

# 데이터베이스시스템

## 12. SQL(3/3) - 데이터 조작어

나 홍 석    교수



12  
LESSON

# SQL(3/3) – 데이터 조작어

# 학습 목표

- 1** SQL의 데이터 조작어의 종류를 나열하고 기능을 설명할 수 있다.
- 2** SQL 명령어를 사용하여 데이터를 삽입, 삭제, 수정할 수 있다.
- 3** SELECT 명령어를 이용하여 데이터를 검색할 수 있다.

# 학습 내용

**1** 데이터 삽입하기

---

**2** 데이터 삭제/수정하기

---

**3** 데이터 검색하기1

---

**4** 데이터 검색하기2

---

# Chapter 01 데이터 삽입하기

# 1 데이터 삽입 명령어

## 1 INSERT INTO 명령어

- ✓ INSERT INTO 명령어를 사용하여 데이터를 삽입
- ✓ 데이터의 삽입은 행(로우, 튜플) 단위로 이루어 짐
- ✓ DEFAULT 제약조건이 있는 경우,  
삽입되는 값이 없을 때 DEFAULT 값이 들어감

# 1 데이터 삽입 명령어

## 2 형식 및 예제

형식 **INSERT INTO** <테이블명> [(<컬럼 리스트>)] **VALUES** (<값 리스트>);

---

### 설명

- 테이블(=릴레이션)에 하나의 행(=튜플, 로우) 단위로 저장
  - SELECT 문을 이용해서 정상적으로 삽입이 되었는지 확인
  - 텍스트 형식의 컬럼은 작은 따옴표로 묶어서 표현
  - 숫자는 따옴표로 묶지 않고 그대로 사용
- 

### 형식

**INSERT INTO** student ( st\_id, st\_name, st\_sex, st\_phn\_mbl, st\_addr, st\_birth, dept\_id)  
**VALUES** ('ST001', '최현주','F', '010-1234-1234','서울','1973/01/21','SE');

# 1 데이터 삽입 명령어

## 3 명령어 실행화면

The screenshot displays a SQL IDE interface with the following components:

- Navigator:** Shows a tree view of the database schema. The 'academy' database is expanded, showing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The 'student' table is selected under 'Tables'.
- Query Editor:** Contains the following SQL code:

```
39 • USE academy;  
40  
41 • INSERT INTO student ( st_id, st_name, st_sex, st_phn_mbl, st_addr, st_birth, dept_id)  
42   VALUES ('ST001', '최현주', 'F', '010-1234-1234', '서울', '1973/01/21', 'SE');  
43  
44  
45  
46
```
- Output:** Shows the execution results in a table format:

#	Time	Action	Message	Duration / Fetch
1	02:54:19	INSERT INTO student ( st_id, st_name, st_sex, st_phn_mbl, st_addr, st_birth, ...	1 row(s) affected	0.016 sec

The 'Output' tab is active, showing the 'Action Output' section. The message '1 row(s) affected' indicates the successful execution of the INSERT statement.



# 1 데이터 삽입 명령어

## 4 데이터 삽입 확인

- ✓ **SELECT** 명령어를 사용해서 테이블에 입력된 데이터 확인
- ✓ 형식 → **SELECT \* FROM 테이블명**

The screenshot shows the SQL Developer interface. On the left, the 'sys' schema is selected in the 'Schemas' pane. The main window displays the following SQL commands:

```
41 • INSERT INTO student ( st_id, st_name, st_sex, st_phn_mbl, st_addr, st_birth, dept_id)
42     VALUES ('ST001', '최현주', 'F', '010-1234-1234', '서울', '1973/01/21', 'SE');
43
44 • SELECT * FROM student;
45
```

Below the SQL editor, the 'Result Grid' is displayed, showing the data inserted into the 'student' table:

	st_name	st_id	st_sex	st_phn_mbl	st_addr	st_birth	dept_id
	최현주	ST001	F	010-1234-1234	서울	1973-01-21 00:00:00	SE
▶*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The bottom of the window shows the 'Administration' and 'Schemas' tabs, with 'Schemas' selected. The 'Information' pane at the bottom left shows 'student 1'.

