8장 데이터 검색 명령문 2

1. FROM 절에서 테이블 명세
2. 열 명세
3. 다중 테이블 명세
4. 가명
5. FROM절의 다양한 예제
6. 반드시 가명을 사용해야 하는 경우

8.1 FROM 절에서 테이블 명세

- FROM 절은 사용하려는 테이블을 지정하기 위해서 사용
- 테이블 참조는 가능하다면 가명이 따라오는 테이블 명세로 구성
- 테이블 명세는 테이블 이름으로 구성, 뷰나 동의어를 지정 하여 사용 가능
- 테이블 명세를 두 부분으로 구성 : 테이블 소유자의 이름.테이블의 이름
- 사용자가 테이블의 소유자라면 소유자의 이름은 지정 불필요

[예제 8-1] "SOL"가 생성한 STUDENT 테이블의 전체 내용을 "LEE"이 보고자 할 때 적절한 SELECT 명령문을 완성하라(KIM은 STUDENT 테이블을 질의할 수 있는 권한이 있다고 가정한다).

```
SELECT  * FROM  SOL.STUDENT;
```

"SOL"이란 사용자가 STUDENT 테이블의 내용을 보고자 한다면

```
SELECT  * FROM  STUDENT;
```

8.2 열 명세

- SELECT 명령문은 열의 이름 앞에 열이 포함되어 있는 테이블의 이름을 지정

<table specification> . <column specification>

앞부분은 STUDENT이나 ATTEND과 같은 테이블 이름을 사용하고 생략도 가능
뒷부분은 stu_no이나 stu_name과 같은 열의 이름으로 반듯이 기록

[예제 8-2] 각 학생의 학번을 나타내어라(두 가지 방법을 사용할 수 있다).

```
SELECT   STU_NO
FROM     STUDENT;
```

```
SELECT   STUDENT.STU_NO
FROM     STUDENT;
```

8.3 다중 테이블 명세

-FROM 절에 두 개 이상의 테이블 명세 사용
-서로 다른 테이블로부터 데이터를 가져오려면 FROM 절에 여러 테이블 지정

[예제 8-3] 각 학생의 학번과 이름, 수강년도, 학기, 수강과목코드, 교수코드를 나타내어라.

```
mysql> select student.stu_no, stu_name,
    -> att_year, att_term, sub_code, prof_code
    -> from student, attend
    -> where student.stu_no = attend.stu_no;
```

#FROM 절을 실행한 후 중간 결과

```
+----------+----------+----------+----------++.....+----------+----------+
| stu_no   | stu_name | att_year | att_term ||.....| sub_code | prof_code |
+----------+----------+----------+----------++.....+----------+----------+
| 20141001 | 박도상   | 2014     |        1 ||.....| 4001     | 4002      |
| 20141001 | 박도상   | 2014     |        1 ||.....| 4002     | 4003      |
| 20141001 | 박도상   | 2014     |        1 ||.....| 4003     | 4004      |
| 20141001 | 박도상   | 2014     |        1 ||.....| 4004     | 4001      |
| 20141001 | 박도상   | 2014     |        1 ||.....| 4005     | 4007      |
| 20141001 | 박도상   | 2014     |        1 ||.....| 4006     | 4008
|.....
.....
306 rows in set (0.08 sec)
```

- FROM절의 중간 결과 테이블의 내용 : 카티션 프로덕트(Cartesian product)

  FROM 절의 중간 결과 행의 수 : A 테이블의 행수 * B 테이블의 행 수
  FROM 절의 중간 결과 열의 수 : A 테이블의 열수 + B 테이블의 열 수

예 : STUDENT 테이블의 행수가 17행이고, 열의 수가 15열
ATTEND 테이블의 행수가 18 이고, 열의 수가 11열

- FROM 절의 중간 결과의 행수와 열수

  행 수 : 17 * 18 = 306 행
  열 수 : 15 + 11 = 26 열

- WHERE절의 중간 결과

  student.stu_no = attend.stu_no (“테이블 명세.열의 명세” 형식 사용)
  학생신상 테이블 학번과 수강 테이블의 학번이 같은 경우는 18행만 해당

