3. SQL

Introduction to SQL

(Data Language 중 DDL 관련 내용 多)

Generated by 이유림

참고 강의 : https://www.edwith.org/database/lecture/27853/

HISTORY

SQL의 역사

- 1970s 초 : IBM Sequel Language (시스템 R 프로젝트, 관계형 DB 시스템)
- Sequel → Structured Query Language (SQL)
- ANSI와 ISO가 SQL의 표준을 만듦
- Commercial system : 최소 SQL-92 표준 기능 제공

- 표준이 존재하지만, 시스템마다 조금씩 다르기 때문에 예제가 모든 시스템에서 다 동작하진 않을 수 있음!

DATA LANGUAGE

- DDL (Data Definition Language, 데이터 정의어)

  : Relation의 스키마를 정의, 수정, 또는 Relation을 삭제할 때 사용

- DML (Data Manipulation Language, 데이터 조작어)

  : 데이터의 검색, 삽입, 삭제, 수정 등을 요청할 때 사용

  : DB의 정보를 질의하고, DB에 튜플을 삽입, 기존 튜플들을 삭제, 수정

DATA DEFINITION LANGUAGE (DDL)

: SQL의 DDL은 Relation에 관한 정보들을 명시하는 데 사용됨

- Relation의 스키마 : 각 relation이 어떠한 attribute를 가지고 있는지 정의
- Attribute에 관련된 값들의 도메인 : 테이블 구조와 타입 정의
- Integrity constraints (무결성 제약조건)

  : DB에 저장된 데이터가 만족시켜야 하는 제약조건
- 기타 : Relation의 Index에 대한 정보

  : 보안과 권한에 대한 정보

DOMAIN TYPES IN SQL

- **char(n)** : 길이가 n인 고정 길이 문자열
- **varchar(n)** : 최대 길이가 n인 가변 길이 문자열
- **int** : integer 정수형 (Machine-dependent : machine에 따라 표현 범위가 달라질 수 있음)
- **Smallint** : Small integer (Machine-dependent)
- **numeric(p, d)** : 고정 소수점 실수. p=부호 포함 전체 숫자의 길이, d=소수점 이하 숫자 길이
- **Real, double precision** : 부동 소수점 실수
- **float(n)** : 부동 소수점 실수, 길이 n을 사용자가 설정함
- **NULL & NOT NULL** : 속성이 null 값을 가질 수 있는지의 여부를 선언 (default=null 허용)

CREATE TABLE CONSTRUCT

- **create table** *r* (A1 D1, A2 D2, …, An Dn, (제약조건1), …, (제약조건k))
  - *r* = relation의 이름
  - An = Relation r의 스키마에 있는 Attribute의 이름
  - Dn = 이 attribute들의 도메인에 있는 값들의 데이터 타입
- **insert into** *r* **values** (*위에서 지정한 r 릴레이션의 속성값 순서대로*)
  - 문자열은 작은따옴표로 표시
  - 각 값은 콤마로 구분
  - 속성값들의 순서는 Relation 스키마의 속성 순서와 일치해야 함!

CREATE TABLE CONSTRUCT

```
create table instructor (
        ID              char(5),                    // ID=길이가 5로 고정된 문자열
        name            varchar(20) not null,       // name=최대 길이가 20인 문자열(null X)
        dept_name       varchar(20),                // dept_name=최대 길이가 20인 문자열
        salary          numeric(8, 2) );
        // ↑전체 숫자의 최대 길이가 8이고, 소수점 이하에 2자리의 값을 가지는 실수
        // 참고) 달러 표시하려고 salary를 numeric 자료형으로 씀

// ID='10211', name='Smith', dept_name='Biology', salary=66000이 저장됨
insert into Instructor values ('10211', 'Smith', 'Biology', 66000);
```

명령 실행 결과 →

ID	name	dept_name	salary
10211	Smith	Biology	66000

name 속성에 null 값을 넣으려고 하면, DB가 name 속성에 null 값이 들어오면 tuple이 삽입되는 것을 거부함.

INTEGRITY CONSTRAINTS IN CREATE TABLE (1)

Integrity Constraints (무결성 제약조건) – SQL에서 지원

1. 객체 무결성 제약조건

- **not null constraint** : not null로 선언된 속성은 null 값을 가질 수 없음.
  해당 속성의 값이 null인 tuple을 삽입하려 하거나 속성의 값을 null로 수정하려고 하면 거부당함.
- **primary key** : 1) primary key(기본키)의 유일성을 보장
  2) 기본키를 구성하는 속성들은 null이나 중복된 값을 가질 수 없음
  3) 기본키 값으로 이미 Relation 안에 존재하는 값을 가진 tuple은 삽입 X

선언 방법) **primary key**(A1, …, An) : A1~An 속성들로 구성된 키를 primary key로 선언.

