

REPORT

데이터베이스

SQL실습 과제



32201805 박정민

1. DBMS 설치 내용 : MySQL 8.0

- Mysql <https://dev.mysql.com/downloads/repo/yum/> 에서 설치한 후 CentOS7에 putty에서 명령어 입력

실행 내용
yum install
https://dev.mysql.com/get/mysql80-community-release-el7-6.noarch.rpm
실행 결과
<pre>[root@localhost ~]# yum install https://dev.mysql.com/get/mysql80-community-release-el7-6.noarch.rpm Loaded plugins: fastestmirror mysql80-community-release-el7-6.noarch.rpm 11 kB 00:00 Examining /var/tmp/yum-root-sh0Arm/mysql80-community-release-el7-6.noarch.rpm: m ysql80-community-release-el7-6.noarch Marking /var/tmp/yum-root-sh0Arm/mysql80-community-release-el7-6.noarch.rpm to b e installed Resolving Dependencies --> Running transaction check --> Package mysql80-community-release.noarch 0:el7-6 will be installed --> Finished Dependency Resolution Dependencies Resolved ===== Package Arch Version Repository Size ===== Installing: mysql80-community-release noarch el7-6 /mysql80-community-release-el7-6.noarch 10 k Transaction Summary ===== Install 1 Package Total size: 10 k Installed size: 10 k Is this ok [y/d/N]: y Downloading packages: Running transaction check Running transaction test Transaction test succeeded Running transaction Installing : mysql80-community-release-el7-6.noarch 1/1 Verifying : mysql80-community-release-el7-6.noarch 1/1 Installed: mysql80-community-release.noarch 0:el7-6 Complete!</pre>

- Mysql 8.0 설치

실행 내용
yum install mysql-server
실행 결과

```
[root@localhost ~]# yum install mysql-server
Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: mirror.elice.io
 * extras: mirror.elice.io
 * updates: mirror.elice.io
base | 3.6 kB | 00:00
extras | 2.9 kB | 00:00
mysql-connectors-community | 2.6 kB | 00:00
mysql-tools-community | 2.6 kB | 00:00
mysql80-community | 2.6 kB | 00:00
updates | 2.9 kB | 00:00
(1/7): base/7/x86_64/group_gz | 153 kB | 00:00
(2/7): extras/7/x86_64/primary_db | 249 kB | 00:00
(3/7): mysql-tools-community/x86_64/primary_db | 92 kB | 00:00
(4/7): mysql-connectors-community/x86_64/primary_db | 99 kB | 00:01
(5/7): updates/7/x86_64/primary_db | 20 MB | 00:01
(6/7): mysql80-community/x86_64/primary_db | 239 kB | 00:01
(7/7): base/7/x86_64/primary_db | 6.1 MB | 00:03
Resolving Dependencies
--> Running transaction check
--> Package mysql-community-server.x86_64 0:8.0.33-1.el7 will be installed
--> Processing Dependency: mysql-community-common(x86-64) = 8.0.33-1.el7 for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Processing Dependency: mysql-community-icu-data-files = 8.0.33-1.el7 for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Processing Dependency: mysql-community-client(x86-64) >= 8.0.11 for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Processing Dependency: /usr/bin/perl for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Processing Dependency: net-tools for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Processing Dependency: perl(Getopt::Long) for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Processing Dependency: perl(strict) for package: mysql-community-server-8.0.33-1.el7.x86_64
--> Running transaction check
--> Package mysql-community-client.x86_64 0:8.0.33-1.el7 will be installed
--> Processing Dependency: mysql-community-client-plugins = 8.0.33-1.el7 for package: mysql-community-client-8.0.33-1.el7.x86_64
--> Processing Dependency: mysql-community-libs(x86-64) >= 8.0.11 for package: mysql-community-client-8.0.33-1.el7.x86_64
--> Package mysql-community-common.x86_64 0:8.0.33-1.el7 will be installed
--> Package mysql-community-icu-data-files.x86_64 0:8.0.33-1.el7 will be installed
```

• 오류 발생

```
Total 2.3 MB/s | 101 MB 00:43
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Importing GPG key 0xF4A80EB5:
Userid : "CentOS-7 Key (CentOS 7 Official Signing Key) <security@centos.org>"
Fingerprint: 6341 ab27 53d7 8a78 a7c2 7bb1 24c6 a8a7 f4a8 0eb5
Package : centos-release-7-9.2009.0.el7.centos.x86_64 (@anaconda)
From : /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Is this ok [y/N]: y
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql-2022

GPG key retrieval failed: [Errno 14] curl#37 - "Couldn't open file /etc/pki/rpm-gpg/RPM-GPG-KEY-mysql-2022"
```

- 오류 수정 : gpgcheck 옵션 확인하여 '[mysql80-community]' 항목의 'gpgcheck' 옵션을 'gpgcheck=0' 로 수정

실행 내용
vi /etc/yum.repos.d/mysql-community.repo
실행 결과

```
[root@localhost ~]# vi /etc/yum.repos.d/mysql-community.repo

[mysql80-community]
name=MySQL 8.0 Community Server
baseurl=http://repo.mysql.com/yum/mysql-8.0-community/el/7/$basearch
enabled=1
gpgcheck=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql-2022
        file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql
```

- 재설치 - 오류 수정됨

실행 내용
yum install mysql-server
실행 결과
<pre>[root@localhost ~]# yum install mysql-server Replaced: mariadb-libs.x86_64 1:5.5.68-1.el7 Complete!</pre>

- 설치된 MySQL 버전 확인

실행 내용
mysqld -V
실행 결과
<pre>[root@localhost ~]# mysqld -V /usr/sbin/mysqld Ver 8.0.33 for Linux on x86 64 (MySQL Community Server - GPL)</pre>

- MySQL 시작 및 자동 실행 등록 / 초기 비밀번호 확인

실행 내용
systemctl start mysqld
systemctl enable mysqld
grep 'temporary password' /var/log/mysqld.log
실행 결과

```
[root@localhost ~]# systemctl start mysqld
[root@localhost ~]# systemctl enable mysqld
[root@localhost ~]# grep 'temporary password' /var/log/mysqld.log
2023-05-03T07:56:27.425962Z 6 [Note] [MY-010454] [Server] A temporary password i
s generated for root@localhost: (ashcf-sn4>T
[root@localhost ~]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.33

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

- 비밀번호 'jungmin0103!'으로 변경

실행 내용
mysql -u root -p
실행 결과
<pre>[root@localhost ~]# mysql -u root -p Enter password: Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 9 Server version: 8.0.33 MySQL Community Server - GPL Copyright (c) 2000, 2023, Oracle and/or its affiliates. Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement. mysql> set global validate_password.policy=LOW; Query OK, 0 rows affected (0.00 sec) mysql> alter user 'root'@'localhost' identified by 'jungmin0103'; Query OK, 0 rows affected (0.01 sec)</pre>

2. 6장 SQL문

6.1 SQL 데이터 정의문

6.1.1 스키마와 카탈로그

- 스키마 생성 및 권한 부여

실행 내용
CREATE USER 'SHLEE' IDENTIFIED BY 'TEST1234';
GRANT CREATE ON *.* TO 'SHLEE';
CREATE SCHEMA UNIVERSITY;
GRANT ALL PRIVILEGES ON UNIVERSITY.* TO 'SHLEE';
실행 결과