● SELECT절의 최종 결과 (6개 열 값만 출력)

stu_no	stu_name	att_year	att_term	sub_code	prof_code
20141001	박도상	2014	1	4001	4002
20141001	박도상	2014	1	4002	4003
20141001	박도상	2014	1	4003	4004
20141001	박도상	2014	1	4004	4001
20141001	박도상	2014	1	4005	4007
20141001	박도상	2014	1	4006	4008
20141001	박도상	2014	2	4007	4009
20141001	박도상	2014	2	4008	4005
20141001	박도상	2014	2	4009	4006
20141001	박도상	2014	2	4010	4001
20141001	박도상	2014	2	4011	4002
20141001	박도상	2014	2	4012	4003
20161001	박정인	2016	1	4001	4002
20161001	박정인	2016	1	4002	4003
20161001	박정인	2016	1	4003	4004
20161001	박정인	2016	1	4004	4001
20161001	박정인	2016	1	4005	4007
20161001	박정인	2016	1	4006	4008

18 rows in set (0.00 sec)

[예제8-4] 학생들의 학번, 이름, 수강신청구분을 나타내어라. 단, 수강신청구분은 ATTEND 테이블에 있다.

```
mysql> select s.stu_no, stu_name, att_div
    -> from student s, attend a
    -> where s.stu_no = a.stu_no;
```

8.4 가명(Alias)

- FROM 절에 여러 개의 테이블이 사용되는 경우에 가명 사용하면 편리

[예제8-5] 학생들의 학번, 이름, 등록여부를 나타내어라. 단, 학적 테이블의 가명을 S, 등록테이블의 가명은 F로 정의한다.

```
mysql> select s.stu_no, stu_name, fee_div
    -> from student s, fee f
    -> where s.stu_no = f.stu_no;
```

-가명 S와 A는 FROM절이 먼저 처리되기 때문에 SELECT 절에서 사용되고 있지만 아무런 문제는 없다.
-테이블 이름의 크기를 줄여서 SELECT 절을 좀 더 편리하고 쉽게 사용
-가명은 256문자까지 가능하며 영문자, 숫자, 밑줄 문자를 사용
-첫 번째 문자는 반드시 영문자로 시작
-동일한 가명을 1개의 문장에 사용할 수 없다.

8.5 FROM절의 다양한 예제

[예제 8-6] 수강테이블의 학번과 이름, 과목코드, 교수코드, 교수명을 출력하라.

```
mysql> select s.stu_no, stu_name, sub_code, p.prof_code, prof_name
    -> from student s, attend a, professor p
    -> where s.stu_no = a.stu_no and
    -> a.prof_code = p.prof_code;
```

[예제 8-7] 학적테이블의 학번과 이름, 수강테이블의 수강년도, 학기, 수강교과목코드, 교과목테이블의 교과목명을 나타내어라.

```
mysql> select s.stu_no, stu_name, att_year, att_term, a.sub_code, sub_name
    -> from student s, attend a, subject su
    -> where s.stu_no = a.stu_no and
    -> a.sub_code = su.sub_code;
```

stu_no	stu_name	att_year	att_term	sub_code	sub_name
20141001	박도상	2014	1	4001	데이터베이스 응용
20141001	박도상	2014	1	4002	웹사이트 구축
20141001	박도상	2014	1	4003	소프트웨어공학
20141001	박도상	2014	1	4004	웹프로그래밍
20161001	박정인	2016	1	4002	웹사이트 구축
20161001	박정인	2016	1	4003	소프트웨어공학
20161001	박정인	2016	1	4004	웹프로그래밍
20161001	박정인	2016	1	4005	컴퓨터구조
20161001	박정인	2016	1	4006	정보처리실무

18 rows in set (0.20 sec)

