

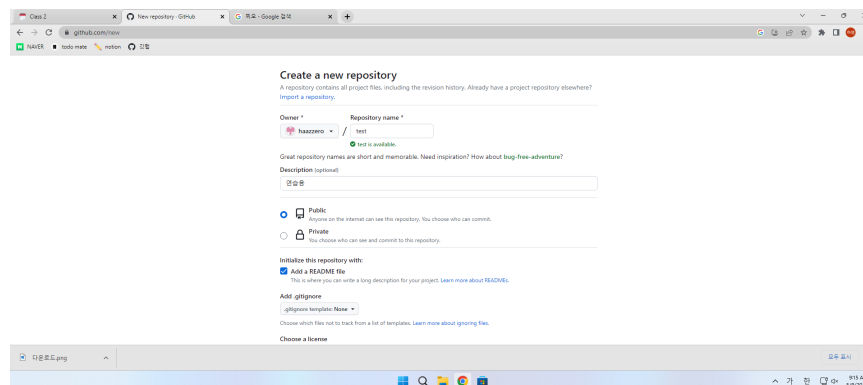


# Class 2\_정하영

Dates	@2023년 5월 9일
Type	Lesson
Topic	

## ▼ GITHUB 사용법

- GITHUBS DESKTOP
  - Create a new repository
    - 3번째 선택, choose(경로)
    - 이름과 local path가 동일하면 안됨.
    - 그 폴더에 파일 옮김
    - 그럼 자동동기화
    - 메모 후 publish



## ▼ 개발도구

- 자바 (이클립스, 인텔리제이)
- 파이썬(파이참)
- R(스튜디오)

## • GITHUB 연관툴

- gitkraken(유료)
- github desktop(무료, 깃헙 자체 제작)

- Visual Studio Code (모든언어)
- ANACONDA (파이썬,알파,베타,셋타) : 데이터분석 기본
- cf. intelliJ (개발툴, 유료)
- cf. google keep 개발자 비밀번호

### ▼ 참고 “” 와 “, 1

java는 확실히 차이가 있음.

SQL은 구분하지는 않는다 (권장은 “)

SQL과 R은 1부터 시작한다.



## 연산자

### ▼ 비교 연산자



#### 비교연산자

= 같다

≠, !=, (<>) 다르다

> 크다

<작다

> =크거나 같다

< =작거나 같다

- = 같다 예시

```
SELECT *
FROM employees
WHERE employee_id = 100;
```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD PRES	24000	(null)	(null)	90

## ▼ 논리 연산자



논리연산자

**AND** 모든 조건을 동시에 다 만족할 때만 true

**OR** 조건 중 하나만 만족해도 true

**NOT** 조건의 반대 결과를 반환한다

## 기본형 문법

```
SELECT *
FROM 테이블
WHERE 조건1
AND 조건2;
```

### • AND

```
SELECT *
FROM employees
WHERE salary > 4000
AND job_id = 'IT_PROG';
```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT PROG	9000	(null)	102	60
2	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-07	IT PROG	6000	(null)	103	60
3	105	David	Austin	DAUSTIN	590.423.4569	25-JUN-05	IT PROG	4800	(null)	103	60
4	106	Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06	IT PROG	4800	(null)	103	60
5	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07	IT PROG	4200	(null)	103	60

### • OR

```
SELECT *
FROM employees
WHERE salary > 10000
OR job_id = 'IT_PROG';
```

1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD	PRES	24000	(null)	(null)	90
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD	VP	17000	(null)	100	90
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD	VP	17000	(null)	100	90
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT	PROG	9000	(null)	102	60
5	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-07	IT	PROG	6000	(null)	103	60
6	105	David	Austin	DAUSTIN	590.423.4569	25-JUN-05	IT	PROG	4800	(null)	103	60
7	106	Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06	IT	PROG	4800	(null)	103	60
8	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07	IT	PROG	4200	(null)	103	60
9	108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI	MGR	12008	(null)	101	100