## 2 초기데이터 입력

### 1 student 테이블 #1

☑ **INSERT INTO 명령어**를 이용해 각 행을 하나씩 입력

☑ (참고) 전체 행삭제 → DELETE FROM student;

**student**

st_id	st_name	st_sex	st_phn_mbl	st_addr	st_birth	dept_id
ST001	최현주	F	010-1234-1234	서울	1973/01/21	SE
ST002	강하늘	M	010-2222-2344	서울	1990/11/30	BZ
ST003	이성민	M	010-3293-9345	서울	1978/06/30	SE
ST004	박정수	F	010-8323-8342	경기	2000/07/12	EE
ST005	홍민호	M	010-2342-6547	대전	1985/03/03	BZ

## 2 초기데이터 입력

# 1 student 테이블 #2

```
41 • INSERT INTO student ( st_id, st_name, st_sex, st_phn_mbl, st_addr, st_birth, dept_id)
42     VALUES ('ST001', '최현주', 'F', '010-1234-1234', '서울', '1973/01/21', 'SE');
43 • INSERT INTO student ( st_id, st_name, st_sex, st_phn_mbl, st_addr, st_birth, dept_id)
44     VALUES ('ST002', '강하늘', 'M', '010-2222-2344', '서울', '1990/11/30', 'BZ');
45 • INSERT INTO student ( st_id, st_name, st_sex, st_phn_mbl, st_addr, st_birth, dept_id)
46     VALUES ('ST003', '이성민', 'M', '010-3293-9345', '서울', '1978/06/30', 'SE');
47 • INSERT INTO
```

[illegible]

## 2 초기데이터 입력

### 2 course 테이블 #1

- ✓ **INSERT INTO 명령어**를 이용해 각 행을 하나씩 입력
- ✓ (참고) 전체 행삭제 → DELETE FROM course;

**course**

co_num	co_name	co_location
IT111	컴퓨터학개론	인촌관
IT222	데이터베이스시스템	화정관
BZ001	경영학원론	고려대학교
MD123	웹디자인기초	화정관

## 2 초기데이터 입력

### 2 course 테이블 #2

```
54
55 • INSERT INTO course (co_num, co_name, co_location) VALUES ('IT111', '컴퓨터학개론', '인촌관' );
56 • INSERT INTO course (co_num, co_name, co_location) VALUES ('IT222', '데이터베이스시스템', '화정관');
57 • INSERT INTO course (co_num, co_name, co_location) VALUES ('BZ001', '경영학원론', '고려대');
58 • INSERT INTO course (co_num, co_name, co_location) VALUES ('MD123', '웹디자인기초', '화정관');
59
60 • SELECT * FROM course;
61
```

Result Grid			
Filter Rows: <input type="text"/>			
Edit:			
Export/Import:			
Wrap Cell Content:			
	co_num	co_name	co_location
▶	BZ001	경영학원론	고려대
	IT111	컴퓨터학개론	인촌관
	IT222	데이터베이스시스템	화정관
	MD123	웹디자인기초	화정관
*	NULL	NULL	NULL

## 2 초기데이터 입력

### 3 enrol 테이블 #1

enrol	st_id	co_num	grade	mid	final
	ST001	IT111	A	90	95
	ST001	IT222	B	80	87
	ST001	BZ001	A	92	100
	ST002	IT111	C	75	75
	ST002	BZ001	A	96	95
	ST003	IT222	A	90	100
	ST003	BZ001	B	84	80
	ST004	IT111	B	80	84
	ST004	IT222	C	80	78
	ST005	MD123	C	74	78