```
mysql> CREATE USER 'SHLEE' IDENTIFIED BY 'TEST1234';
Query OK, 0 rows affected (0.03 sec)

mysql> GRANT CREATE ON *.* TO 'SHLEE';
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE SCHEMA UNIVERSITY;
Query OK, 1 row affected (0.02 sec)

mysql> GRANT ALL PRIVILEGES ON UNIVERSITY.* TO 'SHLEE';
Query OK, 0 rows affected (0.03 sec)

mysql> SHOW DATABASES;
+-----+
| Database                |
+-----+
| UNIVERSITY              |
| information_schema      |
| mysql                   |
| performance_schema      |
| sys                     |
+-----+
5 rows in set (0.00 sec)
```

6.1.2 도메인 정의문

- 도메인 생성

: MYSQL 8.0에서는 DOMAIN을 직접 정의할 수 없기 때문에 CREATE TABLE을 이용하여 칼럼의 도메인 정의함

실행 내용
USE UNIVERSITY;
CREATE TABLE DEPARTMENT(DEPT CHAR(4) DEFAULT '??', CHECK (DEPT IN ('COMP','ME','EE','ARCH','???')));
DESC DEPARTMENT;
실행 결과


```
mysql> USE UNIVERSITY;
Database changed
mysql> CREATE TABLE DEPARTMENT(
    -> DEPT CHAR(4) DEFAULT '???',
    -> CHECK (DEPT IN ('COMP', 'ME', 'EE', 'ARCH', '???'))
    -> );
Query OK, 0 rows affected (0.95 sec)

mysql> DESC DEPARTMENT
    -> ;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| DEPT  | char(4) | YES  |     | ???     |       |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.05 sec)
```

- 도메인 삭제
- : 도메인을 삭제할 때에는 그 테이블을 삭제하면 도메인도 삭제된다.

실행 내용
DROP TABLE DEPARTMENT;
DESC DEPARTMENT;
실행 결과
<pre>mysql> DROP TABLE DEPARTMENT; Query OK, 0 rows affected (0.20 sec) mysql> DESC DEPARTMENT; ERROR 1146 (42S02): Table 'UNIVERSITY.DEPARTMENT' doesn't exist</pre>

6.1.3 기본 테이블의 생성

- 대학 데이터베이스에 학생, 과목, 등록 테이블을 각각 생성
- 학생 테이블

실행 내용
USE UNIVERSITY;
CREATE TABLE STUDENT
(Sno INT NOT NULL,
Sname CHAR(12),
Year INT,
Dept CHAR(12),
PRIMARY KEY(Sno));
DESC STUDENT;
실행 결과

```
mysql> CREATE TABLE STUDENT
-> (Sno INT NOT NULL,
-> Sname CHAR(12),
-> Year INT,
-> Dept CHAR(12),
-> PRIMARY KEY(Sno));
Query OK, 0 rows affected (0.13 sec)

mysql> DESC STUDENT;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int       | NO   | PRI | NULL    |       |
| Sname | char(12)  | YES  |     | NULL    |       |
| Year  | int       | YES  |     | NULL    |       |
| Dept  | char(12)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.01 sec)
```

• 과목 테이블

실행 내용
USE UNIVERSITY;
CREATE TABLE COURSE
(Cno CHAR(6) NOT NULL,
Cname CHAR(12),
Credit INT,
Dept CHAR(12),
PRname CHAR(12),
PRIMARY KEY(Cno));
DESC COURSE;
실행 결과


```
mysql> USE UNIVERSITY;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CREATE TABLE COURSE
  -> (Cno INT NOT NULL,
  -> Cname CHAR(12),
  -> Credit INT,
  -> Dept CHAR(12),
  -> PRname CHAR(12),
  -> PRIMARY KEY(Cno));
Query OK, 0 rows affected (0.09 sec)

mysql> DESC COURSE;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| Cno   | int    | NO   | PRI | NULL    |       |
| Cname | char(12) | YES  |     | NULL    |       |
| Credit | int    | YES  |     | NULL    |       |
| Dept  | char(12) | YES  |     | NULL    |       |
| PRname | char(12) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

- 등록 테이블

실행 내용
<pre>USE UNIVERSITY; CREATE TABLE ENROL (Sno INT NOT NULL, Cno CHAR(6) NOT NULL, Grade INT, PRIMARY KEY(Sno, Cno), FOREIGN KEY(Sno) REFERENCES STUDENT(Sno) ON DELETE CASCADE ON UPDATE CASCADE, FOREIGN KEY(Cno) REFERENCES COURSE(Cno) ON DELETE CASCADE ON UPDATE CASCADE, CHECK(Grade>=0 AND Grade<=100)); DESC ENROL;</pre>
실행 결과

```
mysql> USE UNIVERSITY;
Database changed
mysql> CREATE TABLE ENROL
  -> (Sno INT NOT NULL,
  -> Cno INT NOT NULL,
  -> GRADE INT,
  -> PRIMARY KEY(Sno, Cno),
  -> FOREIGN KEY(Sno) REFERENCES STUDENT(Sno)
  -> ON DELETE CASCADE
  -> ON UPDATE CASCADE,
  -> FOREIGN KEY(Cno) REFERENCES COURSE(Cno)
  -> ON DELETE CASCADE
  -> ON UPDATE CASCADE,
  -> CHECK(Grade>=0 AND Grade<=100));
Query OK, 0 rows affected (0.13 sec)

mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int  | NO   | PRI | NULL    |       |
| Cno   | int  | NO   | PRI | NULL    |       |
| GRADE | int  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)
```

6.1.4 기본 테이블의 제거와 변경

- 기본 테이블 제거
- DROP TABLE CASCADE로 과목 테이블을 제거하려했더니 등록 테이블에 외래키가 참조되어있어 외래키 제약조건 삭제 후 과목 테이블 제거

실행 내용

실행 결과

```
mysql> DROP TABLE COURSE CASCADE;
ERROR 3730 (HY000): Cannot drop table 'COURSE' referenced by a foreign key constraint 'ENROL_ibfk_2' on table 'ENROL'.
```

```
mysql> ALTER TABLE ENROL
  -> DROP FOREIGN KEY ENROL_ibfk_2;
Query OK, 0 rows affected (0.10 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> DROP TABLE COURSE;
Query OK, 0 rows affected (0.07 sec)

mysql> DESC COURSE;
ERROR 1146 (42S02): Table 'UNIVERSITY.COURSE' doesn't exist
```

- 스키마 제거

실행 내용
DROP DATABASE UNIVERSITY; SHOW DATABASES;
실행 결과
<pre>mysql> DROP DATABASE UNIVERSITY; Query OK, 2 rows affected (0.13 sec) mysql> SHOW DATABASES; +-----+ Database +-----+ information_schema mysql performance_schema sys +-----+ 4 rows in set (0.00 sec)</pre>

- 기본 테이블 변경 및 제거
- 기본 테이블 변경

실행 내용
DESC ENROL;
ALTER TABLE ENROL ADD Final CHAR DEFAULT 'F';
DESC ENROL;
실행 결과

```
mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int  | NO   | PRI | NULL    |       |
| Cno   | int  | NO   | PRI | NULL    |       |
| GRADE | int  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> ALTER TABLE ENROL
    -> ADD Final CHAR DEFAULT 'F';
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int       | NO   | PRI | NULL    |       |
| Cno   | int       | NO   | PRI | NULL    |       |
| GRADE | int       | YES  |     | NULL    |       |
| Final | char(1)   | YES  |     | F       |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

- 기본 테이블 제거

실행 내용
DESC ENROL;
ALTER TABLE ENROL DROP Grade CASCADE;
DESC ENROL;
실행 결과

```
mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int       | NO   | PRI | NULL    |       |
| Cno   | int       | NO   | PRI | NULL    |       |
| Grade | int       | YES  |     | NULL    |       |
| Final | char(1)   | YES  |     | F       |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> ALTER TABLE ENROL DROP Grade CASCADE;
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int       | NO   | PRI | NULL    |       |
| Cno   | int       | NO   | PRI | NULL    |       |
| Final | char(1)   | YES  |     | F       |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

