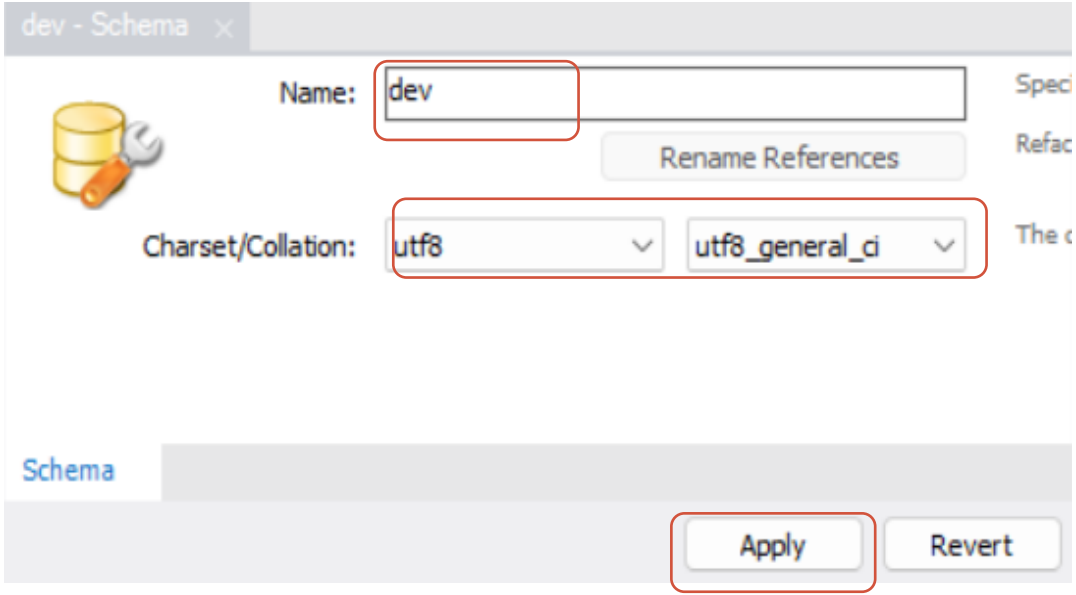
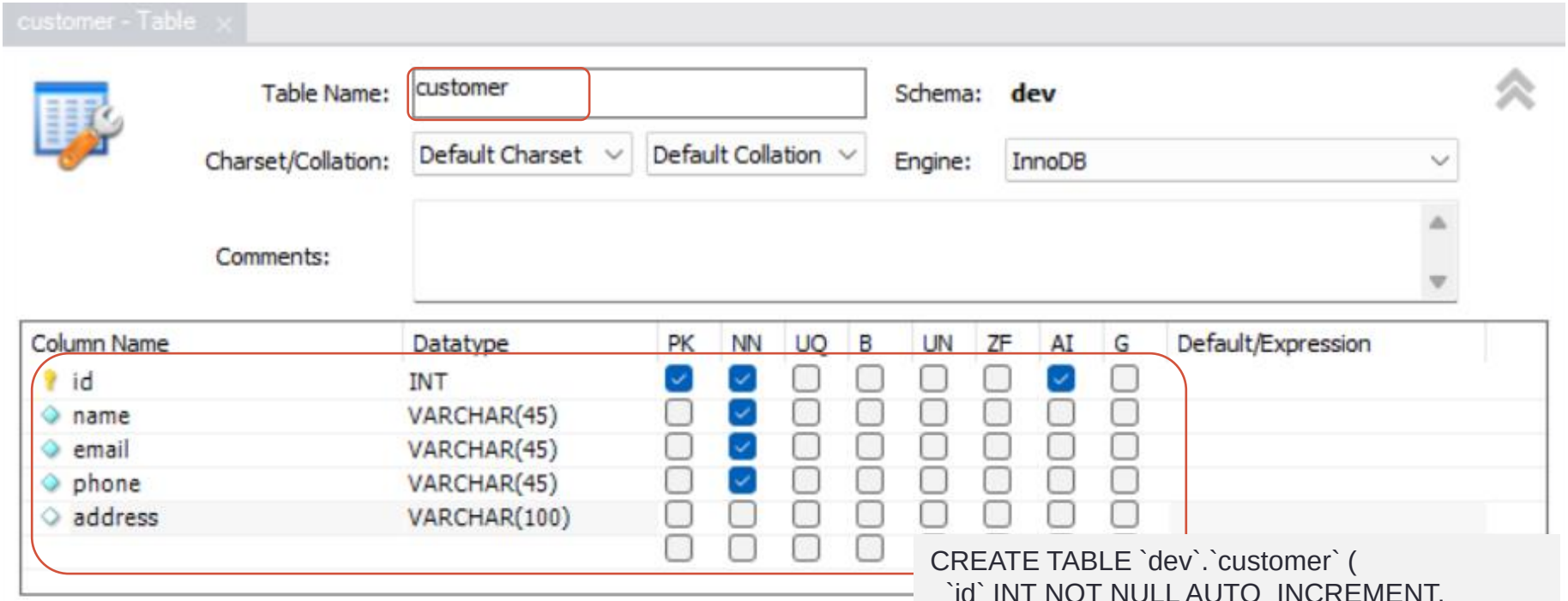
- 관계형 데이터베이스 관리 시스템(RDBMS)
- MySQL
  - GUI 관리툴은 내장되어 있지 않음. 별도로 설치
  - CLI 명령으로 데이터베이스를 관리하고, 데이터를 백업, 상태를 검사, 데이터베이스 구조를 생성, 또는 데이터 레코더를 작성하는 명령어를 이용
- MySQL 워크벤치
  - 공식적인 MySQL 프론트엔드 툴로서 오라클에 의해 개발되었으며, 자유롭게 사용할 수 있다

