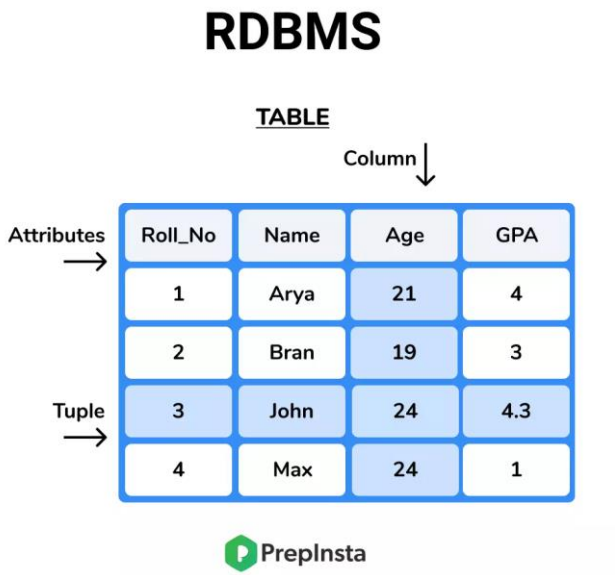
DMBS

2일차

강사 : 김근형

◆ 관계형 데이터베이스(RDBMS)

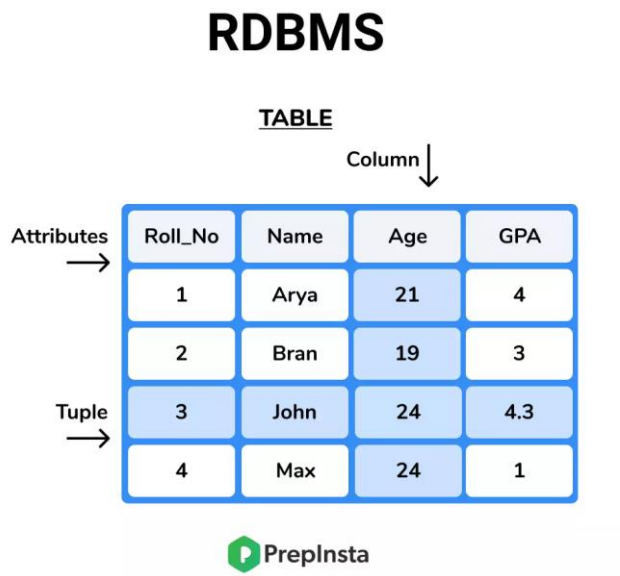
- 데이터가 하나 이상의 열과 행의 테이블에 저장
- 서로 다른 데이터 구조가 어떻게 관련되어 있는지 쉽게 파악하고 이해할 수 있도록 사전 정의된 관계로 데이터를 구성
- 관계 : 테이블 간의 상호작용을 기반으로 설정되는 여러 테이블 간의 논리적 연결



RDBMS

◆ 릴레이션

- 표(테이블)을 릴레이션이라 부름
- DB에서 정보를 구분하여 저장하는 기본 단위
- 동일한 DB 내에서는 같은 이름을 가진 릴레이션이 존재할 수 없음
- 행과 열로 표현



◆ 애트리뷰트(속성)

- 테이블(릴레이션)의 속성 / 하나의 열은 하나의 속성 정보를 표시
- 테이블 내부에 저장될 데이터(튜플)들의 이름을 의미
- 내가 저장하고 싶은 데이터(튜플)의 항목

사원번호	이름	전화번호	부서
2013100100	이아람	01012345667	1
2013100101	노은경	01099992222	2
2013100102	이은재	01023451111	4
2013100103	요오리	01033399234	1

속성

http://jhnyang.tistory.com

◆ 차수(degree)

- 한 릴레이션에 들어 있는 애트리뷰트의 수
- 현재 사진에서는 차수는 4

사원번호	이름	전화번호	부서
2013100100	이아람	01012345667	1
2013100101	노은경	01099992222	2
2013100102	이은재	01023451111	4
2013100103	윤우리	01033399234	1

속성

http://jhnyang.tistory.com

◆ 튜플(tuple, 레코드, 데이터)

- 릴레이션의 각 행
- 한 마디로 릴레이션에 실제적으로 들어 있는 값(데이터)
- 한 릴레이션의 포함된 튜플의 개수는 계속 변함

사원번호	이름	전화번호	부서
2013100100	이아람	01012345667	1
2013100101	노은경	01099992222	2
2013100102	이은재	01023451111	4
2013100103	요오리	01033399234	1

속성

http://jhnyang.tistory.com

◆ 카디날리티(Cardinality)