## • AND와 OR 섞어쓰기

```
SELECT *
FROM employees
WHERE salary > 4000
AND job_id = 'IT_PROG'
OR job_id = 'FI_ACCOUNT';
```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID	
1	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT	PROG	9000	(null)	102	60
2	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-07	IT	PROG	6000	(null)	103	60
3	105	David	Austin	DAUSTIN	590.423.4569	25-JUN-05	IT	PROG	4800	(null)	103	60
4	106	Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06	IT	PROG	4800	(null)	103	60
5	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07	IT	PROG	4200	(null)	103	60
6	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI	ACCOUNT	9000	(null)	108	100
7	110	John	Chen	JCHEN	515.124.4269	28-SEP-05	FI	ACCOUNT	8200	(null)	108	100
8	111	Ismael	Sciarra	ISCIARRA	515.124.4369	30-SEP-05	FI	ACCOUNT	7700	(null)	108	100
9	112	Jose Manuel	Urman	JMURMAN	515.124.4469	07-MAR-06	FI	ACCOUNT	7800	(null)	108	100
10	113	Luis	Popp	LPOPP	515.124.4567	07-DEC-07	FI	ACCOUNT	6900	(null)	108	100

→ 4000은 꼭 넘어야하고, 그리고 아이디에 prog나 account가 포함된 값 출력

## • NOT

```
SELECT *
FROM employees
WHERE employee_id <> 105;
```

1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD	P
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD	V
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD	V
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT	P
5	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-07	IT	P
6	106	Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06	IT	P
7	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07	IT	P
8	108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI	M
9	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI	P
10	110	John	Chen	JCHEN	515.124.4269	28-SEP-05	FI	P
11	111	Ismael	Sciarra	ISCIARRA	515.124.4369	30-SEP-05	FI	P
12	112	Jose Manuel	Urman	JMURMAN	515.124.4469	07-MAR-06	FI	P

→ 아이디에 105 포함된 것 제외 !

- IS NULL

```
SELECT *
FROM employees
WHERE manager_id IS NULL;
```

- IS NOT NULL

```
SELECT *
FROM employees
WHERE manager_id IS NOT NULL;
```



## 함수

### ▼ 문자 관련 함수

### ▼ 함수 사용

| 단일 행 함수

| 그룹 함수

- **단일 행 함수** : 특정 행에 적용, 데이터 값을 하나씩 계산
  - 대문자/소문자/첫글자만
  - BOY / boy / Boy

```
SELECT last_name,
       LOWER(last_name),
       UPPER(last_name),
       email,
       INITCAP(email)
FROM employees;
```

1	Abel	abel	ABEL	EABEL	Eabel
2	Ande	ande	ANDE	SANDE	Sande
3	Atkinson	atkinson	ATKINSON	MATKINSO	Matkinsc
4	Austin	austin	AUSTIN	DAUSTIN	Daustin
5	Baer	baer	BAER	HBAER	Hbaer
6	Baida	baida	BAIDA	SBAIDA	Sbaida
7	Banda	banda	BANDA	ABANDA	Abanda
8	Bates	bates	BATES	EBATES	Ebates
9	Bell	bell	BELL	SBELL	Sbell
10	Bernstein	bernstein	BERNSTEIN	DBERNSTE	Dbernste
11	Bissot	bissot	BISSOT	LBISSOT	Lbissot

## • 글자 자르기 substr

SUBSTR('원본글자', 시작위치, 자를 개수)

```
SELECT job_id, SUBSTR(job_id,1,2)
FROM employees;
```

job_id	SUBSTR(job_id,1,2)
1 AC ACCOUNT	AC
2 AC MGR	AC
3 AD ASST	AD
4 AD PRES	AD
5 AD VP	AD
6 AD VP	AD
7 FI ACCOUNT	FI

## • 글자바꾸기

- 특정 문자를 찾아서 변경(replace)
- 참고로 진짜 바꾸는게 X 출력을 그렇게 보여주는 것

REPLACE('문자열', '찾을 문자', '바꿀문자')

```
SELECT job_id, REPLACE(job_id, 'ACCOUNT', 'ACC')
FROM employees;
```

JOB_ID	REPLACE(JOB_ID, 'ACCOUNT', 'ACC')
1 AC ACCOUNT	AC ACC
2 AC MGR	AC MGR
3 AD ASST	AD ASST
4 AD PRES	AD PRES
5 AD VP	AD VP
6 AD VP	AD VP
7 FI ACCOUNT	FI ACC
8 FI ACCOUNT	FI ACC
9 FI ACCOUNT	FI ACC
10 FI ACCOUNT	FI ACC
11 FI ACCOUNT	FI ACC
12 FI MGR	FI MGR
13 HR REP	HR REP