- ☑ 외래키인 컬럼에 참조하지 않는 값이 들어오면 에러 발생
- ☑ 기본키 값이 중복되는 경우 에러 발생

## 2 초기데이터 입력

### 3 enrol 테이블 #2

Result Grid | Filter Rows: | Edit:

	st_id	co_num	grade	mid	final
▶	ST001	BZ001	A	92	100
	ST001	IT111	A	90	95
	ST001	IT222	B	80	87
	ST002	BZ001	A	96	95
	ST002	IT111	C	75	75
	ST003	BZ001	B	84	80
	ST003	IT222	A	90	100
	ST004	IT111	B	80	84
	ST004	IT222	C	80	78
	ST005	MD123	C	74	78
*	NULL	NULL	NULL	NULL	NULL

enrol 2 x

```
um, grade, mid, final) VALUES ('ST001', 'IT111', 'A', 90, 95);  
um, grade, mid, final) VALUES ('ST001', 'IT222', 'B', 80, 87);  
um, grade, mid, final) VALUES ('ST001', 'BZ001', 'A', 92, 100);  
um, grade, mid, final) VALUES ('ST002', 'IT111', 'C', 75, 75);  
um, grade, mid, final) VALUES ('ST002', 'BZ001', 'A', 96, 95);  
um, grade, mid, final) VALUES ('ST003', 'IT222', 'A', 90, 100);  
um, grade, mid, final) VALUES ('ST003', 'BZ001', 'B', 84, 80);  
um, grade, mid, final) VALUES ('ST004', 'IT111', 'B', 80, 84);  
um, grade, mid, final) VALUES ('ST004', 'IT222', 'C', 80, 78);  
um, grade, mid, final) VALUES ('ST005', 'MD123', 'C', 74, 78);
```

```
71 • INSERT INTO enrol (st_id, co_num, grade, mid, final) VALUES ('ST004', 'IT222', 'C', 80, 78);  
72 • INSERT INTO enrol (st_id, co_num, grade, mid, final) VALUES ('ST005', 'MD123', 'C', 74, 78);  
73  
74 • SELECT * FROM enrol;
```



실습 영  
상

# 데이터 입력



## Chapter 02 데이터 삭제/수정하기

# 1 데이터 삭제 명령어

## 1 DELETE 명령어

- ☑ DELETE 명령어를 사용하여 데이터를 삽입함
- ☑ 데이터의 삭제는 행(로우, 튜플) 단위로 이루어짐
- ☑ 삭제조건을 주어서 원하는 행만 삭제할 수 있음

# 1 데이터 삭제 명령어

## 2 형식 및 예제

형식 **DELETE FROM** <테이블명> [ **WHERE** <조건> ]

---

설명

- 테이블(=릴레이션)에서 조건에 맞는 행(=튜플, 로우)들을 삭제
  - WHERE 절이 생략되면, 테이블의 모든 행들을 삭제 (참고) TRUNCATE TABLE <테이블 명>
  - 특정 컬럼의 데이터 삭제는 불가(UPDATE 문 이용)
- 

예

**DELETE FROM** enrol **WHERE** st\_id = 'ST001' ;

# 1 데이터 삭제 명령어

## 3 실행화면 #1

```
75
76 • DELETE FROM enrol WHERE st_id = 'ST001';
77
78
```

Output .....



Action Output



	#	Time	Action	Message
✓	1	07:47:41	USE academy	0 row(s) affected
✓	2	08:09:33	DELETE FROM enrol WHERE st_id = 'ST001'	3 row(s) affected

해당되는 3개 행이 삭제됨



# 1 데이터 삭제 명령어

## 3 실행화면 #2

74 • **SELECT \* FROM enrol;**

75

76 • **DELETE FROM enrol WHERE st\_id = 'ST001';**

< **Result Grid** | Filter Rows:  | Edit: | Export/Import: | Wrap Cell Content:

	st_id	co_num	grade	mid	final
▶	ST002	BZ001	A	96	95
	ST002	IT111	C	75	75
	ST003	BZ001	B	84	80
	ST003	IT222	A	90	100
	ST004	IT111	B	80	84
	ST004	IT222	C	80	78
	ST005	MD123	C	74	78
*	NULL	NULL	NULL	NULL	NULL

enrol 1 x

# 1 데이터 삭제 명령어

## 4 데이터 삭제 연습


질의문

student 테이블에서 성별이 남자인 행을 삭제하세요!