- 특정 열 제약조건 변경, 제거

실행 내용
ALTER TABLE ENROL ALTER Grade SET DEFAULT '0'; ALTER TABLE ENROL ALTER Grade DROP DEFAULT;
DESC ENROL;
실행 결과

```
mysql> ALTER TABLE ENROL ALTER Grade SET DEFAULT '0';
Query OK, 0 rows affected (0.03 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int    | NO   | PRI | NULL    |       |
| Cno   | int    | NO   | PRI | NULL    |       |
| Final | char(1) | YES  |     | F        |       |
| Grade | int    | YES  |     | 0        |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> ALTER TABLE ENROL ALTER Grade DROP DEFAULT;
Query OK, 0 rows affected (0.04 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> DESC ENROL;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int    | NO   | PRI | NULL    |       |
| Cno   | int    | NO   | PRI | NULL    |       |
| Final | char(1) | YES  |     | F        |       |
| Grade | int    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

6.2 SQL 데이터 조작문

- 학생, 과목, 등록 테이블 사용
- 학생 테이블 / 과목 테이블

```
mysql> USE UNIVERSITY;
Database changed
mysql> DESC STUDENT;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Sno   | int    | NO   | PRI | NULL    |       |
| Sname | char(12) | YES  |     | NULL    |       |
| Year  | int    | YES  |     | NULL    |       |
| Dept  | char(12) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> INSERT INTO STUDENT VALUES(100,'나수열',4,'컴퓨터');
Query OK, 1 row affected (0.04 sec)

mysql> INSERT INTO STUDENT VALUES(200,'이찬수',3,'전기');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO STUDENT VALUES(300,'정기태',1,'컴퓨터');
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO STUDENT VALUES(400,'송병길',4,'컴퓨터');
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO STUDENT VALUES(500,'박종화',2,'산공');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM STUDENT;
+-----+-----+-----+-----+
| Sno | Sname | Year | Dept |
+-----+-----+-----+-----+
| 100 | 나수열 | 4    | 컴퓨터 |
| 200 | 이찬수 | 3    | 전기 |
| 300 | 정기태 | 1    | 컴퓨터 |
| 400 | 송병길 | 4    | 컴퓨터 |
| 500 | 박종화 | 2    | 산공 |
+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

```
mysql> INSERT INTO COURSE VALUES('C123','프로그래밍',3,'컴퓨터','김성국');
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO COURSE VALUES('C312','자료구조',3,'컴퓨터','황수관');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO COURSE VALUES('C324','화일구조',3,'컴퓨터','이규찬');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO COURSE VALUES('C413','데이터베이스',3,'컴퓨터','이필로');
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO COURSE VALUES('E412','반도체',3,'전자','홍봉진');
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM COURSE;
+-----+-----+-----+-----+-----+
| Cno | Cname | Credit | Dept | PRname |
+-----+-----+-----+-----+-----+
| C123 | 프로그래밍 | 3 | 컴퓨터 | 김성국 |
| C312 | 자료구조 | 3 | 컴퓨터 | 황수관 |
| C324 | 화일구조 | 3 | 컴퓨터 | 이규찬 |
| C413 | 데이터베이스 | 3 | 컴퓨터 | 이필로 |
| E412 | 반도체 | 3 | 전자 | 홍봉진 |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```


• 등록 테이블

```
mysql> CREATE TABLE ENROL
-> (Sno INT NOT NULL,
-> Cno CHAR(6) NOT NULL,
-> Grade CHAR(6),
-> Midterm INT,
-> Final INT,
-> PRIMARY KEY(Sno, Cno),
-> FOREIGN KEY(Sno) REFERENCES STUDENT(Sno)
-> ON DELETE CASCADE
-> ON UPDATE CASCADE,
-> FOREIGN KEY(Cno) REFERENCES COURSE(Cno)
-> ON DELETE CASCADE
-> ON UPDATE CASCADE,
-> CHECK(Midterm>=0 AND Midterm<=100),
-> CHECK(Final>=0 AND Final<=100));
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> INSERT INTO ENROL VALUES(300,'C312','A',90,95);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(300,'C324','C',75,75);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(300,'C413','A',95,90);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(400,'C312','A',90,95);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(400,'C324','A',95,90);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(400,'C413','B',80,85);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(400,'E412','C',65,75);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ENROL VALUES(500,'C312','B',85,80);
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM ENROL;
+-----+-----+-----+-----+-----+
| Sno | Cno | Grade | Midterm | Final |
+-----+-----+-----+-----+-----+
| 100 | C413 | A | 90 | 95 |
| 100 | E412 | A | 95 | 95 |
| 200 | C123 | B | 85 | 80 |
| 300 | C312 | A | 90 | 95 |
| 300 | C324 | C | 75 | 75 |
| 300 | C413 | A | 95 | 90 |
| 400 | C312 | A | 90 | 95 |
| 400 | C324 | A | 95 | 90 |
| 400 | C413 | B | 80 | 85 |
| 400 | E412 | C | 65 | 75 |
| 500 | C312 | B | 85 | 80 |
+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

6.2.1 데이터 검색

- 컴퓨터과 학생의 이름(Sname)과 학번(Sno)을 검색하라.

실행 내용
SELECT Sname, Sno FROM STUDENT WHERE Dept='컴퓨터';
SELECT STUDENT.Sname, STUDENT.Sno FROM STUDENT WHERE STUDENT.Dept='컴퓨터';
실행 결과

```
mysql> SELECT Sname, Sno
-> FROM STUDENT
-> WHERE Dept='컴 퓨 터 ';

+-----+-----+
| Sname   | Sno   |
+-----+-----+
| 나 수 영 | 100   |
| 정 기 태 | 300   |
| 송 병 길 | 400   |
+-----+-----+
3 rows in set (0.01 sec)

mysql> SELECT STUDENT.Sname, STUDENT.Sno
-> FROM STUDENT
-> WHERE STUDENT.Dept='컴 퓨 터 ';

+-----+-----+
| Sname   | Sno   |
+-----+-----+
| 나 수 영 | 100   |
| 정 기 태 | 300   |
| 송 병 길 | 400   |
+-----+-----+
3 rows in set (0.00 sec)
```

(1) 검색 결과에 중복 레코드의 제거

- 학생(STUDENT) 테이블에 어떤 학과(Dept)들이 있는지 검색하라.

실행 내용

SELECT DISTINCT Dept

FROM STUDENT;

실행 결과

```
mysql> SELECT DISTINCT Dept
-> FROM STUDENT;

+-----+
| Dept   |
+-----+
| 컴 퓨 터 |
| 전 기   |
| 산 공   |
+-----+
3 rows in set (0.01 sec)
```

(2) 테이블의 열 전부를 검색하는 경우

- 학생(STUDENT) 테이블 전부를 검색하라.

실행 내용

SELECT * FROM STUDENT;

SELECT Sno, Sname, Year, Dept FROM STUDENT;

실행 결과

mysql> SELECT * FROM STUDENT;

Sno	Sname	Year	Dept
100	나 수 영	4	컴 퓨 터
200	이 찬 수	3	전 기
300	정 기 태	1	컴 퓨 터
400	송 병 길	4	컴 퓨 터
500	박 종 화	2	산 공

5 rows in set (0.00 sec)

mysql> SELECT Sno, Sname, Year, Dept FROM STUDENT;

Sno	Sname	Year	Dept
100	나 수 영	4	컴 퓨 터
200	이 찬 수	3	전 기
300	정 기 태	1	컴 퓨 터
400	송 병 길	4	컴 퓨 터
500	박 종 화	2	산 공

5 rows in set (0.00 sec)

(3) 조건 검색

- 학생(STUDENT) 테이블 학과(Dept)가 '컴퓨터'이고 학년(Year)이 4인 학생의 학번(Sno)과 이름(Sname)을 검색하라.