- 릴레이션의 튜플의 개수
- 현재 사진에서의 카디날리티는 4
- 카니달리티는 0이 될 수 있음

사원번호	이름	전화번호	부서
2013100100	이아람	01012345667	1
2013100101	노은경	01099992222	2
2013100102	이은재	01023451111	4
2013100103	요오리	01033399234	1

속성

http://jhnyang.tistory.com

◆ 도메인(Domain)

- 릴레이션의 포함된 각각의 속성들이 가질 수 있는 값들의 집합
- 릴레이션의 저장되는 데이터 값들이 본래 의도했던 값들만 저장되고 관리하기 위해 사용
- EX) 성별 -> '남', '여' 만 저장('남', '여'가 도메인, 그 외의 것은 저장 X)

사원번호	이름	전화번호	부서
2013100100	이아람	01012345667	1
2013100101	노은경	01099992222	2
2013100102	이은재	01023451111	4
2013100103	요오리	01033399234	1

속성

http://jhnyang.tistory.com

◆ 스키마

- DB의 구조와 제약 조건
- DB 구성 시 데이터가 갖는 값의 제약 조건
- 튜플의 형태 또는 모양 / 외부, 개념, 내부로 나뉨(추후에 다시 설명)

사원번호	이름	전화번호	부서
2013100100	이아람	01012345667	1
2013100101	노은경	01099992222	2
2013100102	이은재	01023451111	4
2013100103	요오리	01033399234	1

속성

http://jhnyang.tistory.com

분산 데이터베이스 환경

◆ 분산 데이터베이스

- 중앙 집중형 데이터베이스
- 물리적으로 여러 대의 서버가 존재하고 네트워크로 하나로 연결된 구조
- 단일 데이터베이스처럼 보여주고 분산된 작업 처리
- 투명성 제공

분산 데이터베이스 투명성

- 위치 투명성
  - 고객은 사용하려는 데이터의 저장 장소를 몰라도 된다.
- 중복 투명성
  - 데이터베이스가 여러 시스템에 중복되어 존재하더라도 일관성은 유지된다.
- 장애 투명성
  - 분산된 데이터베이스에 장애가 발생해도 데이터 무결성은 보장된다
- 병행 투명성
  - 여러 사용자가 동시에 트랜잭션을 수행해도 결과에 이상이 없다.

SQL 자료형

◆ SQL에서의 자료형(데이터 유형, 데이터 타입)

- 숫자형 : TINYINT, INT, FLOAT
- 문자형 : CHAR, VARCHAR, TEXT, ENUM
- 날짜형 : DATE, DATETIME, TIMESTAMP

- 크게 나누면 이렇게 3개

SQL 자료형

◆ 숫자형 자료형

- TINYINT
  - 가장 작은 숫자 자료형
  - 부호가 있는 경우(음수 포함) → -128 ~ 127
  - 부호가 없는 경우(음수 미포함) → 0 ~ 255
  - 1byte의 크기를 가진다

- SMALLINT
  - 부호가 있는 경우 → -32768 ~ 32767
  - 부호가 없는 경우 → 0 ~ 65535
  - 2byte의 크기를 가진다

SQL 자료형

◆ 숫자형 자료형

- MEDIUMINT
  - 부호가 있는 경우 → -8388608 ~ 8388607
  - 부호가 없는 경우 → 0 ~ 16777215
  - 3byte 크기를 가진다

- INT
  - 일반적인 숫자 자료형
  - 부호가 있는 경우 -> -2147483648 ~ 2144483647
  - 부호가 없는 경우 -> 0 ~ 4294967295
  - 4byte 크기를 가진다

SQL 자료형

◆ 숫자형 자료형

- INTEGER
  - INT와 표기법만 다르고 같다

- BIGINT
  - 부호가 있는 경우 -> -9223372036854775808 ~ 9223372036854775807
  - 부호가 없는 경우 -> 0 ~ 18446744073709551615
  - 8byte 크기를 가진다

- FLOAT(실수형)
  - 최소 +_1.175494351E-38 ~ 최대 +_3.402823466E_38
  - 4byte 크기를 가진다

SQL 자료형

◆ 숫자형 자료형

- DOUBLE(실수형)
  - 최소 ±1.7976931348623157E-308 ~ 최대 ±2.2250738585072014E+308
  - 8byte 크기를 가진다

- DECIMAL
  - 소수를 저장하는 용도로 사용
  - 내부적으로는 문자 형태로 저장되는 타입
  - EX) 3.141592에서 141592를 저장

SQL 자료형

◆ 문자형 자료형

- CHAR
  - CHAR(0)도 지원
  - CHAR(N)에서 실제 값이 N만큼의 글자 수보다 작을 때 남는 자릿수는 공백
  - 지정한 크기만큼 공백을 붙여 모두 사용(저장)
  - 1부터 최대 255의 자릿수를 지원, CHAR(N)에서 N으로 지정한 용량만큼 byte 사용
  - "입력한 글자 수" byte 사용

