데이터베이스

인공지능소프트웨어학과

담당교수: 김희숙

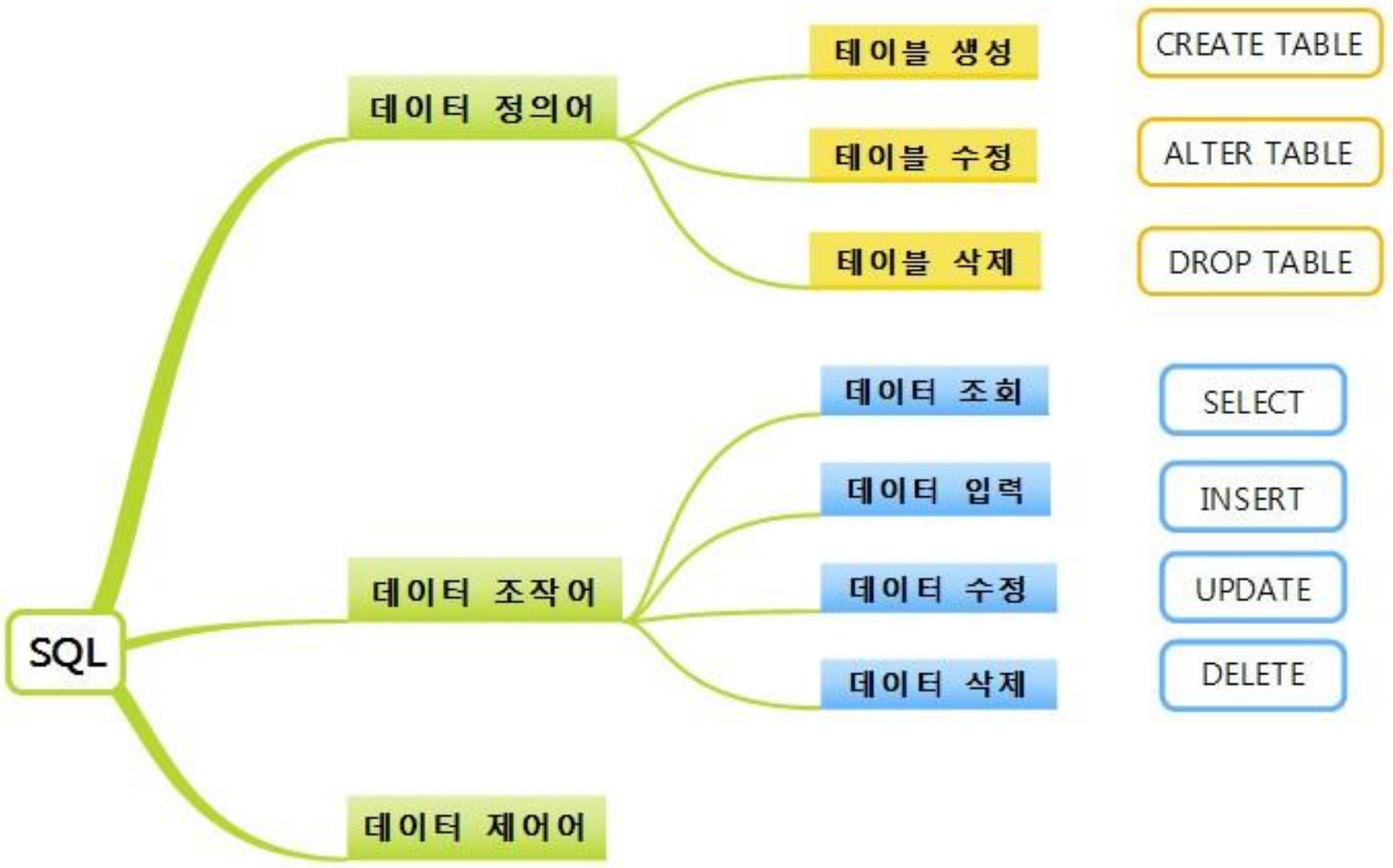
(jasmin11@hanmail.net)