실행 내용
SELECT Sno, Sname FROM STUDENT WHERE Dept='컴퓨터' AND Year=4;
실행 결과
<pre>mysql> SELECT Sno, Sname -> FROM STUDENT -> WHERE Dept='컴 퓨 터 ' AND Year=4; +-----+-----+ Sno Sname +-----+-----+ 100 나 수 영 400 송 병 길 +-----+-----+ 2 rows in set (0.00 sec)</pre>

(4) 순서를 명세하는 검색

- 등록(ENROL) 테이블에서 중간 성적(Midterm)이 90점 이상인 학생의 학번(Sno)과 과목 번호(Cno)를 검색하되 학번(Sno)에 대해서는 내림차순으로, 또 같은 학번에 대해서는 과목 번호(Cno)의 오름차순으로 검색하라.

실행 내용
SELECT Sno, Cno FROM ENROL WHERE Midterm>=90 ORDER BY Sno DESC, Cno ASC;
실행 결과
<pre>mysql> SELECT Sno, Cno -> FROM ENROL -> WHERE Midterm>=90 -> ORDER BY Sno DESC, Cno ASC; +-----+-----+ Sno Cno +-----+-----+ 400 C312 400 C324 300 C312 300 C413 100 C413 100 E412 +-----+-----+ 6 rows in set (0.00 sec)</pre>

(5) 산술식, 문자 스트링, 새로운 열 이름이 명세된 검색

- 등록(ENROL) 테이블에서 과목 번호(Cno)가 'C312'인 중간 성적(Midterm)에 3점을 더한 점수를 '학번', '중간성적='이란 텍스트 내용을 '시험', 그리고 '점수'라는 열 이름으로 검색하라.

실행 내용
SELECT Sno AS '학번', '중간시험=' AS '시험', Midterm+3 AS '점수' FROM ENROL WHERE Cno='C312';
실행 결과
<pre>mysql> SELECT Sno AS '학 번 ', '중 간 시 험 =' AS '시 험 ', Midterm+3 AS '점 수 ' -> FROM ENROL -> WHERE Cno='C312'; +-----+-----+-----+ 학 번 시 험 점 수 +-----+-----+-----+ 300 중 간 시 험 = 93 400 중 간 시 험 = 93 500 중 간 시 험 = 88 +-----+-----+-----+ 3 rows in set (0.00 sec)</pre>

(6) 복수 테이블로부터의 검색

- 과목 번호(Cno) 'C413'에 등록한 학생의 이름(Sname), 학과(Dept), 성적(Grade)을 검색하라.

실행 내용
SELECT S.Sname, S.Dept, E.Grade FROM STUDENT S, ENROL E WHERE S.Sno=E.Sno AND E.Cno='C413';
실행 결과
<pre>mysql> SELECT S.Sname, S.Dept, E.Grade -> FROM STUDENT S, ENROL E -> WHERE S.Sno=E.Sno AND E.Cno='C413'; +-----+-----+-----+ Sname Dept Grade +-----+-----+-----+ 나 수 영 컴 퓨 터 A 정 기 태 컴 퓨 터 A 송 병 길 컴 퓨 터 B +-----+-----+-----+ 3 rows in set (0.00 sec)</pre>

• JOIN문

실행 내용
SELECT Sname, Dept, Grade FROM STUDENT JOIN ENROL ON (STUDENT.Sno=ENROL.Sno) WHERE ENROL.Cno='C413';
SELECT Sname, Dept, Grade FROM STUDENT JOIN ENROL USING(Sno) WHERE ENROL.Cno='C413';
SELECT Sname, Dept, Grade FROM STUDENT NATURAL JOIN ENROL WHERE ENROL.Cno='C413';
실행 결과

```
mysql> SELECT Sname, Dept, Grade
-> FROM STUDENT JOIN ENROL ON (STUDENT.Sno=ENROL.Sno)
-> WHERE ENROL.Cno='C413';
```

Sname	Dept	Grade
나 수 영	컴 퓨 터	A
정 기 태	컴 퓨 터	A
송 병 길	컴 퓨 터	B

```
3 rows in set (0.00 sec)
```

```
mysql> SELECT Sname, Dept, Grade
-> FROM STUDENT JOIN ENROL USING(Sno)
-> WHERE ENROL.Cno='C413';
```

Sname	Dept	Grade
나 수 영	컴 퓨 터	A
정 기 태	컴 퓨 터	A
송 병 길	컴 퓨 터	B

```
3 rows in set (0.00 sec)
```

```
mysql> SELECT Sname, Dept, Grade
-> FROM STUDENT NATURAL JOIN ENROL
-> WHERE ENROL.Cno='C413';
```

Sname	Dept	Grade
나 수 영	컴 퓨 터	A
정 기 태	컴 퓨 터	A
송 병 길	컴 퓨 터	B

```
3 rows in set (0.00 sec)
```

(7) 자기 자신의 테이블에 조인하는 검색

- 같은 학과 학생들의 학번을 쌍으로 검색하라. 단, 첫 번째 학번은 두 번째 학번보다 작게하라.

실행 내용
SELECT S1.Sno, S2.Sno FROM STUDENT S1, STUDENT S2 WHERE S1.Dept=S2.DEPT AND S1.Sno<S2.Sno;
실행 결과


```
mysql> SELECT S1.Sno, S2.Sno
-> FROM STUDENT S1, STUDENT S2
-> WHERE S1.Dept=S2.DEPT
-> AND S1.Sno<S2.Sno;
+-----+-----+
| Sno | Sno |
+-----+-----+
| 100 | 300 |
| 100 | 400 |
| 300 | 400 |
+-----+-----+
3 rows in set (0.02 sec)
```

(8) 집계 함수를 이용한 검색

- 학생 테이블에 학생 수가 얼마인가를 검색하라.

실행 내용

```
SELECT COUNT(*) AS 학생수 FROM STUDENT;
```

실행 결과

```
mysql> SELECT COUNT(*) AS 학생수 FROM STUDENT;
+-----+
| 학생수 |
+-----+
|      5 |
+-----+
1 row in set (0.11 sec)
```

- 학번(Sno)이 300인 학생이 등록한 과목(Cno)이 몇 개인가?

실행 내용

```
SELECT COUNT(DISTINCT Cno)
```

```
FROM ENROL
```

```
WHERE Sno=300;
```

실행 결과

```
mysql> SELECT COUNT(DISTINCT Cno)
-> FROM ENROL
-> WHERE Sno=300;
+-----+
| COUNT(DISTINCT Cno) |
+-----+
|          3 |
+-----+
1 row in set (0.01 sec)
```

- 과목 'C413'에 대한 중간 성적의 평균은 얼마인가?

실행 내용
SELECT AVG(Midterm) AS 중간평균 FROM ENROL WHERE Cno='C413';
실행 결과
<pre>mysql> SELECT AVG(Midterm) AS 중간평균 -> FROM ENROL -> WHERE Cno='C413'; +-----+ 중간평균 +-----+ 88.3333 +-----+ 1 row in set (0.00 sec)</pre>

(9) GROUP BY를 이용한 검색

- 과목별 기말 성적(Final)의 평균을 검색하라.

실행 내용
SELECT Cno, AVG(Final) AS 기말평균 FROM ENROL GROUP BY Cno;
실행 결과
<pre>mysql> SELECT Cno, AVG(Final) AS 기말평균 -> FROM ENROL -> GROUP BY Cno; +-----+-----+ Cno 기말평균 +-----+-----+ C123 80.0000 C312 90.0000 C324 82.5000 C413 90.0000 E412 85.0000 +-----+-----+ 5 rows in set (0.00 sec)</pre>

(10) HAVING을 사용한 검색

- 3명 이상 등록한 과목의 기말 평균 성적을 검색하라.