[예제 8-8] 학적테이블의 학번과 이름, 보관성적테이블의 성적 취득년도, 학기, 신청학점, 취득학점, 평점평균을 나타내어라.

```
mysql> select s.stu_no, stu_name, sco_year, sco_term,
    -> req_point, take_point, exam_avg
    -> from student s, score sc
    -> where s.stu_no = sc.stu_no;
+----------+-------------+----------+----------+-----------+------------+----------+
| stu_no   | stu_name    | sco_year | sco_term | req_point | take_point | exam_avg |
+----------+-------------+----------+----------+-----------+------------+----------+
| 20141001 | 박도상      | 2014     |        1 |        18 |         18 |      4.5 |
| 20141001 | 박도상      | 2014     |        2 |        18 |         18 |      4.0 |
| 20191001 | 김유신      | 2019     |        1 |        18 |         18 |      4.2 |
| 20191001 | 김유신      | 2019     |        2 |        18 |         18 |      0.0 |
| 20191002 | 홍길동      | 2019     |        1 |        18 |         18 |      4.5 |
| 20191002 | 홍길동      | 2019     |        2 |        18 |         18 |      0.0 |
| 20191005 | 김할리      | 2019     |        1 |        18 |         18 |      4.4 |
| 20191006 | 최에스터    | 2019     |        1 |        18 |         18 |      4.4 |
| 20191007 | 신안나      | 2019     |        2 |        18 |         18 |      0.0 |
+----------+-------------+----------+----------+-----------+------------+----------+
9 rows in set (0.01 sec)
```

[예제 8-9] 학적테이블의 학번과 이름, 수강테이블의 수강신청년도, 학기, 수강신청유무를 나타내어라.

```
mysql> select s.stu_no, stu_name, att_year, att_term, att_div
    -> from student s, attend a
    -> where s.stu_no = a.stu_no;
```

중복된 데이터가 존재하므로 SELECT 절에 DISTINCT를 사용 중복 제게

[예제 8-10] 학적테이블의 학번과 이름, 수강테이블의 수강신청년도, 학기, 수강신청유무를 나타내어라.(단, 중복은 배제한다.)

```
mysql> select distinct s.stu_no, stu_name, att_year, att_term, att_div
    -> from student s, attend a
    -> where s.stu_no = a.stu_no;

+----------+----------+----------+----------+----------+
| stu_no   | stu_name | att_year | att_term | att_div |
+----------+----------+----------+----------+----------+
| 20141001 | 박도상   | 2014     |        1 | Y       |
| 20141001 | 박도상   | 2014     |        2 | Y       |
| 20161001 | 박정인   | 2016     |        1 | Y       |
+----------+----------+----------+----------+----------+
3 rows in set (0.02 sec)
```

최종 결과를 보면 중복된 데이터가 제거되어 18개 행에서 3개 행으로 줄어든 것을 확인

[예제8-11]적어도 한번 이상 장학금을 받은 학생의 이름을 나타내어라.

```
mysql> select distinct s.stu_no, stu_name
    -> from student s, fee f
    -> where s.stu_no = f.stu_no and
    -> not jang_code is null;

+----------+--------------+
| stu_no   | stu_name     |
+----------+--------------+
| 20141001 | 박도상       |
| 20161001 | 박정인       |
| 20191004 | 이순신       |
| 20191005 | 김할리       |
| 20191006 | 최에스터     |
| 20191007 | 신안나       |
| 20191008 | 연개소문     |
| 20201002 | 강감찬       |
+----------+--------------+
8 rows in set (0.00 sec)
```

조건 절에서 NOT JANG_CODE IS NULL의 “NOT IS NULL”은 “NULL값을 같지 않는다” 라고 해석한다.

