SQL Class Day2

비교 연산자

.. = 같다

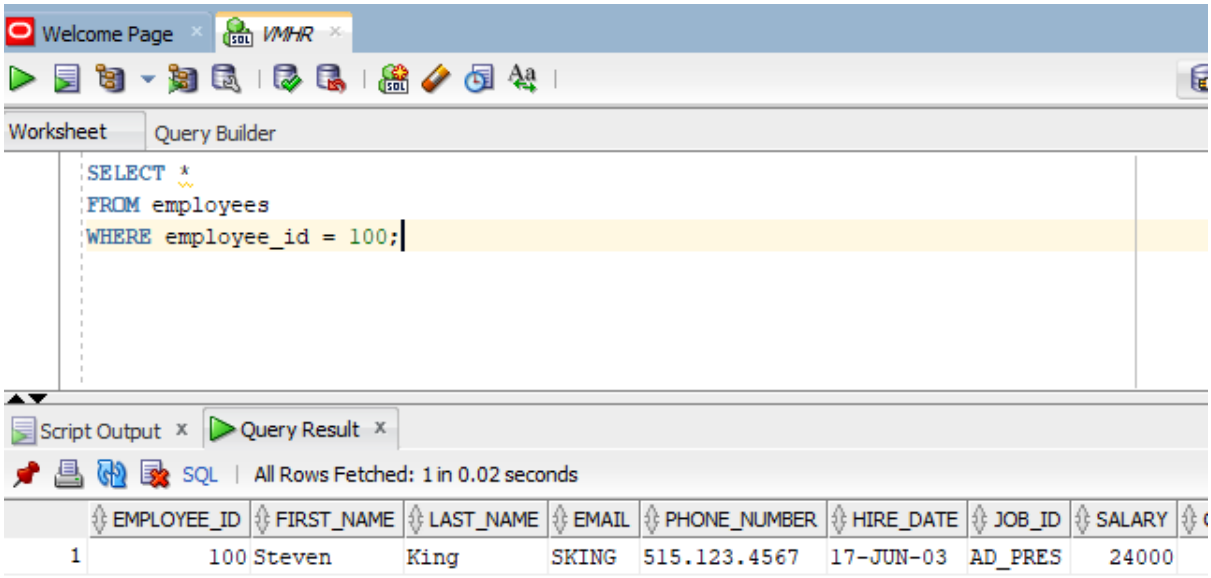
.. != 다르다 (<>)