SQL

**DELETE FROM** student **WHERE** st\_sex = 'M' ;

외래키 때문에 삭제되지  
않음!



질의문

enrol 테이블에서 중간시험 점수가 90점 이상인 행을 삭제하세요!

SQL

**DELETE FROM** enrol **WHERE** mid >= 90 ;

## 2 데이터 수정 명령어

### 1 UPDATE 명령어

- ☑ **UPDATE** 명령어를 사용하여 데이터를 삽입함
- ☑ 데이터의 수정은 **컬럼(열, 애트리뷰트)** 단위로 이루어짐
- ☑ 조건에 맞는 행(로우)들의 지정한 컬럼을 주어진 값으로 바꿈

## 2 데이터 수정 명령어

### 2 형식 및 예제

형식

**UPDATE** <테이블명> **SET** <컬럼1>=<값>, <컬럼2>=<값>, ... [**WHERE** <조건>]

---

설명

- 테이블의 데이터를 변경할 때 사용하는 명령문
  - **WHERE** 절이 생략되면, 컬럼의 모든 값이 업데이트 됨
  - <값>에는 수식이 들어갈 수도 있음
- 

예

**UPDATE** student **SET** st\_addr = '부산' **WHERE** st\_id = 'ST001' ;





## 2 데이터 수정 명령어

### 3 데이터 수정 연습 #2

질의문 모든 학생의 중간고사 점수를 1점 올리세요.

SQL **UPDATE** student **SET** mid = mid + 1 ;

	st_id	co_num	grade	mid	final
▶	ST001	BZ001	A	92	100
	ST001	IT111	A	90	95
	ST001	IT222	B	80	87
	ST002	BZ001	A	96	95
	ST002	IT111	C	75	75
	ST003	BZ001	B	84	80
	ST003	IT222	A	90	100
	ST004	IT111	B	80	84
	ST004	IT222	C	80	78
	ST005	MD123	C	74	78
*	NULL	NULL	NULL	NULL	NULL



	st_id	co_num	grade	mid	final
▶	ST001	BZ001	A	93	100
	ST001	IT111	A	91	95
	ST001	IT222	B	81	87
	ST002	BZ001	A	97	95
	ST002	IT111	C	76	75
	ST003	BZ001	B	85	80
	ST003	IT222	A	91	100
	ST004	IT111	B	81	84
	ST004	IT222	C	81	78
	ST005	MD123	C	75	78
*	NULL	NULL	NULL	NULL	NULL



실습 영상

# 데이터 삭제/수정

## Chapter 03 데이터 검색하기 - 기본 검색

# 1 데이터 검색 명령어

## 1 SELECT 명령어

- ☑ 관계형데이터베이스는 데이터를 가져오기 위해  
SELECT 명령어를 사용함
- ☑ 데이터의 검색은 행(로우, 튜플, 레코드) 단위로 이루어짐
- ☑ 원하는 행 또는 원하는 열의 데이터만 가져올 수 있음

# 1 데이터 검색 명령어

## 2 형식 및 예제

### 형식

```
SELECT [DISTINCT] <컬럼리스트>  
FROM <테이블 리스트>  
[WHERE <조건>]  
[GROUP BY <컬럼명> [HAVING <그룹 조건>]]  
[ORDER BY <컬럼명> [ASC | DESC]]
```

### 설명

- 컬럼리스트에는 출력을 원하는 컬럼명을 사용, 모든 컬럼 → \* 사용
- FROM 절에는 데이터를 가져올 테이블을 모두 명시
- WHERE 절에는 조건식을 명시

### 예

```
SELECT * FROM student;  
SELECT * FROM student WHERE st_sex = 'M' and dept_id = 'SE' ;
```



## 2 컬럼 검색

### 2 테이블의 컬럼 일부를 검색하는 경우

질의문 학생의 학번, 이름을 검색하세요.