- VARCHAR
  - 저장할 수 있는 길이(N)은 1 ~ 255 글자 수 지정
  - VHACHAR(N)에서 사용한 길이까지 저장 / 남는 자리 사용 X
  - CHAR 형보다 기억 장치를 효율적으로 사용
  - "입력한 글자 수" + 1 byte 사용

SQL 자료형

◆ 문자형 자료형

- TINYBLOB
  - 최대 255개의 문자를 저장
  - "지정한 용량" + 1 byte 용량 사용
- TINYTEXT
  - TINYBLOB과 같음
- BLOB
  - 최대 65535개의 문자를 저장
  - "지정한 용량" + 2 byte 용량 사용
- TEXT
  - BLOB과 같음

SQL 자료형

◆ 문자형 자료형

- MEDIUMBLOB
  - 최대 16777215개의 문자를 저장
  - "지정한 용량" + 3 byte 용량 사용
- MEDIUMTEXT
  - MEDIUMBLOB과 같음
- LONGBLOB
  - 최대 429496729개의 문자를 저장
  - "지정한 용량" + 4 byte 용량 사용
- LONGTEXT
  - LONGBLOB과 같음

SQL 자료형

◆ 문자형 자료형

- ENUM
    - 입력한 문자 형태의 값을 숫자로 저장
    - 잘 사용하지 않음

    - 사용하지 않는 이유
        - 정규화 위반
        - 데이터 수정이 어려움
        - 속성 추가 및 연관 정보 저장 불가능
        - 조회가 어려움
        - 최적화 효과가 없음
        - 다른 테이블에서 재사용 불가능

SQL 자료형

◆ 날짜형 자료형

- DATE
  - '1001-01-01' ~ '9999-12-31' 저장 가능
  - 저장 용량은 3byte
  - 'YYYY-MM-DD'와 같은 형식
  - Y → 연도 / M → 월 / D → 일

- DATETIME
  - 날짜와 시간을 같이 저장
  - '1001-01-01 00:00:00' ~ '9999-12-31 23:59:59' 저장 가능
  - 저장 용량은 8byte
  - 'YYYY-MM-DD HH:MM:SS'와 같은 형식
  - HH → 시간 / MM → 분 / SS → 초

SQL 자료형

◆ 날짜형 자료형

- TIMESTAMP
  - '1970-01-01 00:00:00' 이후부터 초를 숫자로 저장하는 자료형
  - 저장 용량은 4byte
  - 약 40억 초를 저장할 수 있음(2037년 까지)

- TIME
  - '`838:59:59' ~ '838:59:59' 저장 가능
  - 저장 용량은 3byte
  - 'HH:MM:SS' 같은 형식

SQL 자료형

◆ 날짜형 자료형

- YEAR
  - 연도만 저장하는 자료형
  - YEAR(N)와 같은 형식
  - N은 2와 4를 지정할 수 있음
  - N이 2인 경우에 값의 범위는 70 ~ 69
  - N이 4인 경우에 값의 범위는 1970 ~ 2069
  - 저장 용량은 1byte

SQL 기초

◆ CREATE 문

- 데이터베이스와 테이블(릴레이션)을 제작하는 SQL

- 데이터베이스 생성
- EX) ‘DB_NAME’ 이라는 이름을 가진 데이터베이스를 생성하는 예
    - CREATE DATABASE DBNAME;

SQL 기초

◆ CREATE 문

- 데이터베이스와 테이블(릴레이션)을 제작하는 SQL

- 테이블 생성

- EX) 'TABLE_NAME' 이라는 이름을 가진 테이블을 생성하는 예
  - CREATE TABLE TABLE_NAME(
    column_name1, data_type(size)
    column_name2, data_type(size)
    column_name2, data_type(size)
    ...);

    - TABLE_NAME : 테이블 명
    - column_name : 속성(애트리뷰트)의 이름
    - data_type(size) : 속성(애트리뷰트)에 있는 데이터의 타입을 명시 (EX, varchar, int 등..)