[예제8-12] 등록한 학생의 이름, 등록년도, 학기, 장학코드, 장학금총액, 등록구분을 나타내어라.

```
mysql> select stu_name, fee_year, fee_term, jang_code, jang_total, fee_div
    -> from student s, fee f
    -> where s.stu_no = f.stu_no
    -> and f.fee_div = 'Y';
```

stu_name	fee_year	fee_term	jang_code	jang_total	fee_div
박도상	2014	1	1	500000	Y
박도상	2014	2	10	2500000	Y
박도상	2015	1	11	2000000	Y
박도상	2015	2	21	800000	Y
박도상	2019	1	10	2500000	Y
박도상	2019	2	10	2500000	Y
박정인	2016	1	10	2500000	Y
박정인	2016	2	10	2500000	Y
박정인	2019	2	10	2500000	Y
이순신	2019	1	1	500000	Y
이순신	2019	2	11	2000000	Y
김할리	2019	1	1	500000	Y
김할리	2019	2	NULL	NULL	Y
최에스터	2019	1	1	500000	Y
최에스터	2019	2	NULL	NULL	Y
신안나	2019	1	1	500000	Y
신안나	2019	2	NULL	NULL	Y
연개소문	2019	1	1	500000	Y
연개소문	2019	2	NULL	NULL	Y
강감찬	2020	1	1	500000	Y
강감찬	2020	2	10	2500000	Y

24 rows in set (0.00 sec)

- FROM 절에서 테이블 명세가 사용되는 순서는 무관

①
```
SELECT S.STU_NO
FROM  STUDENT S, FEE F
WHERE S.STU_NO = F.STU_NO
```

②
```
SELECT S.STU_NO
FROM  FEE F, STUDENT S
WHERE F.STU_NO = S.STU_NO
```

①번과 ②번은 FROM 절에서 테이블의 순서가 서로 바뀌었으나 결과 값은 동일

- SELECT 절은 열이 출력되는 순서를 정의
- ORDER BY 절은 행의 순서를 정의

[예제 8-13] 2007년에 등록한 학생의 학번과 이름을 나타내어라.

```
mysql> select distinct s.stu_no, s.stu_name
    -> from student s, fee f
    -> where s.stu_no = f.stu_no
    -> and fee_year = 2019;

+----------+-------------+
| stu_no   | stu_name    |
+----------+-------------+
| 20141001 | 박도상      |
| 20161001 | 박정인      |
| 20191004 | 이순신      |
| 20191005 | 김할리      |
| 20191006 | 최에스터    |
| 20191007 | 신안나      |
| 20191008 | 연개소문    |
+----------+-------------+
8 rows in set (0.02 sec)
```

8.6 반드시 가명을 사용해야 하는 경우

-FROM 절에서 동일한 테이블 이름이 한 번 이상 사용될 때 반드시 가명 사용

[예제 8-14] 장수인(1999년 02월 09일) 학생보다 먼저 태어난 학생의 이름과 생년월일을 나타내어라.

```
mysql> select s.stu_name, s.birthday
    -> from student s, student st
    -> where st.stu_name = '장수인'
    -> and  s.birthday < st.birthday;
+----------+----------+
| stu_name  | birthday |
+----------+----------+
| 박도상    | 19960116 |
| 박정인    | 19970403 |
| 김유미    | 19990207 |
+----------+----------+
3 rows in set (0.00 sec)
```

장수인의 생년월일(1999년 02월 09일)을 미리 알고 있다면 가명을 사용하지 않아도 됨

SELECT 절에서 S.BIRTHDAY 대신에 ST.BIRTHDAY를 사용하면 원하지 않은 결과를 출력한다. 그러므로 반드시 가명을 옳바르게 기입해 주어야 한다.

```
mysql> select s.stu_name, st.birthday
    -> from student s, student st
    -> where st.stu_name = '장수인'
    -> and  s.birthday < st.birthday;
+----------+----------+
| stu_name | birthday |
+----------+----------+
| 박도상   | 19990209 |
| 박정인   | 19990209 |
| 김유미   | 19990209 |
+----------+----------+
3 rows in set (0.00 sec)
```