실행 내용
SELECT Cno, AVG(Final) AS 평균 FROM ENROL

GROUP BY Cno HAVING COUNT(*)>=3;
실행 결과
<pre>mysql> SELECT Cno, AVG(Final) AS 평균 -> FROM ENROL -> GROUP BY Cno -> HAVING COUNT (*) >=3; +-----+-----+ Cno 평균 +-----+-----+ C312 90.0000 C413 90.0000 +-----+-----+ 2 rows in set (0.00 sec)</pre>

(11) 부속 질의문(Subquery)을 사용한 검색

- 과목 번호(Cno) 'C413' 등록한 학생 이름(Sname)을 검색하라.

실행 내용
<pre>SELECT Sname FROM STUDENT WHERE Sno IN (SELECT Sno FROM ENROL WHERE Cno='C413');</pre>
실행 결과
<pre>mysql> SELECT Sname -> FROM STUDENT -> WHERE Sno IN -> (SELECT Sno -> FROM ENROL -> WHERE Cno='C413'); +-----+ Sname +-----+ 나 수 영 정 기 태 송 병 길 +-----+ 3 rows in set (0.00 sec)</pre>

- 부속 질의문 IN, JOIN문

실행 내용
SELECT Sname FROM STUDENT WHERE Sno IN (100,300,400); SELECT STUDENT.Sname FROM STUDENT, ENROL WHERE STUDENT.Sno=ENROL.Sno AND ENROL.Cno='C413';
실행 결과
<pre>mysql> SELECT Sname -> FROM STUDENT -> WHERE Sno IN (100,300,400); +-----+ Sname +-----+ 나 수 영 정 기 태 송 병 길 +-----+ 3 rows in set (0.00 sec) mysql> mysql> mysql> SELECT STUDENT.Sname -> FROM STUDENT, ENROL -> WHERE STUDENT.Sno=ENROL.Sno -> AND ENROL.Cno='C413'; +-----+ Sname +-----+ 나 수 영 정 기 태 송 병 길 +-----+ 3 rows in set (0.00 sec)</pre>

- 과목 번호 'C413'에 등록하지 않은 학생의 이름을 검색하라.

실행 내용
SELECT Sname FROM STUDENT WHERE Sno NOT IN (SELECT Sno FROM ENROL WHERE Cno='C413');
실행 결과

```
mysql> SELECT Sname
-> FROM STUDENT
-> WHERE Sno NOT IN
-> (SELECT Sno
-> FROM ENROL
-> WHERE Cno='C413');
+-----+
| Sname |
+-----+
| 이 찬 수 |
| 박 종 화 |
+-----+
2 rows in set (0.01 sec)
```

- 학생 '정기태'와 같은 학과에 속하는 학생의 이름과 학과를 검색하라.

실행 내용

```
SELECT Sname, Dept
FROM STUDENT
WHERE Dept=
(SELECT Dept
FROM STUDENT
WHERE Sname='정기태');
```

실행 결과

```
mysql> SELECT Sname, Dept
-> FROM STUDENT
-> WHERE Dept=
-> (SELECT Dept
-> FROM STUDENT
-> WHERE Sname='정 기 태 ');
+-----+-----+
| Sname | Dept |
+-----+-----+
| 나 수 영 | 컴 퓨 터 |
| 정 기 태 | 컴 퓨 터 |
| 송 병 길 | 컴 퓨 터 |
+-----+-----+
3 rows in set (0.01 sec)
```

- 등록(ENROL)테이블에서 학번이 500인 학생의 모든 기말 성적보다 좋은 학기말 성적을 받은 학생의 학번과 과목번호를 검색하라.

실행 내용

```
SELECT Sno, Cno
```

```
FROM ENROL
WHERE Final > ALL
(SELECT Final
FROM ENROL
WHERE Sno=500);
```

실행 결과

```
mysql> SELECT Sno, Cno
-> FROM ENROL
-> WHERE Final > ALL
-> (SELECT Final
-> FROM ENROL
-> WHERE Sno=500);
+-----+-----+
| Sno | Cno |
+-----+-----+
| 100 | C413 |
| 100 | E412 |
| 300 | C312 |
| 300 | C413 |
| 400 | C312 |
| 400 | C324 |
| 400 | C413 |
+-----+-----+
7 rows in set (0.01 sec)
```

(12) LIKE를 사용하는 검색

- 과목번호(Cno)가 C로 시작하는 과목의 과목 번호와 과목이름(Cname)을 검색하라.

실행 내용

```
SELECT Cno, Cname
FROM COURSE
WHERE Cno LIKE 'C%';
```

실행 결과

```
mysql> SELECT Cno, Cname
-> FROM COURSE
-> WHERE Cno LIKE 'C%';
+-----+-----+
| Cno | Cname |
+-----+-----+
| C123 | 프 로 그 래 밍 |
| C312 | 자 료 구 조 |
| C324 | 화 일 구 조 |
| C413 | 데 이 터 베 이 스 |
+-----+-----+
4 rows in set (0.00 sec)
```


(13) NULL을 사용한 검색

- 학과(Dept)가 NULL인 학생의 학번과 이름을 검색하라.

실행 내용
SELECT Sno, Sname FROM STUDENT WHERE Dept IS NULL;
실행 결과
<pre>mysql> INSERT INTO STUDENT VALUES(600, '김 길 동 ', NULL, NULL); Query OK, 1 row affected (0.01 sec) mysql> SELECT * FROM STUDENT; +-----+-----+-----+-----+ Sno Sname Year Dept +-----+-----+-----+-----+ 100 나 수 영 4 컴 퓨 터 200 이 찬 수 3 전 기 300 정 기 태 1 컴 퓨 터 400 송 병 길 4 컴 퓨 터 500 박 종 화 2 산 공 600 김 길 동 NULL NULL +-----+-----+-----+-----+ 6 rows in set (0.00 sec) mysql> mysql> mysql> SELECT Sno, Sname -> FROM STUDENT -> WHERE Dept IS NULL; +-----+-----+ Sno Sname +-----+-----+ 600 김 길 동 +-----+-----+ 1 row in set (0.00 sec)</pre>

(14) EXISTS를 사용하는 검색

- 과목 'C413'에 등록한 학생의 이름을 검색하라.

실행 내용
SELECT Sname FROM STUDENT WHERE EXISTS (SELECT * FROM ENROL WHERE Sno=STUDENT.Sno AND Cno='C413');
실행 결과

```
mysql> SELECT Sname
-> FROM STUDENT
-> WHERE EXISTS
-> (SELECT * FROM ENROL WHERE Sno=STUDENT.Sno AND Cno='C413');
+-----+
| Sname |
+-----+
| 나 수 영 |
| 정 기 태 |
| 송 병 길 |
+-----+
3 rows in set (0.00 sec)
```

- 과목 'C413'에 등록하지 않은 학생의 이름을 검색하라.

실행 내용
SELECT Sname FROM STUDENT WHERE NOT EXISTS (SELECT * FROM ENROL WHERE Sno=STUDENT.Sno AND Cno='C413');
실행 결과
<pre>mysql> SELECT Sname -> FROM STUDENT -> WHERE NOT EXISTS -> (SELECT * FROM ENROL WHERE Sno=STUDENT.Sno AND Cno='C413'); +-----+ Sname +-----+ 이 찬 수 박 종 화 +-----+ 2 rows in set (0.00 sec)</pre>

(15) UNION이 관련된 검색

- 3학년이거나 또는 과목 'C324'에 등록한 학생의 학번을 검색하라.