데이터베이스 생성




```
CREATE SCHEMA `dev` DEFAULT CHARACTER SET utf8 ;
```

테이블 생성




```
CREATE TABLE `dev`.`customer` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  `phone` VARCHAR(45) NOT NULL,
  `address` VARCHAR(100) NULL,
  PRIMARY KEY (`id`));
```

- 데이터 입력

```
insert into dev.customer(id, name, email, phone, address)
values(1, '홍길동', 'hong@gmail.com', '010-111-2222', '');
```

```
insert into dev.customer
set id=1,
    name='홍길동',
    email='hong@gmail.com',
    phone= '010-111-2222'
```

set 절은 mysql만 가능
다른 DBMS는 안됨

- 데이터 수정

```
update dev.customer
set name='홍길동',
    email='hong@gmail.com',
    phone= '010-111-2222'
where id = 1
```

- **데이터 삭제**

```
delete from dev.customer where id = 1;
```

- **데이터 조회**

```
select * from dev.customer

select * from dev.customer where id = 1

select * from dev.customer LIMIT 10;              // limit

select * from dev.customer LIMIT 10 offset 0   // offset, limit

select * from dev.customer LIMIT 0, 10         // offset, limit
```

최대 10개 조회

1페이지 1부터 10개
2페이지를 할꺼면
LIMIT 10 offset 10

위와 같은 내용 다른 방식

Offset 값은 0부터 시작하므로 첫 번째 행 데이터를 가리키는 값은 0 이다.

offset 계산방법(현재페이지 번호 -1)*페이지당 조회 수

mysql 모듈 사용하기

- mysql 모듈 설치

```
> npm install mysql
```

- mysql 연결
  - 사용자계정 생성

```
create user 'dev01'@'%' identified with mysql_native_password by '1234';
```

  - 권한 부여

```
grant all privileges on dev.* to 'dev01'@'%' with grant option;
flush privileges;
```

Mysql 접속

- 연결정보

```
// mysql 모듈 로드
const mysql = require("mysql");

// mysql 접속 정보
const conn = { host: "192.168.0.1",  localhost
               port: "3306",
               user: "dev01",
               password: "1234",
               database: "dev",
};

// DB 커넥션 생성
let connection = mysql.createConnection(conn);
```

- 연결문자열 DBMS 별로 방식이 다름

```
let connection = mysql.createConnection( "mysql://dev01:1234@localhost:3307/dev");
```

연결 테스트 후 쿼리 수행

```
// 1. DB 접속 체크 (생략가능)
connection.connect((err) => {
  if (err) {
    console.log("error connection" + err.stack);
    return;
  }
});

// 2. SQL 실행
sql = "SELECT * FROM customer";
connection.query(sql, function (err, results, fields) {
  if (err) {
    console.log(err);
  }

// 3. 결과 처리
  console.log(results);
});

// DB 접속 종료(비동기이지만 SQL이 모두 실행되면 종료)
connection.end();
```

- DataBase Connectopn Pool
  - 데이터베이스에 연결된 Connection을 미리 만들어 둔후 Pool에 보관하였다가 필요할 때 Pool에서 Connection을 가져다 사용한 후, 다시 Pool에 반환하는 기법 서버성능 향상
  - 새로운 connection을 구축할 때 생기는 오버헤드를 모두 피할 수 있음

```
const mysql = require("mysql");

// mysql 접속 정보
const conn = {
  host: "192.168.0.1",
  port: "3306",
  user: "dev01",
  password: "1234",
  database: "dev",
  connectionLimit: 10,
};

let pool = mysql.createPool(conn);
```

위와 달리 추가 된 내용

connection 콜백 함수 이용

```
let pool = mysql.createPool(conn);

// 1. DB 접속(콜백함수)
pool.getConnection((err, connetion) => {

// 2. SQL 실행
  sql = "SELECT * FROM customer";
  connetion.query(sql, (err, results, fields) => {
    if (err) {
      console.log(err);
    }

    // 3. 결과 처리
    console.log(results);

    // 4. DB 접속 종료
    connetion.release();
  });
  // pool.releaseConnection(conn); connecion을 직접 release 하거나 pool의 release를 사용
});
```