## • LPAD, RPAD

- 특정 문자로 자리 채우기
- 왼쪽부터 채우기, 오른쪽 채우기

LPAD('문자열', 만들어질 자리수, '채울 문자')

```
SELECT first_name ,LPAD(first_name,12,'*')
FROM employees;
```

또는  
빈칸 가능

```
SELECT first_name ,LPAD(first_name,12,' ')
FROM employees;
```

또는

```
SELECT first_name ,RPAD(first_name,12,'*')
FROM employees;
```

SQL | Fetched 30 rows in 0.028 seconds

	FIRST_NAME	LPAD(FIRST_NAME,12,'*')
1	Ellen	* * * * * Ellen
2	Sundar	* * * * * Sundar
3	Mozhe	* * * * * Mozhe
4	David	* * * * * David
5	Hermann	* * * * * Hermann
6	Shelli	* * * * * Shelli
7	Amit	* * * * * Amit
8	Elizabeth	* * * Elizabeth
9	Sarah	* * * * * Sarah
10	David	* * * * * David
11	Laura	* * * * * Laura
12	Hermann	* * * Hermann

	FIRST_NAME	LPAD(FIRST_NAME,12,'')
1	Ellen	Ellen
2	Sundar	Sundar
3	Mozhe	Mozhe
4	David	David
5	Hermann	Hermann
6	Shelli	Shelli
7	Amit	Amit
8	Elizabeth	Elizabeth
9	Sarah	Sarah
10	David	David
11	Laura	Laura

- LTRIM / RTRIM : 삭제하기

LTRIM('문자열')/열이름, '삭제할 문자')

```
SELECT job_id , LTRIM(job_id,'F')
FROM employees;
```

또는

```
SELECT job_id , LTRIM(job_id,'F'), RTRIM(job_id,'T')
FROM employees;
```



	⚡ JOB_ID	⚡ LTRIM(JOB_ID,'F')
1	AC ACCOUNT	AC ACCOUNT
2	AC MGR	AC MGR
3	AD ASST	AD ASST
4	AD PRES	AD PRES
5	AD VP	AD VP
6	AD VP	AD VP
7	FI ACCOUNT	I ACCOUNT

	⚡ JOB_ID	⚡ LTRIM(JOB_ID,'F')	⚡ RTRIM(JOB_ID,'T')
1	AC ACCOUNT	AC ACCOUNT	AC ACCOUN
2	AC MGR	AC MGR	AC MGR
3	AD ASST	AD ASST	AD ASS
4	AD PRES	AD PRES	AD PRES
5	AD VP	AD VP	AD VP
6	AD VP	AD VP	AD VP
7	FI ACCOUNT	I ACCOUNT	FI ACCOUNT

- 가상 테이블? **dual**

- 테이블을 쓸 일은 없지만 문법상 가상의 테이블이 필요하다. → dual
- dual 테이블은 dummy 테이블, 특정 테이블을 사용하지 않고 문법적으로 오류를 회피하고자 할 때 사용하는 일종의 가상의 테이블로 생각

```
SELECT 'test'
FROM dual;
```

```
SELECT LTRIM(' name', ' ')
FROM dual;
```

Script Output x	Query Result x
SQL   All Rows Fetched: 1 in 0.007 seconds	
⚡ LTRIM(NAME, ' ')	
1 name	

## ▼ 그룹 함수 : 그룹 전체 적용, 여러개 값을 그룹으로 계산

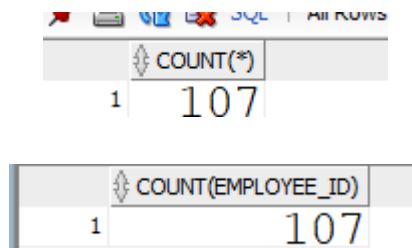
여러 행에 함수가 적용이 되어 하나의 결과를 나타냄

### count 갯수 (null 값도 센다)

```
SELECT COUNT (employee_id)
FROM employees;
```

OR

```
SELECT COUNT (*)
FROM employees;
```



COUNT(*)
107

COUNT(EMPLOYEE_ID)
107

→ count는 특성상 null도 계산하기 때문에 어떠한 열로도 동일한 결과값이 나오기 때문에 '\*'로 주로 사용.