SQL **SELECT** st\_id '학번', st\_name '이름' **FROM** student;

결과

Result Grid     Filter Rows		
	학번	이름
▶	ST001	최현주
	ST002	강하늘
	ST003	이성민
	ST004	박정수
	ST005	홍민호



## 2 컬럼 검색

3 검색 결과에 중복된 로우(레코드)를 제거할 경우

질의문 학생이 소속된 학과만을 출력하세요.

SQL **SELECT** dept\_id **FROM** student;

결과

	dept_id
▶	SE
	BZ
	SE
	EE
	BZ

**SELECT DISTINCT** dept\_id **FROM** student;

	dept_id
▶	SE
	BZ
	EE

# 2

# 컬럼 검색

4

## 질의문

# SQL

```
SELECT * FROM student ORDER BY st_name ASC;
```

```
SELECT * FROM student ORDER BY st_name DESC;
```

## 결과

[illegible]

## 2 컬럼 검색


### 4 검색 결과를 정렬해서 보여주기 #2

질의문 등록테이블의 정보를 성적순(오름차순)으로 정렬하세요.

SQL **SELECT \* FROM enrol ORDER BY grade ASC;**

결과

Result Grid



Filter Rows:

Edit:

	st_id	co_num	grade	mid	final
▶	ST001	BZ001	A	93	100
	ST001	IT111	A	91	95
	ST002	BZ001	A	97	95
	ST003	IT222	A	91	100
	ST001	IT222	B	81	87
	ST003	BZ001	B	85	80
	ST004	IT111	B	81	84
	ST002	IT111	C	76	75
	ST004	IT222	C	81	78
	ST005	MD123	C	75	78
✱	NULL	NULL	NULL	NULL	NULL



## 2 컬럼 검색

### 5 계산해서 보여주기

질의문 등록테이블에서 시험점수 점수합계를 포함해서 보여주세요.

SQL `SELECT *, mid+final FROM enrol WHERE co_num = 'IT111' ;`  
`SELECT *, mid+final '총점' FROM enrol WHERE co_num = 'IT111' ;`

결과

Result Grid    Filter Rows: <input type="text"/>   Export:  Wrap Cell C						
	st_id	co_num	grade	mid	final	mid+final
▶	ST001	IT111	A	91	95	186
	ST002	IT111	C	76	75	151
	ST004	IT111	B	81	84	165



실습 영상

# 데이터 검색하기 - 기본검색

## Chapter 04 데이터 검색하기 - 조건 검색

# 1 조건검색

## 1 WHERE 절

**SELECT** <컬럼리스트> **FROM** <테이블리스트> **WHERE** <행검색조건>

- ☑ 원하는 조건을 만족하는 행만을 검색하고자 할 경우 **WHERE절에 조건을 명시**
- ☑ WHERE절에는 **조건식**이 들어감
- ☑ 조건식은 **비교연산자와 논리연산자**를 사용하여 구성

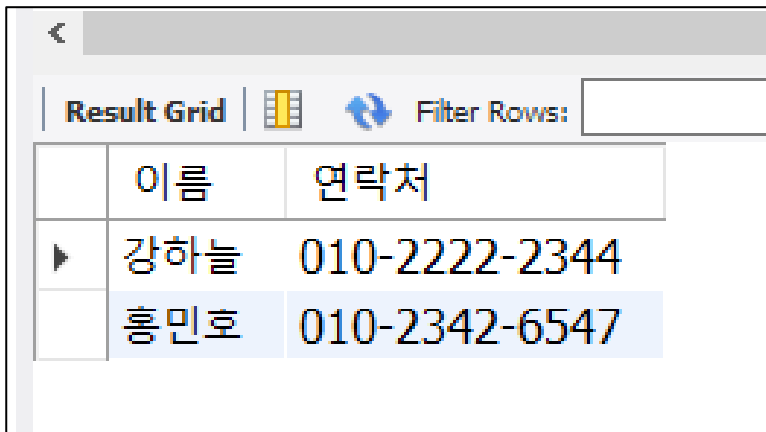
# 1 조건검색

## 2 조건검색 예제

질의문 경영학과(BZ)에 재학중인 학생의 이름과 연락처를 검색하세요.