.. > 크다

.. < 작다

.. >= 크거나 같다

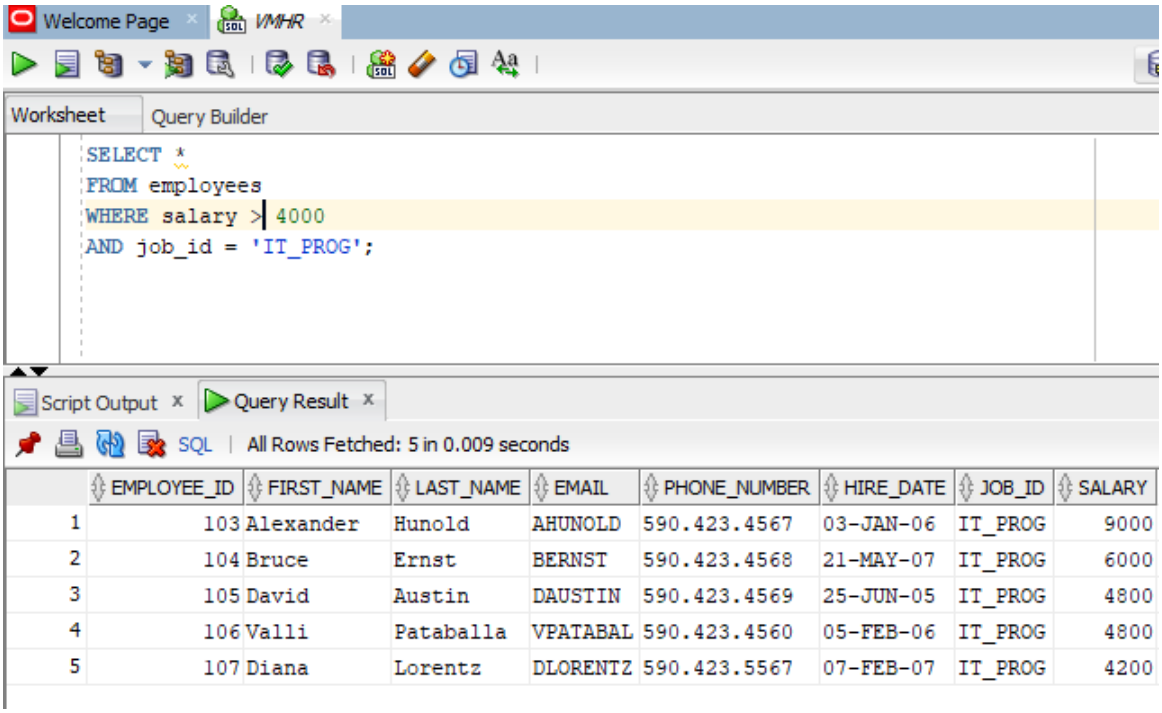
.. <= 작거나 같다



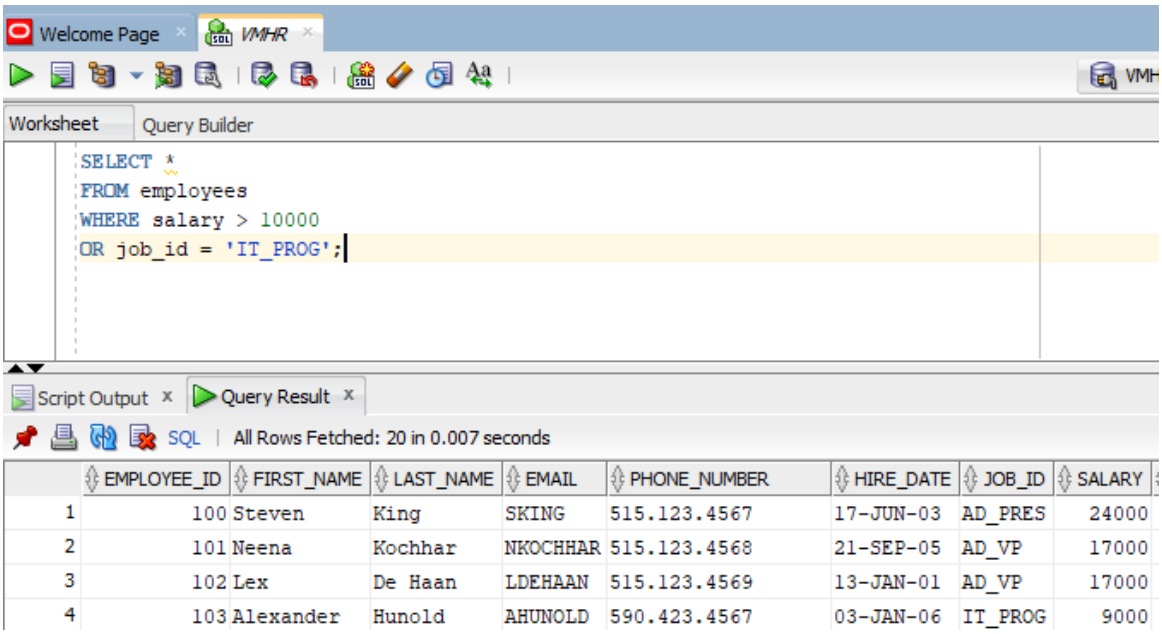
논리 연산자

- AND 모든 조건을 동시에 다만족할 때만 true
- OR 조건 중 하나만 만족해도 true
- NOT 조건의 반대 결과를 반환한다.

SELECT *
FROM 테이블
WHERE 조건1
AND 조건2;



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



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000

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

Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.005 seconds

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

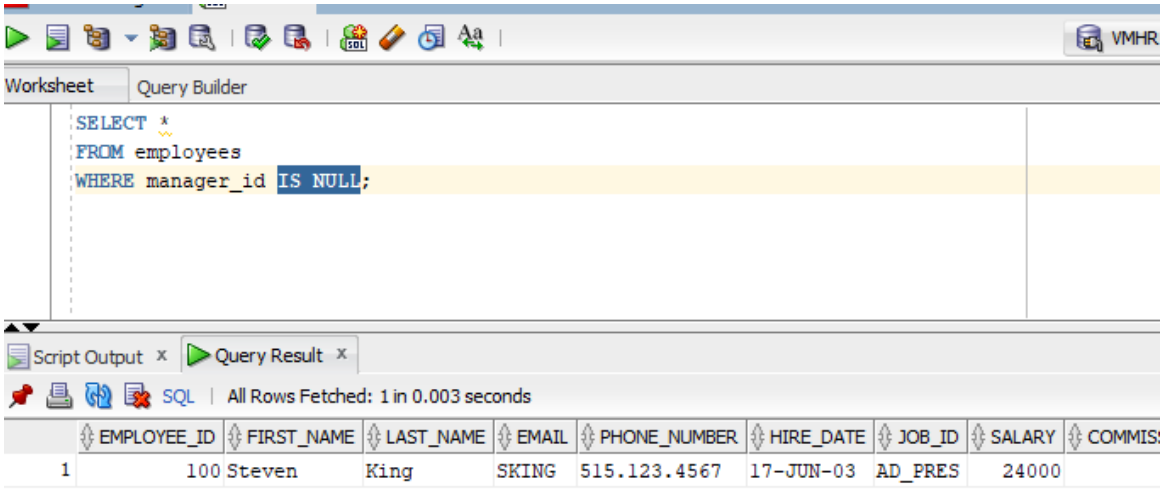
Worksheet | Query Builder

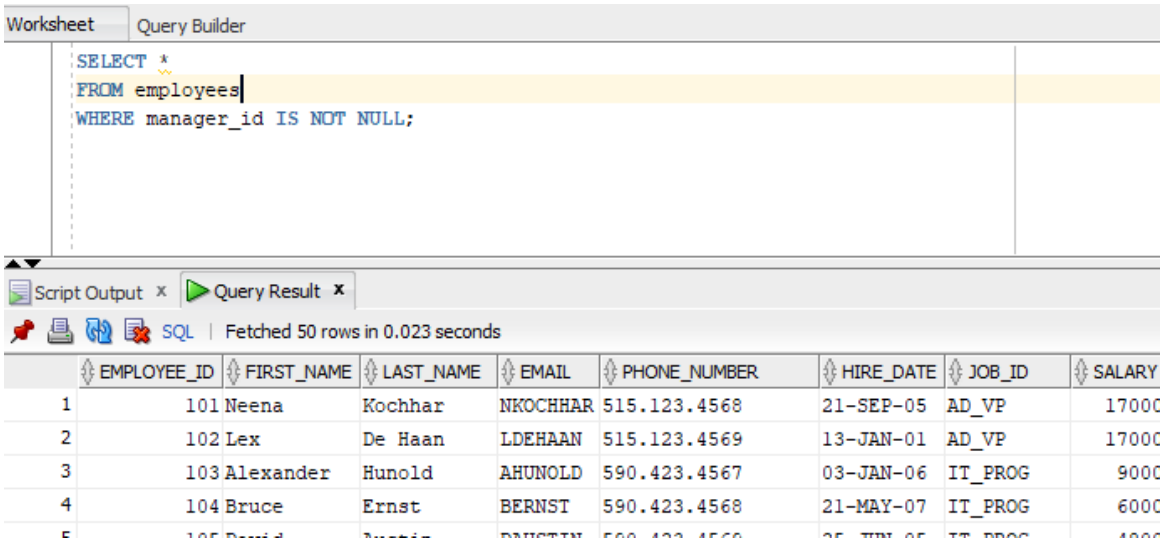
```
SELECT *
FROM employees
WHERE employee_id <> 105;
```