### sum 합계 (null 값 제외 계산)

### avg 평균 (null 값 제외 계산)

### max 최대값 (null 값 제외 계산)

### min 최소값 (null 값 제외 계산)

```
SELECT AVG (salary) 급여평균,
SUM(salary) 급여합계,
MAX(salary) 최대급여,
MIN(salary) 최저급여
FROM employees;
```

	급여평균	급여합계	최대급여	최저급여
1	6461.831775700934579439252336448598130841	691416	24000	2100

```
SELECT MAX (first_name) 최대값,
       MIN(first_name) 최저값
FROM employees;
```

	최대값	최저값
1	Winston	Adam

- GROUPN 그룹으로 묶기

```
SELECT job_id, AVG(salary)
FROM employees
GROUP BY job_id;
```

	JOB_ID	AVG(SALARY)
1	IT PROG	5760
2	AC MGR	12008
3	AC ACCOUNT	8300
4	ST MAN	7280
5	PU MAN	11000
6	AD ASST	4400
7	AD VP	17000
8	SH CLERK	3215
9	FI ACCOUNT	7920
10	FI MGR	12008
11	PU CLERK	2780
12	SA MAN	12200
13	MTZ MAN	12000

- GROUP BY + ORDER BY

```
SELECT job_id, AVG(salary) 급여평균, SUM(salary) 급여합계, COUNT(*)
부서인원수 FROM employees
GROUP BY job_id
ORDER BY 급여평균 DESC, 부서인원수 DESC;
```

	JOB_ID	급여평균	급여합계	부서인원수
1	AD PRES	24000	24000	1
2	AD VP	17000	34000	2
3	MK MAN	13000	13000	1
4	SA MAN	12200	61000	5
5	AC MGR	12008	12008	1
6	FI MGR	12008	12008	1
7	PU MAN	11000	11000	1
8	PR REP	10000	10000	1
9	SA REP	8350	250500	30
10	AC ACCOUNT	8300	8300	1

## ▼ 숫자 관련 함수

- **ROUND - 반올림**

ROUND('열이름', 반올림할 위치)

```
SELECT salary 월급, salary/30 일급
, ROUND(salary/30, 0) 자리0
, ROUND(salary/30, 1) 자리1까지
, ROUND(salary/30, -1) 자리1부터
FROM employees;
```

[illegible]

- **TRUNC** - 버림, 절삭

TRUNC( '열이름', 버림할 위치)

```
SELECT salary 월급, salary/30 일급
, TRUNC(salary/30, 0) 자리0
, TRUNC(salary/30, 1) 자리1까지
, TRUNC(salary/30, -1) 자리1부터
FROM employees;
```

[illegible]

▼ 데이터 형 변환(Casting) 교재 99p 참고

### ▼ 자동 형변환 - 필요시 데이터형을 자동으로 변환

- = 목시적 형변환

	$1+2'$	
1	3	

```
SELECT 1 + '2'
FROM dual;
```

→ 숫자 2는 작은 따옴표가 붙어있어 숫자가 아닌 문자,

하지만 예외가 발생하지 않고 원하는 방향으로 계산이 이루어짐. 시스템이 자동  
으로 알아서 숫자로 변환하여 계산하였기 때문

→ 그러나 항상 사용자가 원하는 의도대로 변환이 자동 형변환이 완벽하게 이루어지는 X

Therefore 자동형변환이 잘 되더라도 수동형변환을 의도적으로 명시하는 것이 바람직함.