실행 내용
SELECT Sno FROM STUDENT WHERE Year=3 UNION SELECT Sno FROM ENROL WHERE Cno='C324';
실행 결과

```
mysql> SELECT Sno
      -> FROM STUDENT
      -> WHERE Year=3
      -> UNION
      -> SELECT Sno
      -> FROM ENROL
      -> WHERE Cno='C324';
+-----+
| Sno |
+-----+
| 200 |
| 300 |
| 400 |
+-----+
3 rows in set (0.03 sec)
```

6.2.2 데이터의 갱신

(1) 하나의 레코드 변경

- 학번이 300인 학생의 학년을 2로 변경하라.

실행 내용

```
UPDATE STUDENT
SET Year=2
WHERE Sno=300;
```

실행 결과

```
mysql> UPDATE STUDENT
      -> SET Year=2
      -> WHERE Sno=300;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM STUDENT;
+-----+-----+-----+-----+
| Sno | Sname   | Year | Dept   |
+-----+-----+-----+-----+
| 100 | 나 수 영 | 4    | 컴 퓨 터 |
| 200 | 이 찬 수 | 3    | 전 기   |
| 300 | 정 기 태 | 2    | 컴 퓨 터 |
| 400 | 송 병 길 | 4    | 컴 퓨 터 |
| 500 | 박 종 화 | 2    | 산 공   |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

(2) 복수의 레코드 변경

- '컴퓨터' 과 과목의 학점(Credit)을 1학점씩 증가시켜라.

실행 내용
SELECT * FROM COURSE;
 UPDATE COURSE SET Credit=Credit+1 WHERE Dept='컴퓨터';
SELECT * FROM COURSE;
실행 결과
<pre>mysql> SELECT * FROM COURSE; +-----+-----+-----+-----+-----+ Cno Cname Credit Dept PRname +-----+-----+-----+-----+-----+ C123 프로그래밍 3 컴퓨터 김성국 C312 자료구조 3 컴퓨터 황수관 C324 화일구조 3 컴퓨터 이규찬 C413 데이터베이스 3 컴퓨터 이일로 E412 반도체 3 전자 홍봉진 +-----+-----+-----+-----+-----+ 5 rows in set (0.00 sec) mysql> UPDATE COURSE -> SET Credit=Credit+1 -> WHERE Dept='컴퓨터'; Query OK, 4 rows affected (0.02 sec) Rows matched: 4 Changed: 4 Warnings: 0 mysql> SELECT * FROM COURSE; +-----+-----+-----+-----+-----+ Cno Cname Credit Dept PRname +-----+-----+-----+-----+-----+ C123 프로그래밍 4 컴퓨터 김성국 C312 자료구조 4 컴퓨터 황수관 C324 화일구조 4 컴퓨터 이규찬 C413 데이터베이스 4 컴퓨터 이일로 E412 반도체 3 전자 홍봉진 +-----+-----+-----+-----+-----+ 5 rows in set (0.00 sec)</pre>

(3) 부속 질의문을 이용한 변경

- '컴퓨터' 과 학생의 기말 성적을 5점씩 가산하라.

실행 내용
SELECT * FROM ENROL;
 UPDATE ENROL SET Final=Final+5 WHERE Sno IN (SELECT Sno FROM STUDENT

WHERE Dept='컴퓨터');
SELECT * FROM ENROL;
실행 결과
<pre> +-----+-----+-----+-----+-----+ Sno Cno Grade Midterm Final +-----+-----+-----+-----+-----+ 100 C413 A 90 95 100 E412 A 95 95 200 C123 B 85 80 300 C312 A 90 95 300 C324 C 75 75 300 C413 A 95 90 400 C312 A 90 95 400 C324 A 95 90 400 C413 B 80 85 400 E412 C 65 75 500 C312 B 85 80 +-----+-----+-----+-----+-----+ 11 rows in set (0.00 sec) mysql> UPDATE ENROL -> SET Final=Final+5 -> WHERE Sno IN -> (SELECT Sno -> FROM STUDENT -> WHERE Dept='컴퓨터 '); Query OK, 9 rows affected (0.02 sec) Rows matched: 9 Changed: 9 Warnings: 0 mysql> SELECT * FROM ENROL; +-----+-----+-----+-----+-----+ Sno Cno Grade Midterm Final +-----+-----+-----+-----+-----+ 100 C413 A 90 100 100 E412 A 95 100 200 C123 B 85 80 300 C312 A 90 100 300 C324 C 75 80 300 C413 A 95 95 400 C312 A 90 100 400 C324 A 95 95 400 C413 B 80 90 400 E412 C 65 80 500 C312 B 85 80 +-----+-----+-----+-----+-----+ 11 rows in set (0.00 sec) </pre>

- 모든 4학년 학생의 학과를 '데이터베이스' 과목을 개설한 학과로 갱신하라.

실행 내용
UPDATE STUDENT
SET Dept=(SELECT Dept FROM COURSE WHERE Cname='데이터베이스')
WHERE Year=4;
실행 결과

```
mysql> UPDATE STUDENT
-> SET Dept=(SELECT Dept FROM COURSE WHERE Cname='데 이 터 베 이 스 ')
-> WHERE Year=4;
Query OK, 2 rows affected (0.00 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> SELECT * FROM STUDENT;
+-----+-----+-----+-----+
| Sno | Sname      | Year | Dept      |
+-----+-----+-----+-----+
| 100 | 나 수 영   | 4    | 컴 퓨 터  |
| 200 | 이 찬 수   | 3    | 전 기     |
| 300 | 정 기 태   | 2    | 컴 퓨 터  |
| 400 | 송 병 길   | 4    | 컴 퓨 터  |
| 500 | 박 종 화   | 2    | 산 공     |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

6.2.3 데이터의 삽입

(1) 레코드의 직접 삽입

- 학번 : 600, 이름 : '박상철', 학년 : 1, 학과 : '컴퓨터'인 학생을 삽입하라.

실행 내용

```
SELECT * FROM STUDENT;
```

```
INSERT INTO STUDENT VALUES(600,'박상철',1,'컴퓨터');
```

```
SELECT * FROM STUDENT;
```