Script Output | Query Result

SQL | Fetched 50 rows in 0.005 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID
1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG
5	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-07	IT_PROG
6	106	Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06	IT_PROG





함수 사용

- 단일 행 함수 : 특정 행에 적용, 데이터 값을 하나씩 계산한다.

대문자/소문자/첫글자만대문자 (BOY/boy/Boy)

Worksheet Query Builder

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

Script Output x Query Result x

SQL | Fetched 50 rows in 0.026 seconds

	LAST_NAME	LOWER(LAST_NAME)	UPPER(LAST_NAME)	EMAIL	INITCAP(EMAIL)
1	Abel	abel	ABEL	EABEL	Eabel
2	Ande	ande	ANDE	SANDE	Sande
3	Atkinson	atkinson	ATKINSON	MATKINSO	Matkinso
4	Austin	austin	AUSTIN	DAUSTIN	Daustin
5	Baer	baer	BAER	HBAER	Hbaer

▼ 문자 관련 함수들

글자 자르기 substr

SUBSTR('원본글자', 시작위치, 자를 개수)

Worksheet Query Builder

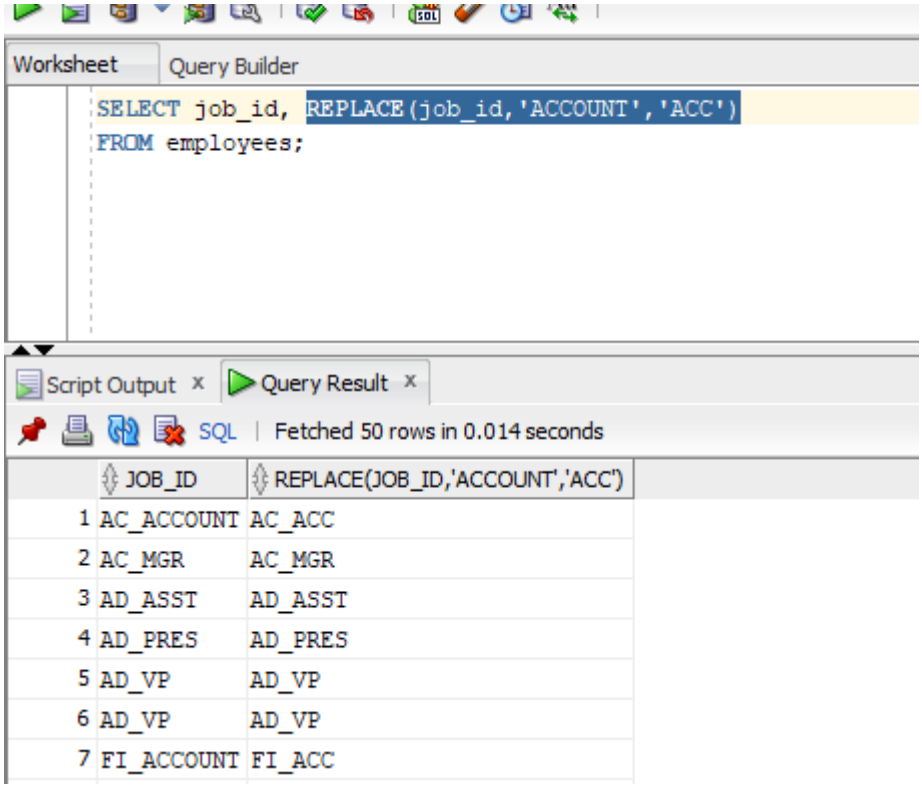
```
SELECT job_id, SUBSTR(job_id,1,2)
FROM employees;
```

Script Output x Query Result x

SQL | Fetched 50 rows in 0.005 seconds

	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

글자 바꾸기 - 특정 문자를 찾아서 변경 replace
REPLACE('문자열','찾을 문자','바꿀 문자')



LPAD, RPAD - 특정 문자로 자리 채우기

왼쪽부터 채우기 / 오른쪽 채우기

LPAD('문자열', 만들어질 자리수, '채울 문자')

Worksheet | Query Builder

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

Script Output | Query Result

SQL | Fetched 50 rows in 0.004 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