## ▼ 수동 형변환 - 수동으로 데이터형 변환

= 명시적 형변환

- | TO\_CHAR : 문자로 형변환
- | TO\_NUMBER : 숫자로 형변환
- | TO\_DATE : 날짜로 형변환

```
SELECT 1 + TO_NUMBER('2')
FROM dual;

//결과 동일
```

VARCHAR2 (varchar) → Number (integer)

NUMBER → VARCHAR2

## ▼ 그 외 함수

- null이 있는 경우?

```
SELECT *
FROM employees
ORDER BY commission_pct;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT
165	David	Lee	DLEE	011.44.1346.529268	23-FEB-08	SA REP	6800	0.1
179	Charles	Johnson	CJOHNSON	011.44.1644.429262	04-JAN-08	SA REP	6200	0.1
164	Mattea	Marvins	MMARVINS	011.44.1346.329268	24-JAN-08	SA REP	7200	0.1
167	Amit	Banda	ABANDA	011.44.1346.729268	21-APR-08	SA REP	6200	0.1
173	Sundita	Kumar	SKUMAR	011.44.1343.329268	21-APR-08	SA REP	6100	0.1
166	Sundar	Ande	SANDE	011.44.1346.629268	24-MAR-08	SA REP	6400	0.1
163	Danielle	Greene	DGREENE	011.44.1346.229268	19-MAR-07	SA REP	9500	0.15

- Salary와 commision\_pct 값을 계산하려고 하면 문제가 발생.  
이유는 커미션을 지급받지 않는 직원 = null 값이 있기에 문제 발생  
→ null로 계산하면 결과는 null이 된다.
- null 값을 1로 치환하여 계산하면 전체 계산에 문제를 발생시키지 않게 된다.

```
SELECT salary * commiss
FROM employees
ORDER BY commission_pct
```

Script Output x Query Result: \*

SQL | Fetched 100 rows in 0.024 seconds

	SALARY*COMMISSION_PCT
31	2250
32	3500
33	3325
34	3150
35	5600
36	(null)
37	(null)
38	(null)
39	(null)
40	(null)
41	(null)
42	(null)
43	(null)
44	(null)
...	...

- NVL 함수

- NVL은 null값을 특정 값으로 치환하여 계산이 이루어지도록 처리

```
SELECT salary * NVL(commission_pct,1)
FROM employees
ORDER BY commission_pct;
```

\*\* null 값이 있으면 1을 곱해줘

1	680
2	620
3	720
4	620
5	610
6	640
7	1425
8	1050
9	1050
10	1095
11	1110
12	1720
13	2000
14	1920

- **DECODE : 조건 처리하기**

**DECODE(열이름, 조건값, 치환값, 기본값)**

치환값 - 조건을 만족할 경우

기본값 - 조건을 만족하는 경우

```
SELECT department_id,
       employee_id,
       first_name,
       salary 원래급여,
       DECODE(department_id, 60, salary * 1.1, salary),
       DECODE(department_id, 60, '급여인상', ' ')
FROM employees;
```

#	EMPLOYEE_ID	FIRST_NAME	원래급여	DECODE(DEPARTMENT_ID,60,SALARY*1.1,SALARY)	DECODE(DEPARTMENT_ID,60,'급여인상',' ')
1	100	Steven	24000	24000	
2	101	Neena	17000	17000	
3	102	Lex	17000	17000	
4	103	Alexander	9000	9900	급여인상
5	104	Bruce	6000	6600	급여인상
6	105	David	4800	5280	급여인상
7	106	Valli	4800	5280	급여인상
8	107	Diana	4200	4620	급여인상
9	108	Nancy	12008	12008	
10	109	Daniel	9000	9000	
11	110	John	8200	8200	
12	111	Tomas	7700	7700	

- **CASE : 경우의 수가 여러개일 경우 즉 복잡한 조건 처리**

```
SELECT job_id, employee_id, first_name, salary,
       CASE
         WHEN salary >= 9000 THEN '상급개발자'
         WHEN salary >= 5000 THEN '중급개발자'
         ELSE '하급개발자'
       END AS 구분
FROM employees
WHERE job_id = 'IT_PROG';
```

#	JOB_ID	EMPLOYEE_ID	FIRST_NAME	SALARY	구분
1	IT_PROG	103	Alexander	9000	상급개발자
2	IT_PROG	104	Bruce	6000	중급개발자
3	IT_PROG	105	David	4800	하급개발자
4	IT_PROG	106	Valli	4800	하급개발자
5	IT_PROG	107	Diana	4200	하급개발자



- 순위 매기기 3가지 방법 numbering

RANK : 공통 순위 만큼 건너 뛰어 순위 매기기 1,2,2,4

DENSE\_RANK : 공통순위를 건너 뛰지 않고 순위 1,2,2,3

ROW\_NUMBER : 공통 순위 없이 출력 1,2,3,4

Q. 그럼 같은 점수면 가나다 기준? 주로 고유번호