데이터베이스

10주차

담당교수: 김희숙
(jasmin11@hanmail.net)

SQL


SQL
데이터 정의어
데이터 조작어
데이터 제어어
테이블 생성
테이블 수정
테이블 삭제
데이터 조회
데이터 입력
데이터 수정
데이터 삭제
CREATE TABLE
ALTER TABLE
DROP TABLE
SELECT
INSERT
UPDATE
DELETE

SQL: SELECT 문법(SELECT 기초)

(customer 테이블)

담당교수: 김희숙

(jasmin11@hanmail.net)

[예제] SELECT (데이터 조회)

[실습] (customer.sql)

```
CREATE  TABLE  customer (
  cno     char(4)          NOT  NULL  ,
  cname  varchar(10)   NOT  NULL  ,
  city     varchar(20)  ,
  point    int  ,
 CONSTRAINT  pk_customer_cno  PRIMARY  KEY(cno)
);
```

Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

테이블 이름	열 이름	데이터 형식	NULL 유무	기본키
customer	cno	char(4)	NOT NULL	PK
	cname	varchar(10)	NOT NULL	
	city	varchar(20)		
	point	int		

```
insert into customer values('c101','홍길동','서울',500);
insert into customer values('c102','임꺽정','인천',300);
insert into customer values('c103','박찬호','안양',800);
insert into customer values('c204','신동엽','과천',350);
insert into customer values('c205','정진우','고양',400);

/* 새 레코드를 추가하고 SELECT문 예제 실습하시오 */
insert into customer values('c307','정동우','서울', NULL);
```

[실습1] SELECT (데이터 조회)

[실습] (customer.sql)

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습] (SELECT 기초, NULL, LIKE, Order by)

```
/* customer(cno, cname, city, point) */
-- 고객(고객번호, 고객명, 거주지, 포인트)

-- 1-1) 테이블의 모든 열을 검색하라
-- 1-2) 테이블의 모든 열을 검색(필드명 사용)
-- 1-3) 고객의 고객명, 거주지를 검색하라(테이블의 특정 열을 검색)
-- 1-4) cname 은 성명, city는 거주지로 출력하라
-- (화면에 표시되는 열 이름 변경하여 검색)
-- 1-5) customer 테이블에서 거주지를 검색하라
-- 1-6) 거주지를 검색하는데 중복 행을 제거하여 한 번씩만 검색하라
```

[실습1] SELECT (데이터 조회)

[실습] (customer.sql)

Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습] (SELECT 기초, NULL, LIKE, Order by)

```
/* customer(cno, cname, city, point) */
-- 고객(고객번호, 고객명, 거주지, 포인트)

-- 1-1) 테이블의 모든 열을 검색하라
-- 1-2) 테이블의 모든 열을 검색(필드명 사용)
-- 1-3) 고객의 고객명, 거주지를 검색하라(테이블의 특정 열을 검색)
-- 1-4) cname 은 성명, city는 거주지로 출력하라
-- (화면에 표시되는 열 이름 변경하여 검색)
-- 1-5) customer 테이블에서 거주지를 검색하라
-- 1-6) 거주지를 검색하는데 중복 행을 제거하여 한 번씩만 검색하라
```

[실습2] SELECT (데이터 조회)

[실습] (customer.sql)

Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습] (SELECT 기초, NULL, LIKE, Order by)

```
/* customer(cno, cname, city, point) */
-- 고객(고객번호, 고객명, 거주지, 포인트)


-- 2-5) 포인트가 350 부터 500 사이인 고객이름, 거주지, 포인트를 검색하라
-- 부등호 사용
-- BETWEEN ... AND 사용
-- 2-6) 거주지가 서울 이거나 안양인 고객번호, 이름, 거주지를 검색하라
-- 부등호 사용
-- IN 사용
-- 2-7) 거주지가 서울이 아니거나 안양이 아닌 고객번호, 이름, 거주지를 검색하라
-- 부등호 사용
-- NOT IN 사용
```