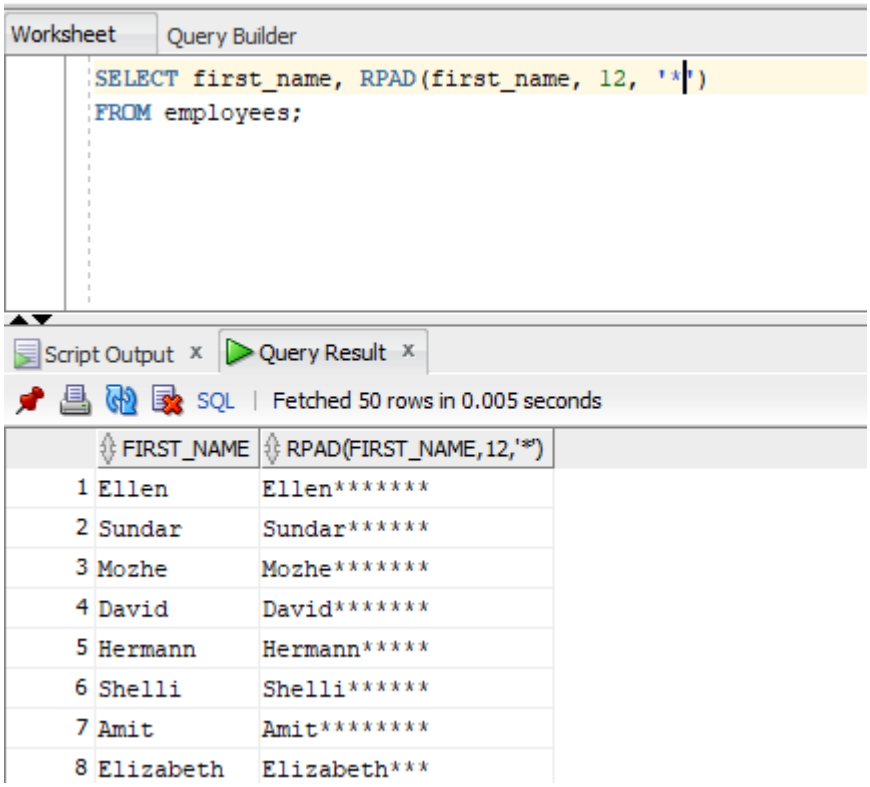
Worksheet | Query Builder

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

Script Output | Query Result

SQL | Fetched 50 rows in 0.01 seconds

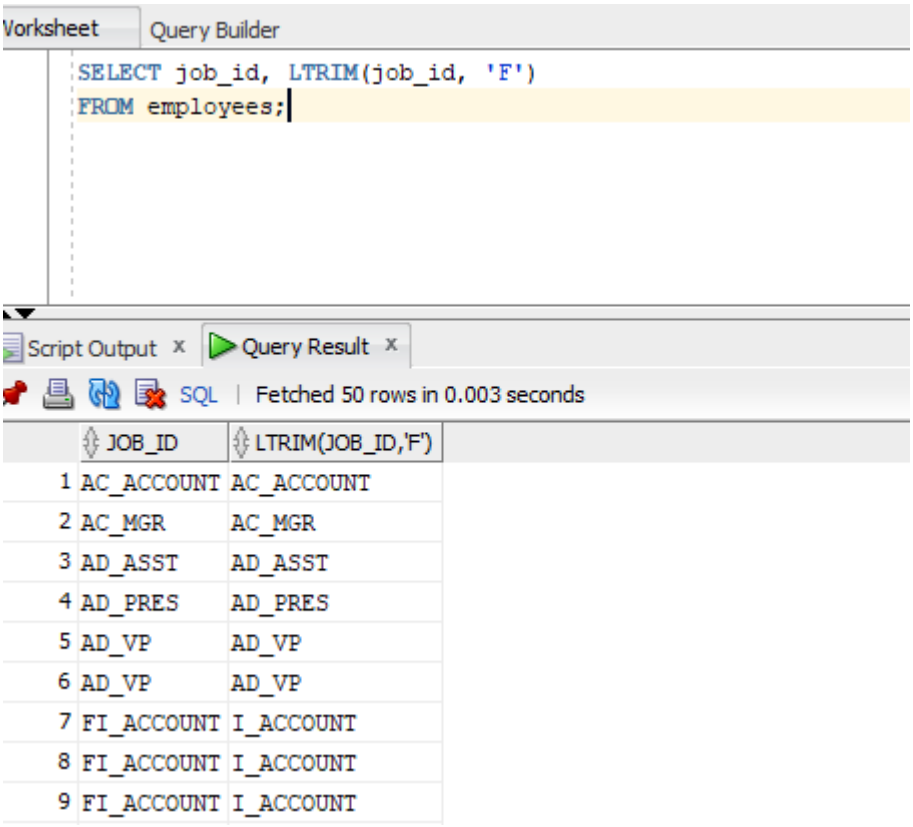
	FIRST_NAME	LPAD(FIRST_NAME,12,'')
1	Ellen	Ellen
2	Sundar	Sundar
3	Mozhe	Mozhe
4	David	David
5	Hermann	Hermann
6	Shelli	Shelli
7	Amit	Amit
8	Elizabeth	Elizabeth
9	Sarah	Sarah