```
SELECT employee_id, first_name, salary,
       RANK()OVER(ORDER BY salary DESC) rank순위,
       DENSE_RANK()OVER(ORDER BY salary DESC) dense순위,
       ROW_NUMBER()OVER(ORDER BY salary DESC) row순위
FROM employees;
```

	EMPLOYEE_ID	FIRST_NAME	SALARY	RANK순위	DENSE순위	ROW순위
4	145	John	14000	4	3	4
5	146	Karen	13500	5	4	5
6	201	Michael	13000	6	5	6
7	108	Nancy	12008	7	6	7
8	205	Shelley	12008	7	6	8
9	147	Alberto	12000	9	7	9
10	168	Lisa	11500	10	8	10
11	114	Den	11000	11	9	11
12	148	Gerald	11000	11	9	12
13	174	Ellen	11000	11	9	13
14	149	Eleni	10500	14	10	14
15	162	Clara	10500	14	10	15
16	156	Janette	10000	16	11	16
17	150	Peter	10000	16	11	17
18	204	Hermann	10000	16	11	18

## ▼ DML / DDL / DCL

**DML 데이터 조작 언어 Data Manipulation Language**

SELECT 조회 read

INSERT 삽입 create

UPDATE 수정 update

## DELETE 삭제 delete

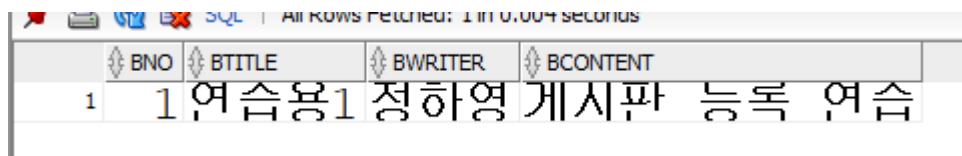
### (웹개발 : CRUD)

- insert

```
INSERT INTO board  
VALUES(1, '연습용1', '정하영', '게시판 등록 연습');
```

확인

```
SELECT * FROM board;
```



	BNO	BTITLE	BWRITER	BCONTENT
1	1	연습용1	정하영	게시판 등록 연습



### 삭제

- DML delete : 데이터만 삭제
- DDL truncate : 구조를 남기고 데이터만 삭제
- DDL drop 구조 포함 데이터 전체 완전히 삭제

## DDL 데이터 정의 언어 Data Definition Language

DB 생성, Table 생성, 삭제 drop

- 새로운테이블 만들기 ~

```
CREATE TABLE 테이블 이름  
(열이름2 속성,  
열이름2 속성  
...  
...);
```

```
CREATE TABLE 테이블 이름
(열이름2 속성,
열이름2 속성
. . .
. . .);
```

진짜 만들어보자

```
CREATE TABLE board
(bno number,
btitle VARCHAR2(50),
bwriter VARCHAR2(10),
bcontent VARCHAR2(500)
);
```

```
-----
GROUP BY job_id
Error report -
Unknown Command

Table BOARD created.
```

실습!

member table



### 이름 정의 방법

- 동일한 이름 테이블이 존재X
- 예약어 즉 이미 사용중인 명령어 등으로는 이름을 사용할 수 없다
- 반드시 문자로 시작해야 한다.
  - 한글도 쓸 수 있지만 절대 사용하지 말자
- 의미있는 단어 사용

지금 말하는건 다 FM스러운 방법

- ALTER TABLE ~ ADD : 테이블 수정(항목변경)

```
ALTER TABLE member  
ADD (mname VARCHAR2(10));
```

- ALTER TABLE ~ MODIFY: (항목수정)?

