데이터베이스

1. Fundamental

Relation DBMS

- Oracle
- MS SQL Server
- MariaDB(MySQL)
- PostgreSQL

DB

- Data Achitect(Data Modeling)
- DBA
- Database Developer (우리!!)

Normalization(정규화)

데이터베이스 & 데이터베이스 관리 시스템

1. 특성

- 실시간 접근성

  > 사용자 요구 즉시 처리 가능

- 계속적인 변화

  > 정확한 값을 유지하려고 삽입/삭제/수정 작업 등을 이용해 데이터를 지속적으로 갱신 가능

- 동시 공유성

  > 사용자마다 서로 다른 목적으로 사용하므로 동시에 여러 사람이 동일한 데이터에 접근 및 이용 가능

- 내용 참조

  > 저장한 데이터 레코드 위치나 주소가 아닌 사용자가 요구하는 데이터의 내용, 즉 데이터 값에 따라 참조

2. DBMS(데이터베이스 관리 시스템, Database Management System)

- 데이터베이스를 관리하는 SW
- 여러 응용 SW 또는 시스템이 동시에 DB에 접근하여 사용할 수 있게 한다.
- 필수 3기능

    - 정의 기능: DB의 논리적, 물리적 구조 정의
    - 조작 기능: 데이터 검색, 삭제, 갱신, 삽입 기능
    - 제어 기능: DB의 내용 정확성과 안전성을 유지하도록 제어하는 기능

파일 시스템과의 비교

- 데이터 종속성 보완, 중복성 제공

3. DBMS 장단점

- **장점**
    - 데이터 중복 최소화
    - 데이터 일관성 및 무결성 유지
    - 데이터 보안 보장
- **단점**
    - 운영비가 비싸다
    - 백업 및 복구에 대한 관리 복잡
    - 부분적 DB 손실이 전체 시스템을 정지

4. DB의 종류

- **관계형 데이터베이스(RDB, Relational Database)**

> 키와 값들을 간단한 관계를 테이블화 시킨 매우 간단한 원칙의 개념의 DB
> 일련의 정형화된 테이블로 구성된 데이터 항목들의 집합이며 각 테이블은 데이터 성격에 따라 여러 개의 컬럼(키)이 포함됨
> 사용자는 SQL이라는 표준 질의어를 통해 데이터를 조작 또는 조회할 수 있다.

- **객체지향 데이터베이스(OODB, Object Oriented Database)**

> 정보를 객체의 형태로 표현하는 DB
> 객체 모델이 그대로 DB에도 적용되어 데이터 모델을 그대로 응용프로그램에 적용, 데이터 변환과 질의 작업이 필요치 않은 장점

- **객체 관계형 데이터베이스(ORDB, Object Relation Database)**

> 관계형 DB에서 사용하는 데이터를 확장
> 관계형 DB를 객체지향 모델링과 데이터를 관리하는 기능을 갖도록 확장한 것

- **NoSQL**

> 대용량 데이터의 웹 서비스와 SNS, 클라우드 컴퓨팅의 확대 보급과 대중화로 최근 주목 받는 DB 기술

관계형 데이터베이스

ORM(Object Relational Mapping)

객체와 관계형 데이터베이스의 데이터를 자동으로 매핑(연결)해주는 것

1. 논리적(개념적) 데이터 모델링 & 물리적 데이터베이스

학생	엔티티(Entity)
학번 이름 학년 학과	**속성(Attribute)**

테이블 - 엔티티(Entity)	행(Column) - 속성(Attribute)	데이터 타입	제약조건
Student	student_id	Number	PK
	name	VARCHAR(80)	
	grade	Number	
	dept	VARCHAR(40)	