❖ **SELECT 문법**

BETWEEN ... AND

IN

[실습3] SELECT (데이터 조회)

[실습] (customer.sql)

Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습] (SELECT 기초, NULL, LIKE, Order by)

```
/* customer(cno, cname, city, point) */
-- 고객(고객번호, 고객명, 거주지, 포인트)

-- 3-1) 정씨 성을 가진 고객의 모든 열을 검색하라
-- 3-2) 이름에 '동' 자가 들어가는 고객의 모든 열을 검색하라
-- 3-3) 이름의 세번째 글자가 '우' 자가 들어가는 고객의 모든 열을 검색하라
-- 3-4) 성이 홍씨, 박씨, 정씨인 고객을 검색하라
-- 성이 홍씨, 박씨, 정씨가 아닌 고객을 검색하라
-- 3-5) 포인트가 없는 고객의 번호, 이름, 포인트를 검색하라
-- 포인트가 있는 고객의 번호, 이름, 포인트를 검색하라
```

❖ 문자열 검색

% 0개 이상 검색

_ 1개 검색

❖ NULL 값 검색

IS NULL

IS NOT NULL

[실습3] SELECT (데이터 조회)

[실습] (customer.sql)

Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습 3]

-- customer(cno, cname, city, point)

-- 3-4) 성이 홍씨, 박씨, 정씨인 고객을 검색하라

-- 성이 홍씨, 박씨, 정씨가 아닌 고객을 검색하라

```
SELECT *
FROM   customer
WHERE  cname LIKE '홍%' or cname LIKE '박%' or cname LIKE '정%';
```

cno	cname	city	point
c101	홍길동	서울	500
c103	박찬호	안양	800
c205	정진우	고양	400
c307	정동우	서울	NULL

```
SELECT *
FROM   customer
WHERE  cname NOT LIKE '홍%' AND cname NOT LIKE '박%' AND cname NOT LIKE '정%';
```

cno	cname	city	point
c102	임꺽정	인천	300
c204	신동엽	과천	350

[실습3] SELECT (데이터 조회)

[실습] (customer.sql)

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습 3]

-- customer(cno, cname, city, point)

-- 3-5) 포인트가 없는 고객의 번호, 이름, 포인트를 검색하라

-- 포인트가 있는 고객의 번호, 이름, 포인트를 검색하라

```
SELECT cno, cname, point
FROM   customer
WHERE  point IS NULL;
```

cno	cname	point
c307	정동우	NULL

```
SELECT cno, cname, point
FROM   customer
WHERE  point IS NOT NULL;
```

cno	cname	point
c101	홍길동	500
c102	임꺽정	300
c103	박찬호	800
c204	신동엽	350
c205	정진우	400

[요약] SELECT (정렬)

❖ 정렬: ORDER BY

오름차순 (Ascending order) ASC

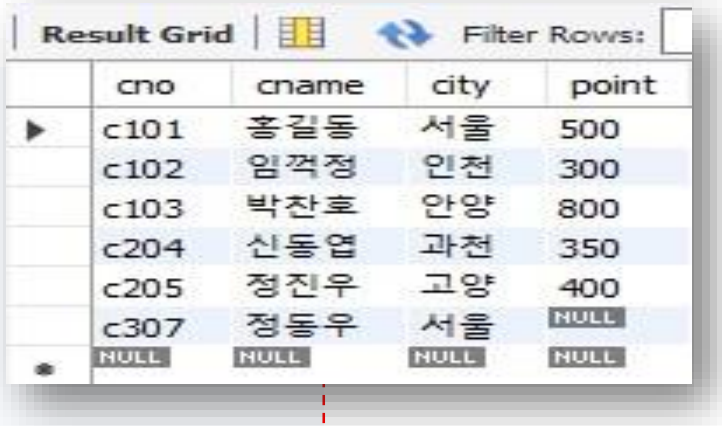
내림차순 (Descending order) DESC

```
SELECT
FROM
ORDER BY
```

```
SELECT
FROM
WHERE
GROUP BY
HAVING
ORDER BY
```

