

- MySQL DBMS 설치
- Command-Line Client(mysql) 사용법
- sql 구문
- Workbench 사용법
- mysql 모듈 사용
- DBCP

- **관계형 데이터베이스 관리 시스템(RDBMS)**
- **MySQL Server**
  - GUI 관리툴은 내장되어 있지 않음. 별도로 설치
  - CLI 명령으로 데이터베이스를 관리하고, 데이터를 백업, 상태를 검사, 데이터베이스 구조를 생성, 또는 데이터 레코드를 작성하는 명령어를 이용
  - 다운로드 : <https://dev.mysql.com/downloads/mysql/>
- **MySQL 워크벤치 GUI client**
  - 공식적인 MySQL 프론트엔드 툴로서 오라클에 의해 개발되었으며, 자유롭게 사용할 수 있음.
  - 다운로드 : <https://dev.mysql.com/downloads/workbench/>

- Oracle DBMS와 차이
  - 데이터타입
  - 시퀀스 , auto\_increment
  - rownum, limit
  - function

```
> mysql -u root -p
```

```
Enter password:
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.
```

```
Your MySQL connection id is 8
```

```
mysql> show databases;
```

```
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
+-----+
```

```
4 rows in set (0.01 sec)
```

- 데이터베이스 생성

```
mysql> create database test  
Query OK, 1 row affected (0.01 sec)
```

- 사용자 생성

```
mysql> create user 'hr' identified with caching_sha2_password by 'hr';  
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> grant all privileges on *.* to 'hr';  
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> flush privileges;  
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> use test
Database changed
```

```
mysql> create table board ( no int primary key, title varchar(100) );
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> insert into board values ( 1, 'test1');
```

```
mysql> select * from board;
+----+-----+
| no | title |
+----+-----+
| 1  | test1 |
+----+-----+
3 rows in set (0.01 sec)
```

- 데이터 입력

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

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

- 데이터 수정

```
update 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
```

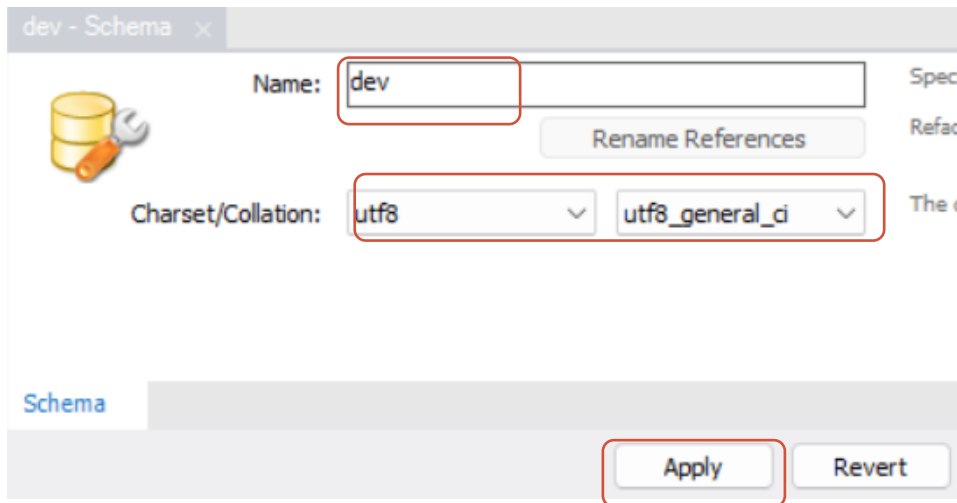
```
select * from dev.customer LIMIT 10 offset 0    // offset, limit
```

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

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




- 데이터베이스 생성



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






- 테이블 생성

customer - Table x

 Table Name:  Schema: **dev**

Charset/Collation:   Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/E
 id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 email	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 phone	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 address	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

```
CREATE TABLE `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`));
```

- mysql 모듈 설치

```
> npm install mysql2
```

- Mysql 접속

- 연결정보 객체이용

```
// mysql 모듈 로드
const mysql = require("mysql2");

// mysql 접속 정보
const conn = { host: "127.0.0.1",
                port: "3306",
                user: "hr",
                password: "1234",
                database: "test" };

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

- 연결문자열 이용

```
let connection = mysql.createConnection( "mysql://hr:1234@localhost:3306/test");
```

- 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();
```

- query 파라미터 전달

- 단건조회

```
const id = 10
sql = "SELECT * FROM customers where id=? ";
pool.query(sql, id, function (err, results, fields) {
  if (err) { console.log(err); }
  console.log(results[0]);
});
});
```

- 등록

```
router.post("/", (req, res) => {
  let sql = "insert into customers set ? ";
  pool.query(sql, req.body, function (err, results, fields) {
    if (err) { console.log(err); }
    res.json(results);
  });
});
```

- query 파라미터 전달
  - 수정

```
router.put("/:id", (req, res) => {  
  let sql = "update customers set ? where id=?";  
  let data = [req.body, req.params.id];  
  pool.query(sql, data, function (err, results, fields) {  
    let resultData = {};  
    if (err) {  
      console.log(err);  
      throw err;  
    }  
    if (results.changedRows > 0) {  
      resultData.result = true;  
      resultData.data = req.body;  
    } else {  
      resultData.result = false;  
    }  
    res.send(resultData);  
  });  
});
```

- DataBase Connectopn Pool

- 데이터베이스에 연결된 Connection을 미리 만들어 둔후 Pool에 보관하였다가 필요할 때 Pool에서 Connection을 가져다 사용한 후, 다시 Pool에 반환하는 기법
- 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);
```



- connection 콜백 함수 이용

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

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

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

// 3. 결과 처리
    console.log(results);

// 4. DB 접속 종료
    connection.release();
  });
});
```

- pool.query 들 이용하여 쿼리 실행

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

// 1. pool 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.
});
```

- pool.query 들 이용하여 쿼리 실행

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

// 1. pool 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.
});
```

- connectionPool 모듈화

filename: pool.js

```
const mysql = require("mysql2");
```

```
// mysql 접속 정보
```

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

```
// DB 커넥션 생성
```

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

```
module.exports = pool;
```

filename: customer.js

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

```
sql = "SELECT * FROM customer";
```

```
mysql.pool.query(sql,  
  function (err, results, fields) {  
    if (err) {  
      console.log(err);  
    }  
    console.log(results);  
  });
```