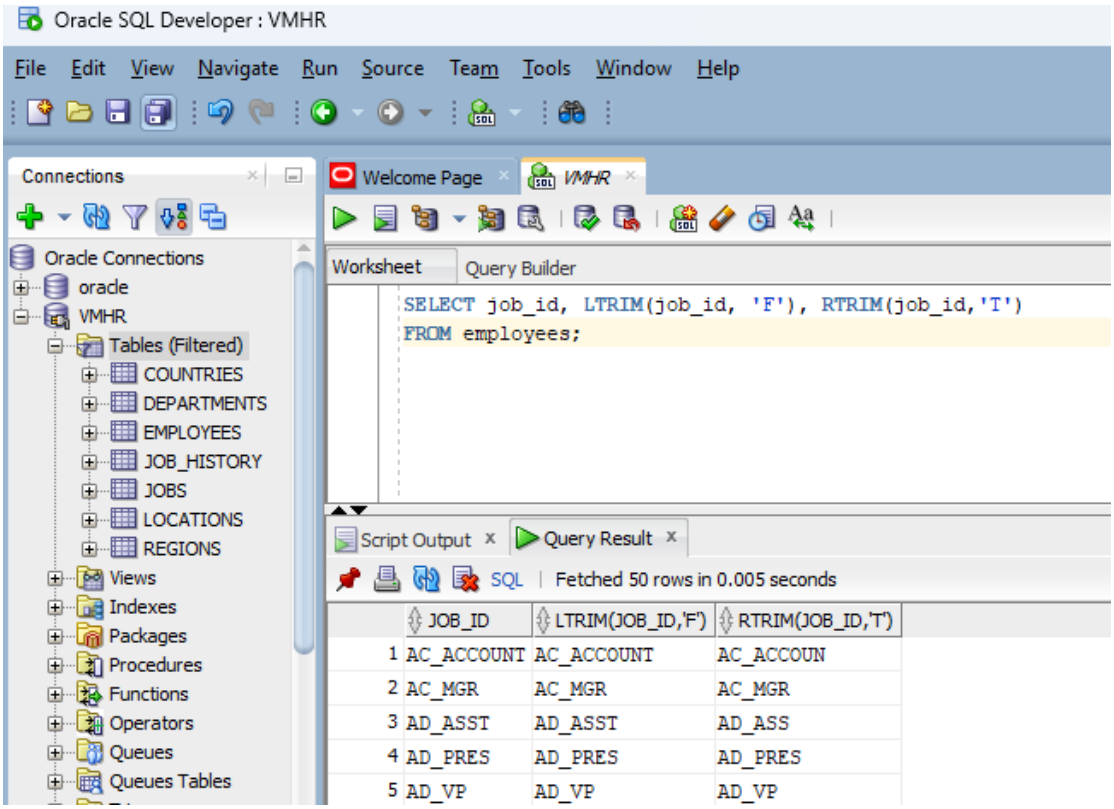
그룹 함수 : 그룹 전체 적용, 여러개 값을 그룹으로 계산한다.

LTRIM / RTRIM - 삭제하기

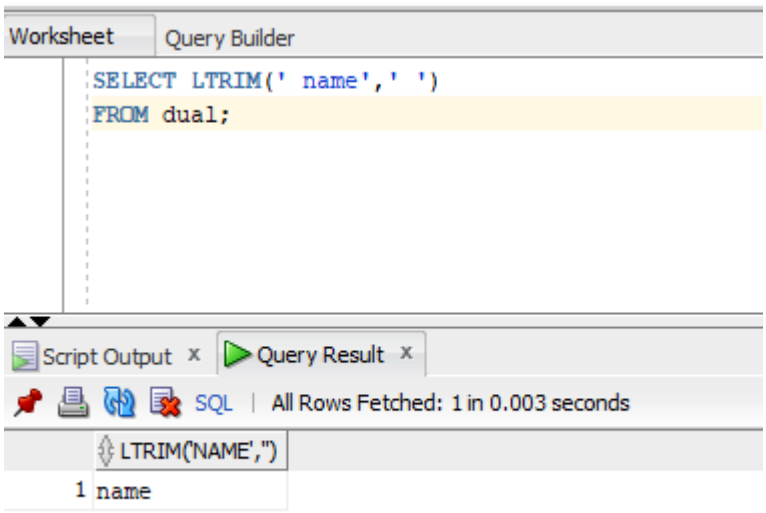
LTRIM('문자열'/열이름, '삭제할문자')



	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
8	FI_ACCOUNT	I_ACCOUNT
9	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

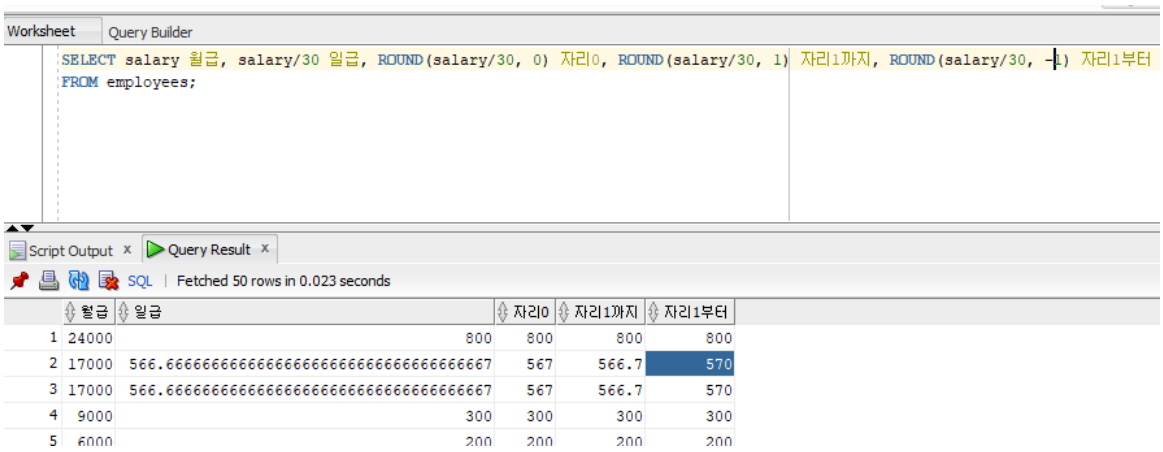


dual 테이블은 dummy 테이블 , 특정 테이블을 사용하지 않고 문법적으로 오류를 회피하고자 할때 사용하는 일종의 가상의 테이블로 생각하자.