◆ CREATE 문 예시1 → 데이터베이스 생성 포함

- CREATE DATABASE SAMPLE_DB;
- USE SAMPLE_DB;
- CREATE TABLE SAMPLE_TABLE(
  sample_col1 varchar(10) not null
  );

SQL 기초

◆ CREATE 문 예시2 → 테이블만 생성

- CREATE TABLE Persons
- (
- PersonID int,
- LastName varchar(255),
- FirstName varchar(255),
- Address varchar(255),
- City varchar(255)
-);

PersonID	LastName	FirstName	Address	City

◆ CREATE문 심화

- CREATE문을 활용한 테이블 생성에는 특정 옵션도 줄 수 있음
- 옵션의 종류
  - NOT NULL : NULL값(아무것도 없는 값)이 들어가지 않아야 함
  - UNIQUE : 유니크한 값(해당 테이블에서 하나인)이 들어가야 함
  - PRIMARY KEY : NOT NULL과 UNIQUE의 조합 / 기본 키라고 부름
  - FOREIGN KEY : 다른 테이블과 매치되는 데이터를 찾기 위한 참조 키 값
  - CHECK : 값이 정해진 조건에 충족하는지 체크
  - DEFAULT : 아무것도 쓰지 않으면 DEFAULT로 정해진 값을 갖는다

SQL 기초

◆ CREATE문 심화 → NOT NULL

- CREATE TABLE PersonsNotNull
- (
- P_Id int NOT NULL,
- LastName varchar(255) NOT NULL,
- FirstName varchar(255),
- Address varchar(255),
- City varchar(255)
-)

SQL 기초

◆ CREATE문 심화 → UNIQUE

- CREATE TABLE Persons
- (
- P_Id int NOT NULL UNIQUE,
- LastName varchar(255) NOT NULL,
- FirstName varchar(255),
- Address varchar(255),
- City varchar(255)
-)

SQL 기초

◆ CREATE문 심화 → PRIMARY KEY

- CREATE TABLE Persons
- (
- P_Id int PRIMARY KEY,
- LastName varchar(255) NOT NULL,
- FirstName varchar(255),
- Address varchar(255),
- City varchar(255)
-)

SQL 기초

- **CREATE문 심화 → FORIEGN KEY**
- **FOREIGN KEY는 다른 테이블의 PRIMARY KEY 값이다**
  - CREATE TABLE Orders
  - (
  - O_Id int NOT NULL PRIMARY KEY,
  - OrderNo int NOT NULL,
  - P_Id int FOREIGN KEY REFERENCES Persons(P_Id)
  -)

SQL 기초

◆ 실습하기 전 PRIMARY KEY와 FOREIGN KEY에 대한 설명

◆ PRIMARY KEY(기본 키)

- 기본 키는 다른 항목과 절대로 중복되어 나타날 수 없는 단일 값
- 기본 키는 절대로 NULL(아무 값이 없는 상태)가 될 수 없음
- EX) 주민등록번호 → 동일 번호 불가능, 모두 다 있음

◆ FOREIGN KEY(외래 키)

- 두 개의 테이블을 연결해주는 다리 역할
- 새롭게 추가되는 데이터(튜플)에서 외래 키에 해당 하는 값이 참조하는 테이블에 존재하는지 체크
- EX) 주문 데이터를 넣으려고 할 때 회원 테이블에 아직 가입하지 않은 아이디일 때 오류

ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다

PersonID	LastName	FirstName	Address	City

- 현재 이렇게 테이블이 있다고 가정하면
- City라는 컬럼(속성)을 삭제하고 싶다면
  - alter table 테이블명 drop column City;

SQL 기초

◆ ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다
- 현재 이렇게 테이블이 있다고 가정하면

PersonID	LastName	FirstName	Address	City

- Sample라는 컬럼(속성)을 추가하고 싶다면
  - alter table 테이블명 add Sample varchar(20);

SQL 기초

◆ ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다

- 이름 변경
  - alter table 테이블명 rename column 기존이름 to 바꾸려는이름;
  - EX) alter table 'sample_table' rename column 'sample1' to 'sample2';

◆ ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다

- 데이터 타입 변경
  - alter table 테이블명 modify column 컬럼명 새로운 데이터 타입;
  - EX) alter table 'sample_table' modify column 'sample1' 'int';

SQL 기초

◆ ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다

- 컬럼명과 데이터 타입 변경
  - alter table 테이블명 change column 기존이름 바꾸려는이름 새로운 데이터 타입;
  - EX) alter table 'sample_table' change column 'sample1' 'smaple2' 'int';

◆ ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다

- 테이블 이름 변경
  - alter table 테이블명 rename to 변경하려는 테이블명
  - EX) alter table 'sample_table' rename to 'smaple_table2';

◆ ALTER 문

- 테이블을 수정할 때 사용하는 구문
- 컬럼(속성)를 추가 및 삭제할 수 있고 변경도 가능합니다

- 기본 키 삭제
  - alter table '테이블 명' drop primary key;
  - EX) alter table 'sample_table' drop primary key;

- 새로운 기본 키 지정
  - alter table '테이블 명' add primary key('열 이름');
  - EX) alter table 'sample_table' add primary key('sample_col1');

DROP 문

- 데이터베이스를 삭제하거나
- 테이블을 삭제하는 용도입니다

- drop database '데이터베이스 이름';
- drop table '테이블 이름';