[실습4] SELECT (데이터 조회)

[실습] (customer.sql)



Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습] (SELECT 기초, NULL, LIKE, Order by)

```
/* customer(cno, cname, city, point) */
-- 고객(고객번호, 고객명, 거주지, 포인트)


-- 4-1) 고객 테이블에서 이름을 오름차순 정렬하라
-- 4-2) 거주지가 서울인 고객의 모든 데이터를 검색하는데, 이름의 오름차순 정렬하여 출력하라
-- 4-3) 거주지의 오름차순으로 정렬하고, 거주지가 같으면 포인트의 내림차순으로 정렬하라
-- 4-4) 포인트가 많은 순으로(내림차순) 먼저 정렬하고, 같은 포인트는 이름의 오름차순으로 정렬하고
-- 이름이 같으면 거주지의 오름차순으로 정렬하여 검색하라
-- 4-5) 다음의 의미는?
SELECT      cno,  cname,  city,  point
FROM        customer
ORDER  BY  3;
```

[실습4] SELECT (정렬)

[실습] (customer.sql)

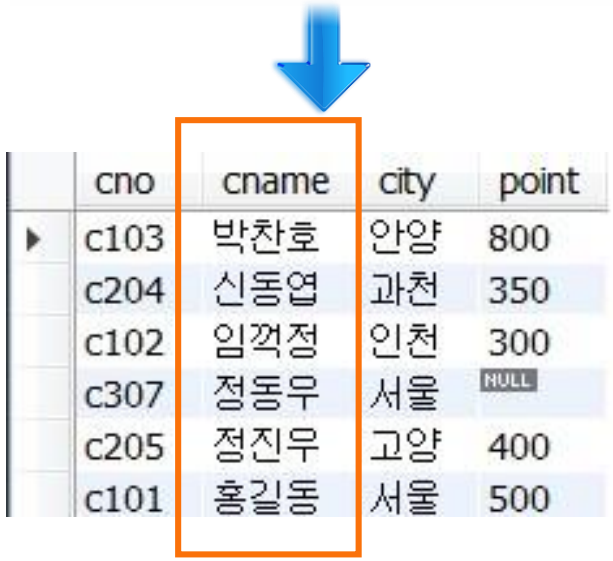
cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL

[실습 4]

-- customer(cno, cname, city, point)

-- 4-1) 고객 테이블에서 이름을 오름차순 정렬하라

```
SELECT *
FROM   customer
ORDER BY cname   ASC;
```



cno	cname	city	point
c103	박찬호	안양	800
c204	신동엽	과천	350
c102	임꺽정	인천	300
c307	정동우	서울	NULL
c205	정진우	고양	400
c101	홍길동	서울	500

[실습4] SELECT (정렬)

[실습] (customer.sql)

Result Grid | Filter Rows:

cno	cname	city	point
c101	홍길동	서울	500
c102	임꺽정	인천	300
c103	박찬호	안양	800
c204	신동엽	과천	350
c205	정진우	고양	400
c307	정동우	서울	NULL
NULL	NULL	NULL	NULL

[실습 4]

-- customer(cno, cname, city, point)

-- 4-2) 거주지가 서울인 고객의 모든 데이터를 검색하는데,

이름의 오름차순 정렬하여 출력하라

```
SELECT *
FROM   customer
WHERE  city = '서울'
ORDER BY cname;
```

cno	cname	city	point
c307	정동우	서울	NULL
c101	홍길동	서울	500

[실습4] SELECT (정렬)