```
ALTER TABLE member MODIFY (mnumber number);
```

- 이름을 바꿔보자

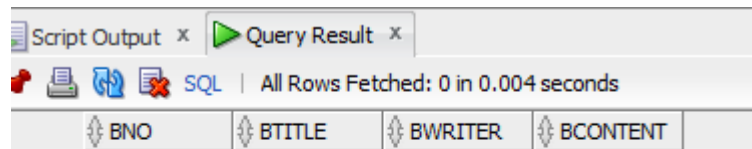
```
ALTER TABLE member RENAME COLUMN MNO to MNUMBER1;
```

- 열을 삭제해보자

```
ALTER TABLE member DROP COLUMN MNAME;
```

- 열 구조자체는 남겨놓는데 안에 데이터는 다 삭제하자

```
TRUNCATE TABLE board;
```

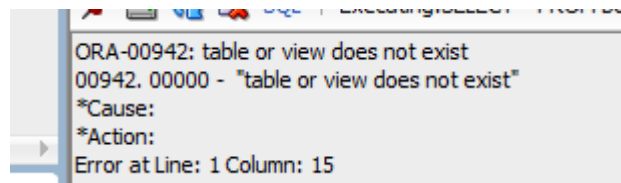


- 테이블 자체를 날려버리자

```
DROP TABLE board;
```

ports

Table BOARD dropped.



## DCL 데이터 제어 언어 Data Control Language

권한관리

bootstrap : 템플릿사이트

AdiminLTE

## ▼ View : 가상의 테이블

- DB에서 중요한 개념
- View 장점 :

- **보안성 제공** (ex. 보안 등급이 낮은 직원은 VIP 회원테이블 정보 접근에 일부 항목 제한)

→ 일부항목만 추출하여 뷰를 만들어 보안등급이 낮은 직원 해당 뷰만 볼 수 있도록 하면 보안성 장점을 확보 가능

○

## 일단 가져와보기

```
SELECT a.employee_id ,a.hire_date,
       b.department_name, b.job_title
FROM employees A, emp_details_view B;
```

	EMPLOYEE_ID	HIRE_DATE	DEPARTMENT_NAME	JOB_TITLE
1	100	17-JUN-03	Marketing	Marketing Representative
2	100	17-JUN-03	Marketing	Marketing Manager
3	100	17-JUN-03	Public Relations	Public Relations Represent.
4	100	17-JUN-03	Sales	Sales Representative
5	100	17-JUN-03	Sales	Sales Representative
6	100	17-JUN-03	Sales	Sales Representative
7	100	17-JUN-03	Sales	Sales Representative
8	100	17-JUN-03	Sales	Sales Representative

```
CREATE VIEW test_view AS
SELECT a.employee_id ,a.hire_date,
       b.department_name, b.job_title
FROM employees A, emp_details_view B;
```






Worksheet

Query Builder