SQL **SELECT** st\_name '이름', st\_phn\_mbl '연락처'  
**FROM** student  
**WHERE** dept\_id = 'BZ' ;

결과



The screenshot shows a database query result grid. At the top, there is a header bar with a back arrow, the text 'Result Grid', a grid icon, a refresh icon, and a 'Filter Rows:' input field. Below this is a table with two columns: '이름' (Name) and '연락처' (Contact). The first row is highlighted with a blue background and contains the values '강하늘' and '010-2222-2344'. The second row is highlighted with a light blue background and contains the values '홍민호' and '010-2342-6547'.

	이름	연락처
▶	강하늘	010-2222-2344
	홍민호	010-2342-6547



# 1 조건검색

## 3 조건식의 연결 #1

<조건식> 논리연산자 <조건식>

☒ 논리연산자를 이용해서 두 개 이상의 컬럼에 대해 조건을 적용

- NOT 논리적 부정
- AND 논리곱, 연산 대상이 모두 참이어야 참
- OR 논리합, 연산 대상 중 하나라도 참이면 참

# 1 조건검색

## 3 조건식의 연결 #2

질의문

소프트웨어공학 과목에서 중간고사 성적이 90점 이상인 학생을 검색하라.

SQL

```
SELECT * FROM enrol  
WHERE co_num = 'IT111' AND mid >=90 ;
```

결과

Result Grid					
Filter Rows: <input type="text"/> Edit:					
	st_id	co_num	grade	mid	final
▶	ST001	IT111	A	91	95
*	NULL	NULL	NULL	NULL	NULL

## 2 집계함수

### 1 집계함수 개요

☑ 통계나 집계 목적으로 사용되는 함수

#### 자주 사용하는 집계함수

- **SUM()** 합계를 구한다.
- **AVG()** 평균을 구한다.
- **MIN(), MAX()** 최소값, 최대값을 구한다.
- **COUNT()** 행의 개수를 구한다.

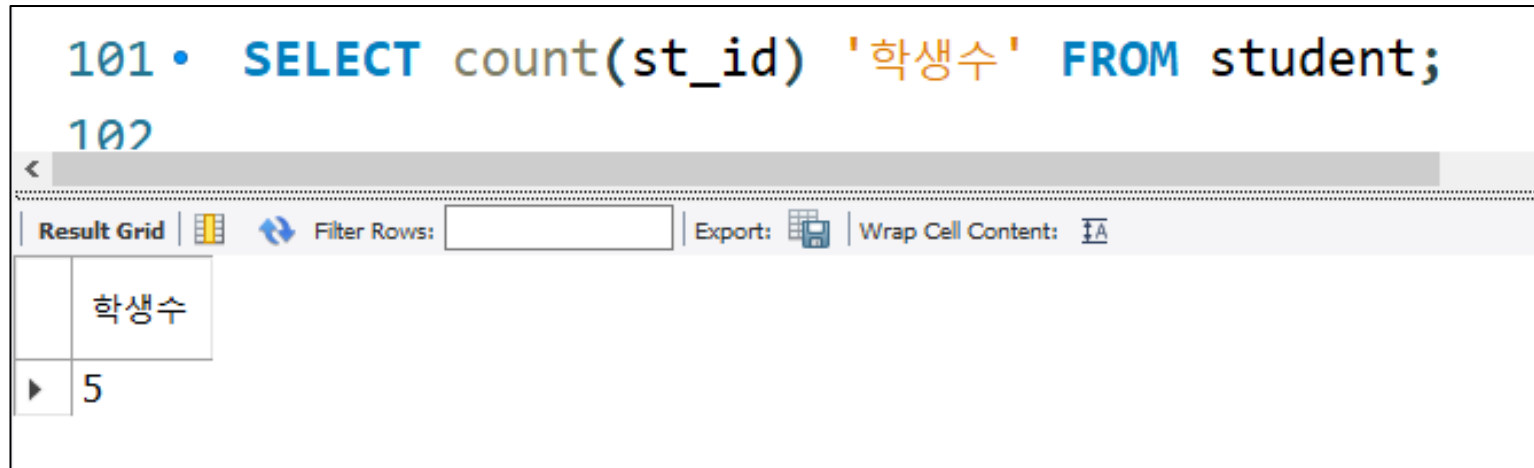
## 2 집계함수

### 2 집계함수 사용 예제 #1

질의문 학생수가 모두 몇 명인지 구하세요.