[실습] (customer.sql)

[실습 4]

-- customer(cno, cname, city, point)

-- 4-3) 거주지의 오름차순으로 정렬하고, 거주지가 같으면

포인트의 내림차순으로 정렬하라

```
SELECT *
FROM   customer
ORDER BY city ASC, point DESC;
```

cno	cname	city	point
c205	정진우	고양	400
c204	신동엽	과천	350
c101	홍길동	서울	500
c307	정동우	서울	NULL
c103	박찬호	안양	800
c102	임꺽정	인천	300

[실습4] SELECT (정렬)

[실습] (customer.sql)

[실습 4]

-- customer(cno, cname, city, point)

-- 4-4) 포인트가 많은 순으로(내림차순) 먼저 정렬하고,

-- 같은 포인트는 이름의 오름차순으로 정렬하고

-- 이름이 같으면 거주지의 오름차순으로 정렬하여 검색하라

```
SELECT *
FROM   customer
ORDER BY point DESC, cname ASC, city ASC;
```

cno	cname	city	point
c103	박찬호	안양	800
c101	홍길동	서울	500
c205	정진우	고양	400
c204	신동엽	과천	350
c102	임꺽정	인천	300
c307	정동우	서울	NULL

[실습4] SELECT (정렬)

[실습] (customer.sql)

	cno	cname	city	point
▶	c101	홍길동	서울	500
	c102	임꺽정	인천	300
	c103	박찬호	안양	800
	c204	신동엽	과천	350
	c205	정진우	고양	400
	c307	정동우	서울	NULL
*	NULL	NULL	NULL	NULL

[실습 4]

-- customer(cno, cname, city, point)

-- 4-5) 다음의 의미는?

```
SELECT    cno, cname, city, point
FROM      customer
ORDER BY  3;
```

```
SELECT    cno, cname, city, point
FROM      customer
ORDER BY  3;
```

	cno	cname	city	point
▶	c205	정진우	고양	400
	c204	신동엽	과천	350
	c101	홍길동	서울	500
	c307	정동우	서울	NULL
	c103	박찬호	안양	800
	c102	임꺽정	인천	300

실습

담당교수: 김희숙
(jasmin11@hanmail.net)

SQL: 그룹화(group by)

10주차

담당교수: 김희숙
(jasmin11@hanmail.net)

[요약] 집계함수

-- 사원(직원코드,성명,직책,연봉

[실습] 집계함수, 그룹화

```
18 • select * from 성적;
```

Result Grid | Filter Rows:

이름	점수
강성범	68
박찬호	75
선동열	70
신동엽	NULL
임꺽정	60
차범근	75
홍길동	87
홍명보	90
NULL	NULL

```
40 • SELECT * FROM 성적2;
```

Result Grid | Filter Rows:

이름	과목	점수
강성범	수학	68
박찬호	국어	75
선동열	영어	70
신동엽	영어	NULL
임꺽정	수학	60
차범근	수학	75
홍길동	영어	87
홍명보	수학	90
NULL	NULL	NULL

COUNT(*): 널 값 포함
COUNT(필드): 널 값 제외

SELECT
FROM

SUM()
AVG()
MAX()
MIN()

SELECT
FROM
GROUP BY
HAVING

-- 성적(이름, 점수)
1) 학생수를 구하시오
2) 시험에 응시한 학생를 구하시오
3) 점수의 평균을 구하시오

-- 성적2(이름, 과목, 점수)
4) 과목별 응시한 학생수를 구하시오
5) 과목별 평균점수를 구하시오

[실습 1-01] SELECT (집계함수)

[실습] (sungjuk_group.sql)

[실습] (집계함수)

```
-- 집계함수
CREATE TABLE 성적(
        이름 varchar(9) NOT NULL primary key,
        점수 int
);

INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '홍길동', 87 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '임꺽정', 60 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '박찬호', 75 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '선동열', 70 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '홍명보', 90 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '차범근', 75 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '강성범', 68 );
INSERT INTO 성적 ( 이름, 점수 ) VALUES ( '신동엽', null);
```