INTEGRITY CONSTRAINTS IN CREATE TABLE (2)

Integrity Constraints (무결성 제약조건) – SQL에서 지원

2. 참조 무결성 제약조건

- **foreign key** : 다른 relation의 기본키를 참조하여 두 relatio의 관계를 표현함.

(외래키는 참조되는 relation의 기본키 값 중에서 가질 수 있음.)

선언 방법) **foreign key**(Am, …, An) **references** *r*(B1, …, Bn) : Am~An 속성들로 구성된 키를 primary key로 선언.

이 foreign key는 relation r의 primary key인 B1~Bn 속성들을 참조함.

EXAMPLE

예) 앞에 나온 예시 *Instructor* relation의 primary key와 foreign key 선언

```
create table instructor (
        ID              char(5),
        name            varchar(20) not null,
        dept_name       varchar(20),
        salary          numeric(8, 2),
        primary key (ID),
        foreign key (dept_name) references department (dept_name) );
```

- *Instructor* relatio에 삽입되는 tuple의 dept_name에 있는 값은 반드시 *department* relation의 dept_name 속성값 중 하나여야 함. **(foreign key 제약조건)**
- **Primary key** 제약조건에는 **not null** 제약조건이 포함되어 있으므로, **primary key** 선언 시 **not null**을 따로 선언하지 않아도 됨.

RELATION 정의 예시 1

```
create table student (                                  // create table relation의 이름
        ID              varchar(5),                     /*
        name            varchar(20) not null,           * 속성(attribute)들과
        dept_name       varchar(20),                    * 각 속성의 도메인 타입 선언
        tot_cred        numeric(3, 0),                  */
        primary key (ID),                               // ID를 student relation의 기본키로 선언
        foreign key (dept_name) references department (dept_name) );
        /*
        *   student relation이 department relation을 참조하기 때문에
        *   department relation의 dept_name을 foreign key로 선언
        */
```

RELATION 정의 예시 2

```
create table takes (
        ID              varchar(5),
        course_id       varchar(8),
        sec_id          varchar(8),
        semester        varchar(6),
        year            numeric(4, 0),
        grade           varchar(2),
        primary key (ID, course_id, sec_id, semester, year),
        foreign key (ID) references student (ID),
        foreign key (course_id, sec_id, semester, year) references selection (course_id,
                    sec_id, semester, year)
);
```

RELATION 정의 예시 3

```
create table course (
        course_id       varchar(8) primary key,    // primary key 정의는 이렇게도 가능!
        title           varchar(50),
        dept_name       varchar(20),
        credits         numeric(2, 0),
        foreign key (dept_name) references department (dept_name)
);
```

Primary key는 속성(attribute)을 선언함과 동시에 선언할 수 있음.

RELATION의 SCHEMA 삭제/수정 (1)

- **Drop table** *relation*

  : SQL 데이터베이스에서 *relation* 제거

  (*relation*에 들어있는 모든 tuple들과 relatio의 schem가 삭제됨)

- **Delete from** *relation*

  : *relation*의 모든 컨텐츠(tuple) 삭제

  : *relation*의 schema는 삭제 **X**

  (→ *relation*을 다시 생성(**create table**)하지 않고도 *relation*에 tuple을 삽입할 수 있음.)

RELATION의 SCHEMA 삭제/수정 (2)

- **alter table** – 추가(add) 또는 삭제(drop)

  1) **add** : 기존 relation에 새로운 속성 추가

  - **alter table** *relation* **add** *attribute domain*

  : 기존의 *relation*에 도메인타입 *domain*을 가지는 새로운 속성 *attribute*를 삽입.

  - *relation*에 들어있던 tuple들은 새로 삽입된 속성에 대해 null 값을 가짐.

  2) **drop** : 기존 relation의 속성 삭제

  - **alter table** *relation* **drop** *attribute*

  : 기존의 *relation*에서 속성 *attribute*를 삭제

  : 참조 무결성 때문에 속성의 삭제는 데이터베이스에서 지원하지 않는 경우가 많음.

RELATION의 SCHEMA 삭제/수정 (3)

예) **alter table** *students* **add** birth DATETIME; // *student*에 birth라는 새로운 속성 추가

ID	name	dept_name	tot_credit
10109	Park	Biology	3.75
10121	Kim	Comp. Sci	3.8

↓ **alter table** *students* **add** birth DATETIME;

ID	name	dept_name	tot_credit	birth
10109	Park	Biology	3.75	null
10121	Kim	Comp. Sci	3.8	null

다른 relation이 참조하는 속성이 삭제되면, 그 속성을 참조하는 다른 relation은 참조할 속성을 잃어버리기 때문에 drop table과 alter table drop은 신중하게 사용해야 함! **주의!**