> PK는 비즈니스와 관계 없는 것으로 잡아야 함!!
>
> - 주민번호와 같은 것은 PK XXXX

Student

no (PK)	name	major	grade
1	둘리	C.S	1
2	김마이콜	C.S	2
3	이마이콜	P	3
4	또치	h	4

변경
Major

no	전공
1	C.S
2	P
3	h

Student

no (PK)	name	major_no (FK)	grade
1	둘리	1	1
2	김마이콜	1	2
3	이마이콜	2	3
4	또치	3	4

SQL 개요

- DB 스키마 생성, 자료 검색, 수정, DB 객체 접근 관리 등을 위해 고안된 언어
- 다수의 DB 관련 프로그램의 표준언어
- SQL 명령어 종류
  - **DML(Data Manipulation Language)** : 데이터 조작어로 검색 및 수정하기 위한 수단 제공

    > ☆**SELECT**☆, INSERT, UPDATE, DELETE, MERGE
    > SELECT - 기본 / 집계 / ☆**조인**☆ / ☆**서브쿼리** ☆
  - **DDL(Data Definition Language)** : 데이터 구조 생성, 변경, 삭제 등의 기능 제공

    > CREATE, ALTER, DROP, RENAME
  - **DCL(Data Control Language)** : 데이터에 대한 권한 관리 및 트랜잭션 제어

    > GRANT, REVOKE

MariaDB 기본

About MySQL

MySQL 구성

- MySQL Server
  - Community Server
  - Enterprise Server
  - Embedded Server
- MySQL GUI Tools
  - Query Browser
  - Administrator
  - Migration Toolkit
  - Visual Studio Plug-in
  - MySQL Workbench
- MySQL Drivers
  - JDBC
  - ODBC
  - .NET

- PHP

Basic Query

MariaDB 실습

- **쿼리문이 끝**나면 **세미콜론(;)**을 붙인다.

```
MariaDB [(none)]> select version(), current_date;
```

> 실행결과
>
> ```
> +-----------------+--------------+
> | version()       | current_date |
> +-----------------+--------------+
> | 10.1.48-MariaDB | 2021-09-30   |
> +-----------------+--------------+
> 1 row in set (0.00 sec)
> ```

- **키워드**는 **대소문자 구별이 없다**.

```
MariaDB [(none)]> SELECT Version(), Current_date;
```

> 실행결과
>
> ```
> +-----------------+--------------+
> | Version()       | Current_date |
> +-----------------+--------------+
> | 10.1.48-MariaDB | 2021-09-30   |
> +-----------------+--------------+
> 1 row in set (0.00 sec)
> ```

- **계산기로도 사용**할 수 있다.

```
MariaDB [(none)]> select sin(pi()/4);
```

> 실행결과
>
> ```
> +--------------------+
> | sin(pi()/4)        |
> +--------------------+
> | 0.7071067811865475 |
> +--------------------+
> 1 row in set (0.00 sec)
> ```

- **여러 문장을 한 줄에 연속으로 붙여서 쿼리 실행이 가능**하다. **각 문장**에 **세미콜론**만 붙여주면 된다!

```
MariaDB [(none)]> select version(); select now();
```

실행결과

```
+-----------------+
| version()       |
+-----------------+
| 10.1.48-MariaDB |
+-----------------+
1 row in set (0.00 sec)

+---------------------+
| now()               |
+---------------------+
| 2021-09-30 19:09:55 |
+---------------------+
1 row in set (0.00 sec)
```

- **Multi-Line Commands**
  문장의 끝을 라인으로 구분하는 것이 아닌 세미콜론으로 구분하기 때문에 여러줄에 거쳐 문장 쓰기도 가능

```
MariaDB [(none)]> select
    -> user()
    -> ,
    -> current_date;
```

실행결과

```
+----------------+--------------+
| user()         | current_date |
+----------------+--------------+
| root@localhost | 2021-09-30   |
+----------------+--------------+
1 row in set (0.00 sec)
```

- **command 취소**
  긴 쿼리를 작성하다가 중간에 취소해야하는 경우 `\c`를 붙여주면 된다!

```
MariaDB [(none)]> select
    -> user()
    -> \c
MariaDB [(none)]>
```

