화 필수는 아니다 -> 사용자 환경에 따라 판단(이유 : join, cartesian product 처리 속도가 느려질 수 있기 때문에에)

4. Physical Modeling

DB언어 선택 => DBMS(MySQL, oracle 등), database 모델(relation, key-value 등등)

속도 처리 중요도, 저장 기간(SSD 안정, HDD 불안정)

-응답시간 최소화

-트랜잭션 동시성

-효율적인 배치 : DB 최적화(메모리, 비트 할당)

5. 데이터베이스 구현

관계 대수(relational algebra)

절차적 언어

대수학, 연산을 이용하여 질의하는 방법법

릴레이션의 수학적 의미(집합)

관계 대수 -> SQL -> 처리 속도 최적화

관계 대수 연산자

1. Select(σ, 시그마)

2. project(Π, 대문자 파이/프로젝트)

3. Union(U,유니온)

4. Set difference(-, 빼기/마이너스)

5. Cartesian product(X, 카르테션 프로덕트)

6. Rename(ρ, 로우/리네임)