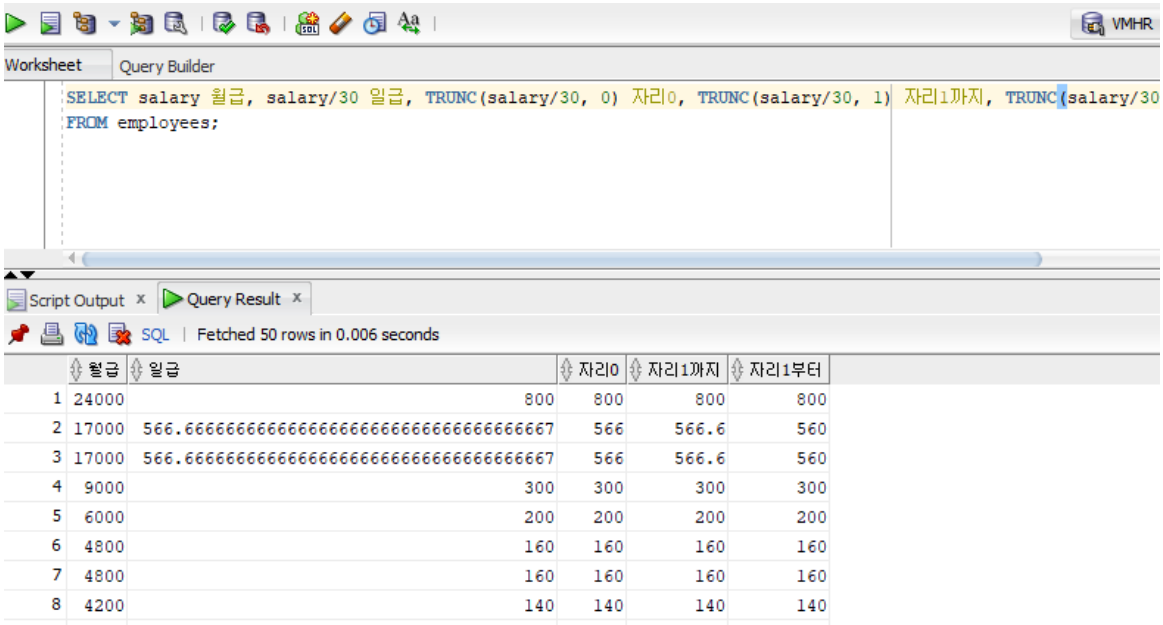
숫자 관련 함수들

round - 반올림

ROUND('열이름', 반올림할 위치)



trunc 버림 / 절삭

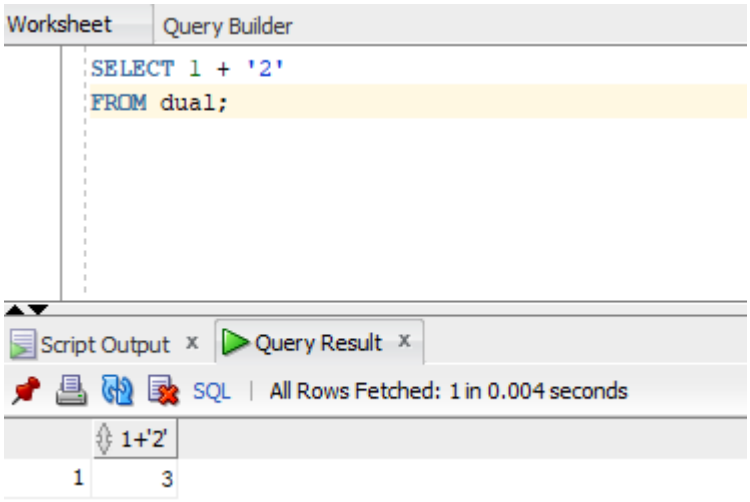


	월급	일급	자리0	자리1까지	자리1부터
1	24000	800	800	800	800
2	17000	566.6666666666666666666666666666666666667	566	566.6	560
3	17000	566.6666666666666666666666666666666666667	566	566.6	560
4	9000	300	300	300	300
5	6000	200	200	200	200
6	4800	160	160	160	160
7	4800	160	160	160	160
8	4200	140	140	140	140

TRUNC(열이름, 자리값)

데이터 형변환 casting

자동 형변환 (묵시적 형변환) - 필요시 데이터 형을 자동으로 변환한다.