실행 결과

```
mysql> SELECT * FROM STUDENT;
+-----+-----+-----+-----+
| Sno | Sname      | Year | Dept      |
+-----+-----+-----+-----+
| 100 | 나 수 영   | 4    | 컴 퓨 터  |
| 200 | 이 찬 수   | 3    | 전 기     |
| 300 | 정 기 태   | 2    | 컴 퓨 터  |
| 400 | 송 병 길   | 4    | 컴 퓨 터  |
| 500 | 박 종 화   | 2    | 산 공     |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql> INSERT INTO STUDENT VALUES(600,'박 상 철 ',1,'컴 퓨 터 ');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM STUDENT;
+-----+-----+-----+-----+
| Sno | Sname      | Year | Dept      |
+-----+-----+-----+-----+
| 100 | 나 수 영   | 4    | 컴 퓨 터  |
| 200 | 이 찬 수   | 3    | 전 기     |
| 300 | 정 기 태   | 2    | 컴 퓨 터  |
| 400 | 송 병 길   | 4    | 컴 퓨 터  |
| 500 | 박 종 화   | 2    | 산 공     |
| 600 | 박 상 철   | 1    | 컴 퓨 터  |
+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

- 학번 : 600, 이름 : '박상철', 학년 : 1인 학생을 삽입하라.

실행 내용
SELECT * FROM STUDENT; INSERT INTO STUDENT(Sno, Sname, Year) VALUES(600,'박상철',1); SELECT * FROM STUDENT;
실행 결과
<pre>mysql> SELECT * FROM STUDENT; +-----+-----+-----+-----+ Sno Sname Year Dept +-----+-----+-----+-----+ 100 나 수 영 4 컴 퓨 터 200 이 찬 수 3 전 기 300 정 기 태 2 컴 퓨 터 400 송 병 길 4 컴 퓨 터 500 박 종 화 2 산 공 +-----+-----+-----+-----+ 5 rows in set (0.00 sec) mysql> INSERT INTO STUDENT(Sno, Sname, Year) VALUES(600,'박 상 철 ',1); Query OK, 1 row affected (0.01 sec) mysql> SELECT * FROM STUDENT; +-----+-----+-----+-----+ Sno Sname Year Dept +-----+-----+-----+-----+ 100 나 수 영 4 컴 퓨 터 200 이 찬 수 3 전 기 300 정 기 태 2 컴 퓨 터 400 송 병 길 4 컴 퓨 터 500 박 종 화 2 산 공 600 박 상 철 1 NULL +-----+-----+-----+-----+ 6 rows in set (0.00 sec)</pre>

(2) 부속 질의문을 이용한 레코드 삽입

- 학생 테이블에서 '컴퓨터'과 학생의 학번, 이름, 학년을 검색하여 테이블 컴퓨터(COMPUTER)에 삽입하라.

실행 내용
CREATE TABLE COMPUTER (Sno INT, Sname CHAR(6), Year INT); INSERT INTO COMPUTER(Sno, Sname, Year) SELECT Sno, Sname, Year FROM STUDENT WHERE Dept='컴퓨터'; SELECT * FROM COMPUTER;
실행 결과

```
mysql> CREATE TABLE COMPUTER
-> (Sno INT,
-> Sname CHAR(6),
-> Year INT);
Query OK, 0 rows affected (0.08 sec)

mysql>
mysql> SELECT * FROM STUDENT WHERE Dept='컴퓨터';
+-----+-----+-----+-----+
| Sno | Sname | Year | Dept |
+-----+-----+-----+-----+
| 100 | 나수영 | 4 | 컴퓨터 |
| 300 | 정기태 | 2 | 컴퓨터 |
| 400 | 송병길 | 4 | 컴퓨터 |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> INSERT INTO COMPUTER(Sno, Sname, Year)
-> SELECT Sno, Sname, Year FROM STUDENT WHERE Dept='컴퓨터';
Query OK, 3 rows affected (0.03 sec)
Records: 3 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM COMPUTER;
+-----+-----+-----+
| Sno | Sname | Year |
+-----+-----+-----+
| 100 | 나수영 | 4 |
| 300 | 정기태 | 2 |
| 400 | 송병길 | 4 |
+-----+-----+-----+
3 rows in set (0.00 sec)
```

6.2.4 데이터의 삭제

(1) 하나의 레코드 삭제

- 학번 100인 학생을 삭제하라.

실행 내용

```
SELECT * FROM STUDENT WHERE Sno=100;
DELETE FROM STUDENT WHERE Sno=100;
SELECT * FROM STUDENT WHERE Sno=100;
```

실행 결과

```
mysql> SELECT * FROM STUDENT WHERE Sno=100;
+-----+-----+-----+-----+
| Sno | Sname | Year | Dept |
+-----+-----+-----+-----+
| 100 | 나수영 | 4 | 컴퓨터 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> DELETE FROM STUDENT WHERE Sno=100;
Query OK, 1 row affected (0.03 sec)

mysql> SELECT * FROM STUDENT WHERE Sno=100;
Empty set (0.00 sec)
```

(2) 복수의 레코드 삭제

- 등록(ENROL) 테이블의 모든 행을 삭제하라.

실행 내용

```
SELECT * FROM ENROL;
```

DELETE FROM ENROL;
SELECT * FROM ENROL;
실행 결과
<pre>mysql> SELECT * FROM ENROL; +-----+-----+-----+-----+-----+ Sno Cno Grade Midterm Final +-----+-----+-----+-----+-----+ 200 C123 B 85 80 300 C312 A 90 100 300 C324 C 75 80 300 C413 A 95 95 400 C312 A 90 100 400 C324 A 95 95 400 C413 B 80 90 400 E412 C 65 80 500 C312 B 85 80 +-----+-----+-----+-----+-----+ 9 rows in set (0.00 sec) mysql> DELETE FROM ENROL; Query OK, 9 rows affected (0.02 sec) mysql> SELECT * FROM ENROL; Empty set (0.00 sec)</pre>

(3) 부속 질의문을 사용한 삭제

- 과목 'C413'의 기말 성적이 60점 미만인 '컴퓨터'과 학생을 등록 테이블에서 삭제하라.

실행 내용
DELETE FROM ENROL WHERE Cno='C413' AND Final<60 AND ENROL.Sno IN (SELECT Sno FROM STUDENT WHERE Dept='컴퓨터');
실행 결과
<pre>mysql> DELETE FROM ENROL -> WHERE Cno='C413' AND Final<60 -> AND ENROL.Sno IN (SELECT Sno FROM STUDENT WHERE Dept='컴퓨터'); Query OK, 0 rows affected (0.00 sec)</pre>

6.3 SQL 뷰

6.3.1 뷰의 생성

- 뷰 생성

실행 내용
CREATE VIEW CSTUDENT(Sno, Sname, Year) AS SELECT Sno, Sname, Year FROM STUDENT WHERE Dept='컴퓨터' WITH CHECK OPTION;
실행 결과

```
mysql> CREATE VIEW CSTUDENT(Sno, Sname, Year)
-> AS SELECT Sno, Sname, Year
-> FROM STUDENT
-> WHERE Dept='컴 퓨 터 '
-> WITH CHECK OPTION;
Query OK, 0 rows affected (0.04 sec)
```

- 집계함수 사용하여 뷰 생성

실행 내용

```
CREATE VIEW DEPTSIZE(Dept, Size)
AS SELECT Dept, COUNT(*)
FROM STUDENT GROUP BY Dept;
```

```
CREATE VIEW DEPTSIZE
AS SELECT Dept, COUNT(*) AS Size
FROM STUDENT GROUP BY Dept;
```

```
SELECT * FROM DEPTSIZE;
```

실행 결과

```
mysql> CREATE VIEW DEPTSIZE(Dept, Size)
-> AS SELECT Dept, COUNT(*)
-> FROM STUDENT GROUP BY Dept;
Query OK, 0 rows affected (0.02 sec)

mysql> DROP VIEW DEPTSIZE;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE VIEW DEPTSIZE
-> AS SELECT Dept, COUNT(*) AS Size
-> FROM STUDENT GROUP BY Dept;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM DEPTSIZE;
+-----+-----+
| Dept      | Size |
+-----+-----+
| 전 기     | 1    |
| 컴 퓨 터   | 2    |
| 산 공     | 1    |
+-----+-----+
3 rows in set (0.00 sec)
```

- JOIN문 사용하여 뷰 생성

실행 내용

```
CREATE VIEW HONOR(Sname, Dept, Grade)
AS SELECT STUDENT.Sname, STUDENT.Dept, ENROL.Final
FROM STUDENT, ENROL
WHERE STUDENT.Sno=ENROL.Sno AND ENROL.Final>90;
```