```
CREATE VIEW test_view AS
SELECT a.employee_id, a.hire_date, b.department_name, b.job_title
FROM employees A, emp_details_view B;
```

Script Output x

Query Result x



Task completed in 0.053 seconds

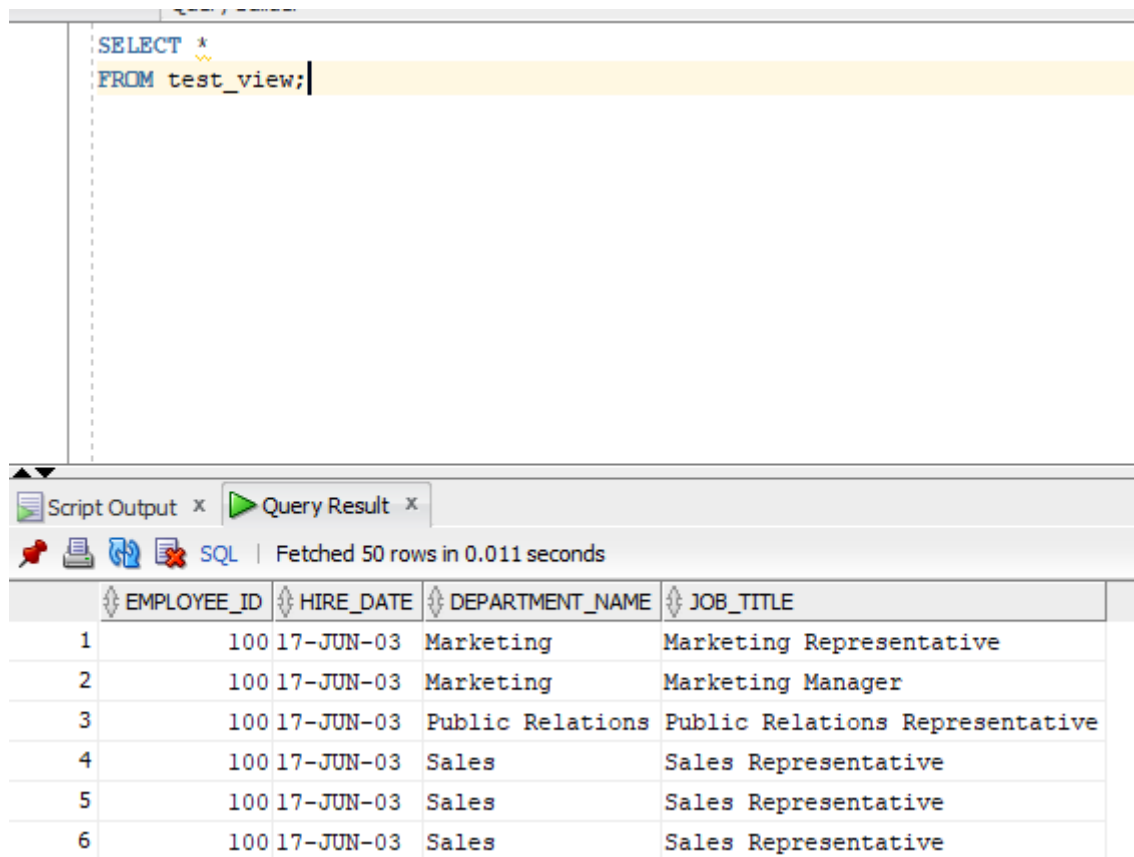
Table MEMBER altered.

Table BOARD truncated.

Table BOARD dropped.

View TEST\_VIEW created.

Act

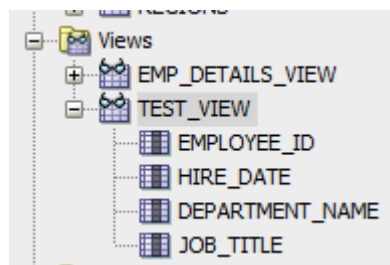


The screenshot shows a SQL IDE with a query editor at the top containing the following SQL statement:

```
SELECT *
FROM test_view;
```

Below the query editor, the 'Query Result' tab is active, displaying the results of the query. The status bar indicates 'SQL | Fetched 50 rows in 0.011 seconds'. The results are shown in a table with the following columns: EMPLOYEE\_ID, HIRE\_DATE, DEPARTMENT\_NAME, and JOB\_TITLE.

	EMPLOYEE_ID	HIRE_DATE	DEPARTMENT_NAME	JOB_TITLE
1	100	17-JUN-03	Marketing	Marketing Representative
2	100	17-JUN-03	Marketing	Marketing Manager
3	100	17-JUN-03	Public Relations	Public Relations Representative
4	100	17-JUN-03	Sales	Sales Representative
5	100	17-JUN-03	Sales	Sales Representative
6	100	17-JUN-03	Sales	Sales Representative



#### ▼ 궁금한거모음

Q. 자꾸 자바랑 헷갈려서 문장 끝에 ; 넣고 오류생긴다 환장

질문 1. 그냥 바카라랑 바카라2랑 무슨차이인지?

Q. varchar와 2의 차이는?

- 

VARCHAR 타입은 문자열 255 길이를 의미한다.



VARCHAR2 타입은 4000byte (한글 2000자, 영문 4000자) 까지 저장이 된다.

그래서 varchar와 varchar2의 차이점을 정확히 알아보면

varchar 는 ms-sql, MySQL(MariaDB),에서 사용하는 형식이고, varchar2는 Oracle 에서 사용하는 형식이다.

결국 문법상으로는 같은 형식이며, 사용하는 DBMS에 따라 사용하는 이름이 다른 것이다.

Q. 오라클 깔면 진짜 왜 이상한거임 컴이?

질문3 . 오라클은 이렇게 입력해놓고 지워놓고 한 파일 안에서 입력하고 지우고 썻쇼하는건지 원래?

Q. 그럼 새로운 똑같은 타입으로 바꾸면 어떻게 되는지?

Q.왜 대문자로 별명 지칭했는데 소문자로도 불러와짐?