	1+'2'
1	3

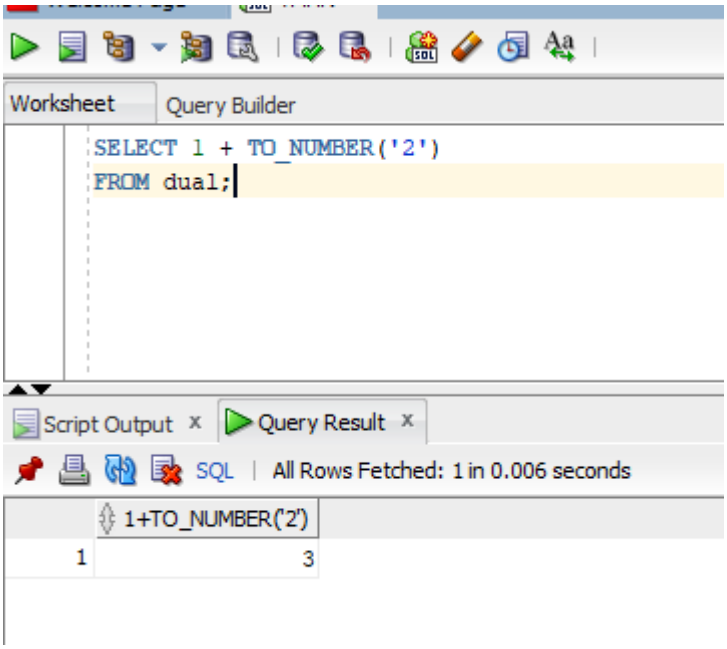
- 숫자 2의 경우는 작은 따옴표가 붙어 있어 숫자가 아닌 문자이다. 하지만 예외가 발생하지 않고 원하는 방향으로 계산이 이루어졌다. 시스템이 자동으로 알아서 숫자로

변환하여 계산하였기 때문이다.

- 이렇듯 자동으로 형변환이 이루어 질수는 있지만 항상 사용자가 원하는 의도대로 변환이 자동으로 완변하게 이루어지지는 않는다. 따라서 자동 형변환이 잘 되더라도 수동으로 의도적으로 명시 해주는 것이 바람직하다.

수동 형변환 (명시적 형변환)- 수동으로 데이터 형변환한다.

- TO_CHAR 문자로 형변환
- TO_NUMBER 숫자로 형변환
- TO_DATE 날짜로 형변환



VARCHAR2 (varchar) —→ NUMBER (integer)

NUMBER ——> VARCHAR2

NVL 함수

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

Script Output | Query Result

SQL | Fetched 50 rows in 0.005 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANA
31	160	Louise	Doran	LDORAN	011.44.1345.629268	15-DEC-05	SA_REP	7500	0.3	
32	156	Janette	King	JKING	011.44.1345.429268	30-JAN-04	SA_REP	10000	0.35	
33	157	Patrick	Sully	PSULLY	011.44.1345.929268	04-MAR-04	SA_REP	9500	0.35	
34	158	Allan	McEwen	AMCEWEN	011.44.1345.829268	01-AUG-04	SA_REP	9000	0.35	
35	145	John	Russell	JRUSSEL	011.44.1344.429268	01-OCT-04	SA_MAN	14000	0.4	
36	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	(null)	
37	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	(null)	
38	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	(null)	

salary 와 commission _pct 값을 계산하려고 하면 문제가 발생한다.

이유는 커미션을 지급받지 않는 직원 즉 null 이 있기에 문제가 생긴다.

null로 계산하면 결과는 null 이 된다.

null 값을 1로 치환하여 계산하면 전체 계산에 문제를 발생시키지 않게 된다.

NVL 함수는 null값을 특정 값으로 치`환하여 계산이 이루어지도록 처리한다.

Worksheet | Query Builder

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

Script Output | Query Result

SQL | Fetched 100 rows in 0.01 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)
45	(null)

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

Script Output | Query Result

SQL | All Rows Fetched: 107 in 0.007 secon

	SALARY*NVL(COMMISSION_PCT,1)
30	3300
31	2250
32	3500
33	3325
34	3150
35	5600
36	24000
37	17000
38	17000
39	9000
40	6000
41	4800
42	4800
43	4200
44	12008

DECODE - 조건 처리하기

DECODE(열이름, 조건값, 치환값, 기본값)

치환값 - 조건을 만족할 경우

기본값 - 조건을 만족하지 않을 경

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

Script Output | Query Result

SQL | Fetched 50 rows in 0.007 seconds

	DEPARTMENT_ID	EMPLOYEE_ID	FIRST_NAME	원래급여	DECODE(DEPARTMENT_ID,60,SALARY*1.1,SALARY)
1	90	100	Steven	24000	24000
2	90	101	Neena	17000	17000
3	90	102	Lex	17000	17000
4	60	103	Alexander	9000	9900
5	60	104	Bruce	6000	6600
6	60	105	David	4800	5280
7	60	106	Valli	4800	5280
8	60	107	Diana	4200	4620
9	100	108	Nancy	12008	12008
10	100	109	Daniel	9000	9000
11	100	110	John	8200	8200

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

Script Output | Query Result | SQL | Fetched 50 rows in 0.005 seconds

	DEPARTMENT_ID	EMPLOYEE_ID	FIRST_NAME	원래급여	DECODE(DEPARTMENT_ID,60,SALARY*1.1,SALARY)	DECODE(
1	90	100	Steven	24000	24000	
2	90	101	Neena	17000	17000	
3	90	102	Lex	17000	17000	
4	60	103	Alexander	9000	9900	급여인상
5	60	104	Bruce	6000	6600	급여인상
6	60	105	David	4800	5280	급여인상
7	60	106	Valli	4800	5280	급여인상
8	60	107	Diana	4200	4620	급여인상
9	100	108	Nancy	12008	12008	
10	100	109	Daniel	9000	9000	

CASE - 경우의 수가 여러개일 경우 즉 복잡한 조건 처리

Worksheet | Query Builder

```
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';
```