[실습 1-01] SELECT (집계함수)

[실습] (sungjuk_group.sql)

[실습] (집계함수)

-- 성적(이름, 점수)

-- 1-1) 최고 점수를 검색하라

-- 1-2) 최저 점수를 검색하라

-- 1-3) 점수합계를 검색하라

-- 1-4) 평균점수를 검색하라

-- 1-5) 학생수는 모두 몇 명인지 검색하라

-- 1-6) 시험에 응시한 학생수는 모두 몇 명인지 검색하라

[실습 1-02] SELECT (그룹화)

[실습] (sungjuk_group.sql)

[실습] (그룹화)

```
-- 그룹화
CREATE TABLE 성적2 (
  이름 varchar(9) NOT NULL primary key ,
  과목 varchar(8),
  점수 int
);

INSERT INTO 성적2 VALUES ('홍길동', '영어',87 );
INSERT INTO 성적2 VALUES ('임꺽정', '수학',60 );
INSERT INTO 성적2 VALUES ('박찬호', '국어',75 );
INSERT INTO 성적2 VALUES ('선동열', '영어',70 );
INSERT INTO 성적2 VALUES ('홍명보', '수학',90 );
INSERT INTO 성적2 VALUES ('차범근', '수학',75 );
INSERT INTO 성적2 VALUES ('강성범', '수학',68 );
INSERT INTO 성적2 VALUES ('신동엽', '영어',null);
```

이름	과목	점수
강성범	수학	68
박찬호	국어	75
선동열	영어	70
신동엽	영어	NULL
임꺽정	수학	60
차범근	수학	75
홍길동	영어	87
홍명보	수학	90
NULL	NULL	NULL

[실습 1-02] SELECT (그룹화)

[실습] (sungjuk_group.sql)

[실습] 그룹화, 부분합(GROUP BY)

```
-- 성적2(이름, 과목, 점수)
-- 2-1) 각 과목수는 몇 개인지 검색하라(DISTINCT 사용)
-- 2-2) 과목별 수강생은 몇 명인지 검색하라(GROUP BY)
-- 2-3) 과목별 평균점수를 검색하라(GROUP BY)
SELECT  과목 ,  COUNT(점수)
FROM   성적2
GROUP  BY  과목;

-- 2-4) 과목별 평균점수 75 보다 높은 과목의
-- 과목별 평균점수를 검색하라(HAVING)
-- 2-5) 점수가 70 이상인 과목이름, 과목 평균점수를
-- 과목의 과목별 평균점수가 75 이상인 것만
-- 과목별 평균점수가 높은 순으로 검색하라(ORDER  BY)
```

```
40 •  SELECT * FROM 성적2;
```

Result Grid | Filter Rows:

이름	과목	점수
강성범	수학	68
박찬호	국어	75
선동열	영어	70
신동엽	영어	NULL
임꺽정	수학	60
차범근	수학	75
홍길동	영어	87
홍명보	수학	90
NULL	NULL	NULL

[실습] [실습 1-01](MySQL) 집계함수(성적) (ans)

```
-- [실습] (집계함수)
-- 성적(이름, 점수)
-- 1-1) 최고 점수를 검색하라
SELECT  MAX(점수)
FROM    성적;

-- 1-2) 최저 점수를 검색하라
SELECT  MIN(점수)
FROM    성적;

-- 1-3) 점수합계를 검색하라
SELECT  SUM(점수)
FROM    성적;

-- 1-4) 평균점수를 검색하라
SELECT  AVG(점수)
FROM    성적;
```

```
-- [실습] (집계함수)
-- 성적(이름, 점수)
-- 1-5) 학생수는 모두 몇 명인지 검색하라
SELECT  COUNT(*)
FROM    성적;

-- 1-6) 시험에 응시한 학생수는 모두 몇 명인지 검색하라
-- (MySQL/MS SQL)
SELECT  COUNT(점수)  as '응시 학생수'
FROM    성적;

-- (Oracle) 출력 필드명 띄어쓰기
SELECT  COUNT(점수)  as  "응시 학생수"
FROM    성적;
```

```
18 •  select * from 성적;
```