- **Database 사용**
  현재 서버에 존재하는 DB에서 찾아보기 위해서 `SHOW statement` 사용

```
MariaDB [(none)]> show databases;
```

실행결과

```
+--------------------+
| Database           |
+--------------------+
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| test               |
| webdb              |
+--------------------+
6 rows in set (0.00 sec)
```

Workbench에서 실습

- **local infile 설정**
  Workbench 홈 화면 > MySQL Connections - webdb 우클릭 > Edit Connection ... > Connection - Advanced - Others > OPT_LOCAL_INFILE=1 추가 > Close
- **실습**

```
-- Basic Query
-- 테이블 만들기
create table pet(
   name varchar(20),
   owner varchar(20),
   species varchar(20),
   gender char(1),
   birth DATE,
   death DATE
);

-- 테이블 삭제
drop table pet;

-- scheme 확인
desc pet;

-- 조회
select name, owner, species, gender, birth, death from pet;

-- 데이터 넣기(생성, create)
insert into pet value ('별이', '김주의', 'dog', 'w', '2016-11-20', null);

-- 데이터 삭제(delete)
delete from pet where name = '별이';

-- load data local infile
load data local infile 'C:/douzone2021-eui/eclipse-workspace/mariadb-
practices/sql-practices/docs/pet.txt' into table pet;

-- update death
update pet set death=null where death='0000-00-00';

-- 조회연습1: where
```

```sql
-- 1990년 이후에 태어난 아이들은?
select name, species, birth from pet where birth > '1990-12-31';

-- Gwen과 함께사는 아이들은?
select name, species, owner from pet where owner = 'Gwen';

-- null 다루기 1 : 살아있는 애들은?
select name, birth, death from pet where death is null;

-- null 다루기 2 : 죽은 애들은?
select name, birth, death from pet where death is not null;

-- like 검색(패턴 매칭) : 이름이 b로 시작하는 아이들은?
select name from pet where name like 'b%';

-- like 검색(패턴 매칭) : 이름이 b로 시작하는 아이들중에 이름이 6자인 아이는?
select name from pet where name like 'b_____';

-- 집계(통계) 함수
select count(*) from pet;

select count(death) from pet;   -- null이 아닌 애들만 count
select count(*) from pet where death is not null;
```

hr 계정 생성 및 employees 불러오기

1. emplayees_db.zip 가져오기

```
[C:\~]$ sftp webmaster@127.0.0.1

sftp:/home/webmaster> put C:\Users\kje_0\Downloads\employees_db.zip
```

> employees_db.zip 파일은 강의자료 MySQL 폴더에 있음!
> put 뒤에는 zip 파일 경로 써주기!!

2. 압축 풀기

```
$ mv /home/webmaster/employees_db.zip /root
$ yum -y install unzip
$ unzip employees_db.zip
```

3. 계정 생성 및 권한 부여

```
MariaDB[(none)]> create database employees;
MariaDB[(none)]> create user 'hr'@'10.0.2.2' identified by 'hr';
MariaDB[(none)]> grant all privileges on employees.* to 'hr'@'10.0.2.2';
MariaDB[(none)]> flush privileges;

MariaDB[(none)]> create user 'hr'@'localhost' identified by 'hr';
MariaDB[(none)]> grant all privileges on employees.* to 'hr'@'localhost';
MariaDB[(none)]> flush privileges;
```

> `'hr'@'localhost'` - 쉘에서 mariadb 접근 가능
> `'hr'@'10.0.2.2'` - Workbench에서 접근 가능

4. mariaDB 로그인 후 SQL 파일 실행

```
$ cd employees_db
$ mysql -p < employees.sql
```

5. Workbench 접속

New Connecion(+ 선택)

> Connection Name: hr
> Port: 3306 (빌드 환경 설정 단계 `DMYSQL_TCP_PORT=포트번호`에서 작성한 포트번호)
> Username: hr
> Password - Store in Vault ... : hr(mariadb user생성에서 작성한 비밀번호)