```
SELECT * FROM HONOR;
```

실행 결과
<pre>mysql> CREATE VIEW HONOR(Sname, Dept, Grade) -> AS SELECT STUDENT.Sname, STUDENT.Dept, ENROL.Final -> FROM STUDENT, ENROL -> WHERE STUDENT.Sno=ENROL.Sno AND ENROL.Final>90; Query OK, 0 rows affected (0.02 sec) mysql> SELECT * FROM HONOR; +-----+-----+-----+ Sname Dept Grade +-----+-----+-----+ 정 기 태 컴 퓨 터 95 송 병 길 컴 퓨 터 95 +-----+-----+-----+ 2 rows in set (0.02 sec)</pre>

- 정의된 뷰 사용하여 또 다른 뷰 생성

실행 내용
<pre>CREATE VIEW COMHONOR AS SELECT Sname FROM HONOR WHERE Dept='컴퓨터';</pre>
<pre>SELECT * FROM COMHONOR;</pre>
실행 결과
<pre>mysql> CREATE VIEW COMHONOR -> AS SELECT Sname FROM HONOR WHERE Dept='컴 퓨 터 '; Query OK, 0 rows affected (0.01 sec) mysql> SELECT * FROM COMHONOR; +-----+ Sname +-----+ 정 기 태 송 병 길 +-----+ 2 rows in set (0.01 sec)</pre>

6.3.2 뷰의 제거

- RESTRICT / CASCADE

실행 내용
<pre>DROP VIEW DEPTSIZE RESTRICT; SELECT * FROM DEPTSIZE;</pre>
<pre>DROP VIEW HONOR CASCADE; SELECT * FROM HONOR; SELECT * FROM COMHONOR;</pre>
실행 결과

```

mysql> DROP VIEW DEPTSIZE RESTRICT;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM DEPTSIZE;
ERROR 1146 (42S02): Table 'UNIVERSITY.DEPTSIZE' doesn't exist
mysql>
mysql> DROP VIEW HONOR CASCADE;
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT * FROM HONOR;
ERROR 1146 (42S02): Table 'UNIVERSITY.HONOR' doesn't exist
mysql> SELECT * FROM COMHONOR;
ERROR 1356 (HY000): View 'UNIVERSITY.COMHONOR' references invali

```

6.3.3 뷰의 조작 연산

(1) 열 부분집합 뷰

- 삽입, 변경, 삭제 가능한 뷰 (기본키 포함)

실행 내용
<pre> CREATE VIEW STUDENT_VIEW1 AS SELECT Sno, Dept FROM STUDENT; SELECT * FROM STUDENT_VIEW1; DELETE FROM STUDENT_VIEW1 WHERE Sno=500; INSERT INTO STUDENT_VIEW1 VALUES(500,'산공'); UPDATE STUDENT_VIEW1 SET DEPT='전기' WHERE Sno=400; SELECT * FROM STUDENT_VIEW1; </pre>
실행 결과


```

mysql> CREATE VIEW STUDENT_VIEW1
-> AS SELECT Sno, Dept
-> FROM STUDENT;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM STUDENT_VIEW1;
+-----+-----+
| Sno | Dept      |
+-----+-----+
| 200 | 전 기      |
| 300 | 컴 퓨 터    |
| 400 | 컴 퓨 터    |
| 500 | 산 공      |
+-----+-----+
4 rows in set (0.00 sec)

mysql> DELETE FROM STUDENT_VIEW1 WHERE Sno=500;
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO STUDENT_VIEW1 VALUES(500,'산 공 ');
Query OK, 1 row affected (0.02 sec)

mysql> UPDATE STUDENT_VIEW1 SET DEPT='전 기 ' WHERE Sno=400;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM STUDENT_VIEW1;
+-----+-----+
| Sno | Dept      |
+-----+-----+
| 200 | 전 기      |
| 300 | 컴 퓨 터    |
| 400 | 전 기      |
| 500 | 산 공      |
+-----+-----+
4 rows in set (0.00 sec)

```

- 삽입, 변경, 삭제 가능하지 않은 뷰 (기본키 포함 X)

실행 내용
CREATE VIEW STUDENT_VIEW2 AS SELECT Sname, Dept FROM STUDENT;
SELECT * FROM STUDENT_VIEW2; DELETE FROM STUDENT_VIEW2 WHERE Sno=500; INSERT INTO STUDENT_VIEW2 VALUES(500,'산공'); UPDATE STUDENT_VIEW2 SET DEPT='전기' WHERE Sno=400;
실행 결과

```
mysql> CREATE VIEW STUDENT_VIEW2
-> AS SELECT Sname, Dept FROM STUDENT;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM STUDENT_VIEW2;
+-----+-----+
| Sname      | Dept      |
+-----+-----+
| 이 찬 수   | 전 기     |
| 정 기 태   | 컴 퓨 터  |
| 송 병 길   | 전 기     |
| NULL      | 산 공     |
+-----+-----+
4 rows in set (0.00 sec)

mysql> DELETE FROM STUDNET_VIEW2 WHERE Sno=500;
ERROR 1146 (42S02): Table 'UNIVERSITY.STUDNET_VIEW2' doesn't
mysql> INSERT INTO STUDENT_VIEW2 VALUES(500,'산 공 ');
ERROR 1423 (HY000): Field of view 'UNIVERSITY.STUDENT_VIEW2'
mysql> UPDATE STUDENT_VIEW2 SET DEPT='전 기 ' WHERE Sno=400;
ERROR 1054 (42S22): Unknown column 'Sno' in 'where clause'
```

(2) 행 부분집합 뷰

- 기본키 포함된 뷰

실행 내용

```
CREATE VIEW STUDENT_VIEW3
AS SELECT Sno, Sname, Year, Dept
FROM STUDENT WHERE Year=4;

SELECT * FROM STUDENT_VIEW3;
DELETE FROM STUDENT_VIEW3 WHERE DEPT='전기';
INSERT INTO STUDENT_VIEW3 VALUES(400,'송병길',4,'전기');
UPDATE STUDENT_VIEW3 SET DEPT='컴퓨터' WHERE Sno=400;
SELECT * FROM STUDENT_VIEW3;
```

실행 결과

```
mysql> SELECT * FROM STUDENT_VIEW3;
+-----+-----+-----+-----+
| Sno | Sname      | Year | Dept      |
+-----+-----+-----+-----+
| 400 | 송 병 길   | 4    | 전 기     |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> DELETE FROM STUDENT_VIEW3 WHERE DEPT='전 기 ';
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO STUDENT_VIEW3 VALUES(400,'송 병 길 ',4,'전 기 ');
Query OK, 1 row affected (0.02 sec)

mysql> UPDATE STUDENT_VIEW3 SET DEPT='컴 퓨 터 ' WHERE Sno=400;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM STUDENT_VIEW3;
+-----+-----+-----+-----+
| Sno | Sname      | Year | Dept      |
+-----+-----+-----+-----+
| 400 | 송 병 길   | 4    | 컴 퓨 터  |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

(3) 집계함수가 포함된 뷰

- 삽입, 변경, 삭제가 허용되지 않는 뷰

실행 내용
<pre>CREATE VIEW COURSEGRADE(Cno, AVpoint) AS SELECT Cno, AVG(Midterm) FROM ENROL GROUP BY Cno; SELECT * FROM COURSEGRADE; DELETE FROM COURSEGRADE WHERE AVpoint=95.0000;</pre>
실행 결과
<pre>mysql> CREATE VIEW COURSEGRADE(Cno, AVpoint) -> AS SELECT Cno, AVG(Midterm) -> FROM ENROL GROUP BY Cno; Query OK, 0 rows affected (0.03 sec) mysql> SELECT * FROM COURSEGRADE; +-----+-----+ Cno AVpoint +-----+-----+ C123 85.0000 C312 90.0000 C324 75.0000 C413 95.0000 +-----+-----+ 4 rows in set (0.01 sec) mysql> DELETE FROM COURSEGRADE WHERE AVpoint=95.0000; ERROR 1288 (HY000): The target table COURSEGRADE of the DELETE is not updatable</pre>