Result Grid | Filter Rows:

이름	점수
강성범	68
박찬호	75
선동열	70
신동엽	NULL
임꺽정	60
차범근	75
홍길동	87
홍명보	90
NULL	NULL

* COUNT(*) : NULL 포함하여 계산
* COUNT(필드) : NULL 제외하여 계산

[실습] [실습 1-02](MySQL) 그룹화(성적2) (ans)

```
-- [실습] (그룹화)
-- 성적2(이름, 과목, 점수)
-- 2-1) 각 과목수는 몇 개인지 검색하라(DISTINCT 사용)
SELECT   COUNT(DISTINCT 과목)
FROM    성적2

-- 2-2) 과목별 수강생은 몇 명인지 검색하라(GROUP BY)
SELECT   과목,  COUNT(점수)
FROM    성적2
GROUP   BY   과목;

-- 2-3) 과목별 평균점수를 검색하라(GROUP BY)
SELECT   과목,  AVG(점수)
FROM    성적2
GROUP   BY   과목;
```

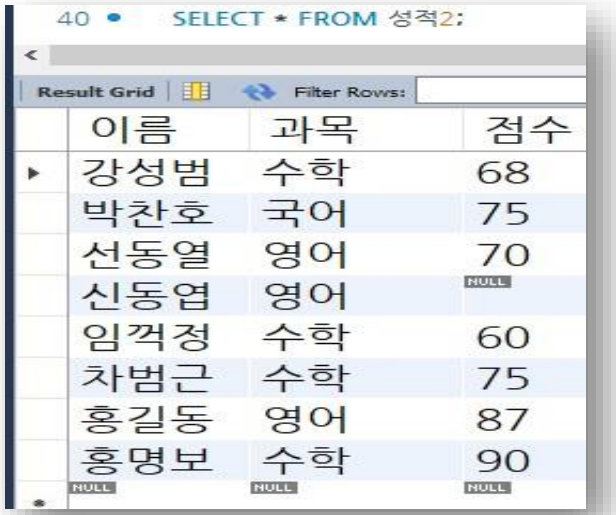
이름	과목	점수
강성범	수학	68
박찬호	국어	75
선동열	영어	70
신동엽	영어	NULL
임꺽정	수학	60
차범근	수학	75
홍길동	영어	87
홍명보	수학	90
NULL	NULL	NULL

[실습] [실습 1-02](MySQL) 그룹화(성적2) (ans)

```
-- [실습] (그룹화)
-- 성적2(이름, 과목, 점수)
-- 2-4) 과목별 평균점수 75 보다 높은 과목의 과목별 평균점수를 검색하라 (HAVING)
SELECT  과목,  AVG(점수)
FROM   성적2
GROUP  BY  과목
HAVING   AVG(점수) >= 75 ;

-- 2-5) 점수가 70 이상인 과목이름, 과목 평균점수를 과목의 과목별 평균점수가 75 이상인 것만 과목별 평균점수가 높은 순으로 검색하라(ORDER  BY)

SELECT  과목,  AVG(점수)
FROM   성적2
WHERE  점수 >= 70
GROUP  BY  과목
HAVING   AVG(점수) >= 75
ORDER BY AVG(점수) DESC;
```




```
/* SELECT 문법 순서 */
SELECT
FROM
WHERE
GROUP  BY
HAVING
ORDER  BY
```

```
-- ROUND() 함수
select 과목, ROUND(AVG(점수), 1)
from 성적2
group by 과목;
```