POOL은 END가 아니라 RELEASE

pool을 이용하여 쿼리실행

간단한 방법

```
let pool = mysql.createPool(conn);

//1. 풀 connection은 생략 가능

// 2. SQL 실행
sql = "SELECT * FROM customer";
pool.query(sql, function (err, results, fields) {
  if (err) {
    console.log(err);
  }

  // 3. 결과 처리
  console.log(results);

 //4. 쿼리가 수행되면 connection은 자동으로 해제된다.
});
```

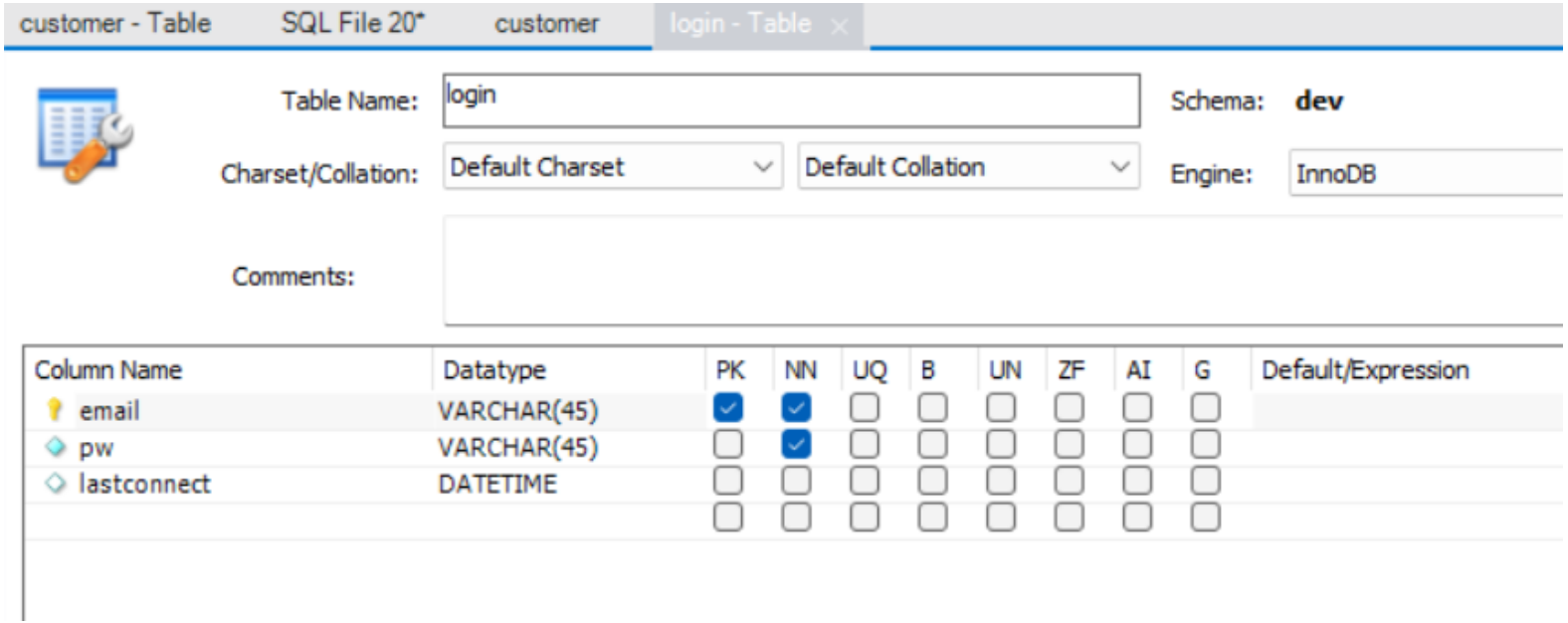
?가 들어가면 ? 수만큼 순서대로 밑에 sql과 function 사이에data값 넣어줘야 됨

connectionPool 모듈화

```
const mysql = require("mysql");

// mysql 접속 정보
const conn = {
  host: "localhost",
  …
  connectionLimit: 10,
};

// DB 커넥션 생성
let pool = mysql.createPool(conn);

module.exports = pool;
```

```
const mysql = require("./mysql_pool");

sql = "SELECT * FROM customer";
mysql.pool.query(sql,
  function (err, results, fields) {
  if (err) {
    console.log(err);
  }
  console.log(results);
});
```

로그인 테이블 생성


customer - Table
SQL File 20*
customer
login - Table
Table Name:
login
Schema:
dev
Charset/Collation:
Default Charset
Default Collation
Engine:
InnoDB
Comments:
Column Name
Datatype
PK
NN
UQ
B
UN
ZF
AI
G
Default/Expression
email
VARCHAR(45)
pw
VARCHAR(45)
lastconnect
DATETIME