SQL **SELECT** count(st\_id) '학생수' **FROM** student ;

결과



The screenshot shows a SQL query execution interface. At the top, the query is displayed: `101 • SELECT count(st_id) '학생수' FROM student;`. Below the query, the result is shown in a table format. The table has one column labeled '학생수' and one row with the value '5'. The interface includes a 'Result Grid' tab, a 'Filter Rows' input field, and an 'Export' button.

학생수
5

## 2 집계함수

### 2 집계함수 사용 예제 #2

질의문 중간고사, 기말고사 점수의 평균은 얼마인가?

SQL **SELECT** avg(mid) '중간평균', avg(final) '기말평균' **FROM** enrol ;

결과

```
103 • SELECT avg(mid) '중간평균', avg(final) '기말평균' FROM enrol ;
104
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: IA

	중간평균	기말평균
▶	85.1000	87.2000

# 3 그룹별 집계

## 1 GROUP BY 절

- ☑ 검색 결과를 일정한 기준에 의해 **그룹으로 묶어주는 역할**을 함
- ☑ 집계함수와 함께 사용되는 경우가 많음
- ☑ 여러 **그룹별 집계자료**를 만드는 경우 유용하게 사용됨

# 3 그룹별 집계

## 2 사용 예제

질의문 과목별 기말시험 점수의 합계를 구하세요.

SQL **SELECT** sum(final) '과목별 총점' **FROM** enrol **GROUP BY** co\_num ;

결과

Result Grid		Filter Rows
	과목별 총점	
▶	275	
	254	
	265	
	78	

**SELECT** co\_num, sum(final) **FROM** enrol  
**GROUP BY** co\_num ;

Result Grid			Filter Rows:
	co_num	sum(final)	
▶	BZ001	275	
	IT111	254	
	IT222	265	
	MD123	78	

# 3 그룹별 집계



## 3 HAVING 절

☑ 그룹별 집계자료에서 조건에 맞는 결과만 보여주도록 함

질의문 과목별 기말시험 점수의 합계를 구하세요.  
단, 합계가 200이상인 과목만 보여줍니다.

SQL **SELECT** co\_num '과목번호', sum(final) '총점' **FROM** enrol  
**GROUP BY** co\_num  
**HAVING** sum(final) >=200 ;

결과

Result Grid     Filter Rows		
	과목 번호	총점
▶	BZ001	275
	IT111	254
	IT222	265





실습 영상

# 데이터 검색하기 - 조건 검색

# 학습 정리



## SQL을 이용한 데이터 삽입/삭제/수정

**INSERT INTO** <테이블명> [(<컬럼 리스트>)]  
**VALUES** (<값 리스트>)

**DELETE FROM** <테이블명> [ **WHERE** <조건> ]

**UPDATE** <테이블명>  
**SET** <컬럼1>=<값>, <컬럼2>=<값>, ...  
[**WHERE** <조건>]

# 학습 정리

## SQL을 이용한 데이터 검색

```
SELECT [DISTINCT] <컬럼리스트>  
FROM <테이블 리스트>  
[WHERE <조건>]  
[GROUP BY <컬럼명> [HAVING <그룹 조건>]]  
[ORDER BY <컬럼명> [ASC | DESC]]
```

# 참고 문헌



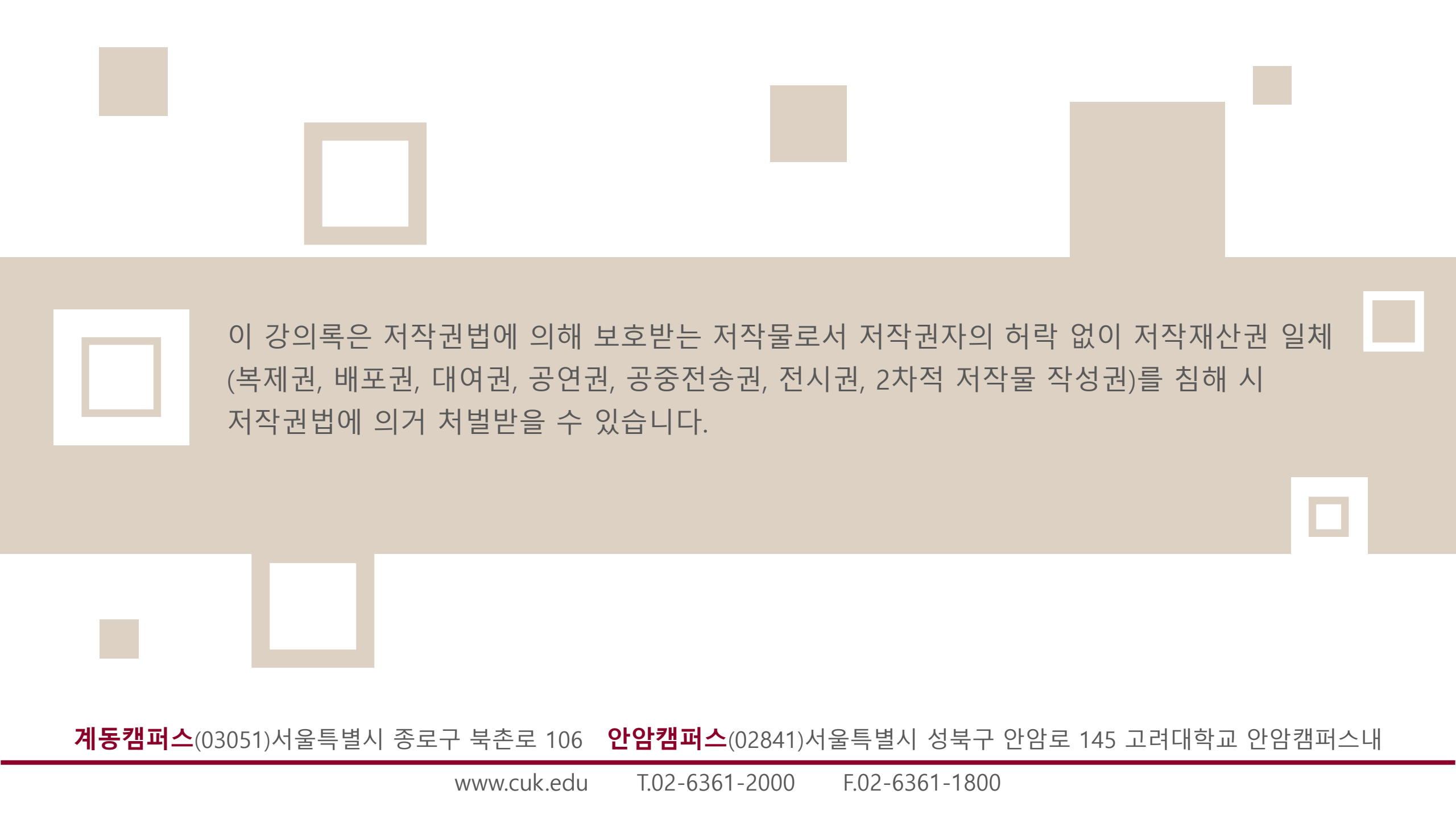
데이터베이스 시스템 7판,  
Ramez Elmasri , Shamkant B. Navathe  
지음, 황규영 등 옮김, 홍릉과학출판사,  
2018년 8월

---



[www.wikipedia.org](http://www.wikipedia.org)

---



이 강의록은 저작권법에 의해 보호받는 저작물로서 저작권자의 허락 없이 저작권재산권 일체 (복제권, 배포권, 대여권, 공연권, 공중전송권, 전시권, 2차적 저작물 작성권)를 침해 시 저작권법에 의거 처벌받을 수 있습니다.