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0.007 seconds

	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 numbering

RANK 공통 순위 만큼 건너 뛰어 순위 매기기 1,2,2,4

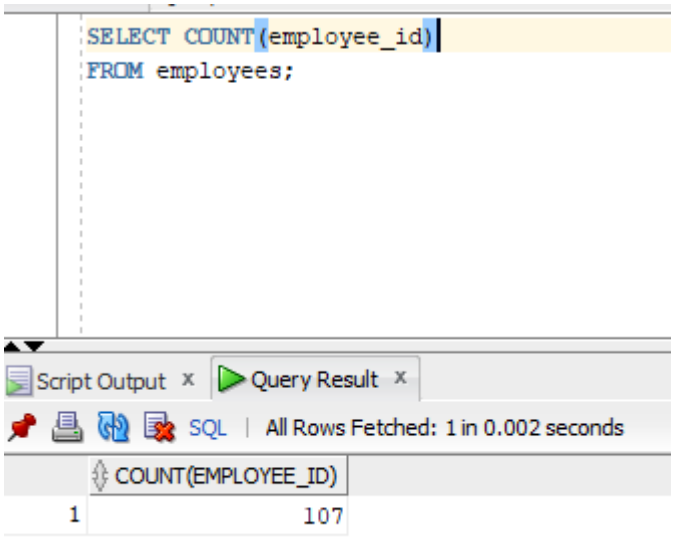
DENSE_RANK 공통 순위를 건너 뛰지 않고 순위 1,2,2,3

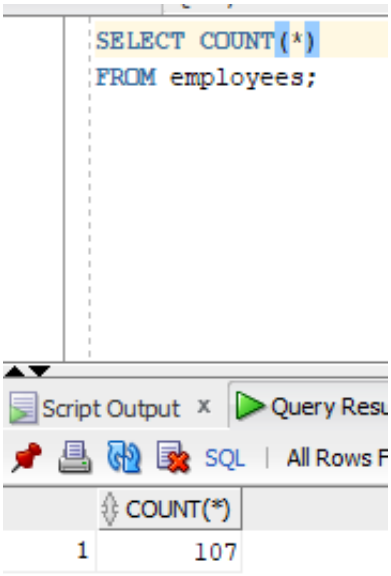
ROW_NUMBER 공통 순위 없이 출력 1.2,3,4

그룹 함수

여러행에 함수가 적용이되어 하나의 결과를 나타낸다.

count 갯수 (null 값도 포함하여 계산)





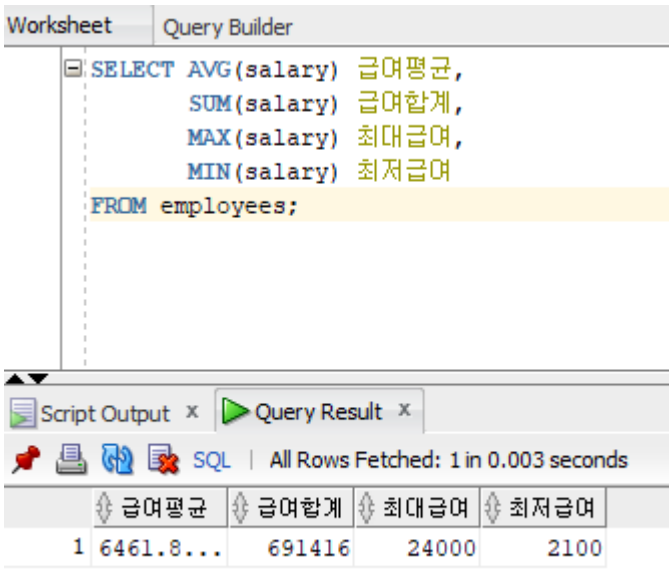
count 는 특성상 null 도 계산하기 때문에 어떠한 열로도 동일한 결과값이 나오기 때문에 * 를 주로 사용한다.

sum 합계 (null 값 제외하고 계산)

avg 평균 (null 값 제외하고 계산)

max 최대값 (null 값 제외하고 계산)

min 최소값 (null 값 제외하고 계산) ex)아스키코드(문자에도 최대,최소가 있다)



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

Script Output | Query Result

SQL | All Rows Fetched: 1 in 0.004 seconds

	최대값	최저값
1	Winston	Adam

SQL 워크시트(W) 내역

워크시트 질의 작성기

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

질의 결과

SQL | 인출된 모든 행: 1(0.008초)

	최대값	최저값
1	Winston	Adam

Group By 그룹으로 묶기

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

Script Output | Query Result

SQL | Executing:SELECT job_id, A

```
ORA-00937: not a single-group group function
00937. 00000 - "not a single-group group function"
*Cause:
*Action:
Error at Line: 1 Column: 8
```

Worksheet | Query Builder

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

Script Output | Query Result

SQL | All Rows Fetched: 19 in 0.007 seconds

	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

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

Script Output | Query Result

SQL | All Rows Fetched: 19 in 0.005 seconds

	JOB_ID	AVG(SALARY)	SUM(SALARY)	COUNT(*)
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

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

Script Output | Query...

SQL | All Rows Fetched: 19 in 0.02 seconds

	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

DML, 데이터 조작 언어 Data Manipulation Language

SELECT 조회 read

INSERT 삽입 create

UPDATE 수정 update

DELETE 삭제 delete

DDL, 데이터 정의 언어 Data Definition Language

DB 생성, Table 생성 , 삭제 drop

DCL, 데이터 제어 언어 Data Control Language

권한관리

새로운 테이블 만들기

```
CREATE TABLE 테이블이름
    (열이름2   속성,
     열이름2   속성
     ...
     ...);
```

```
CREATE TABLE board
    (bno      number,
     btitle   varchar2(50),
     bwriter  varchar2(10),
     bcontent varchar2(500)
     );
```

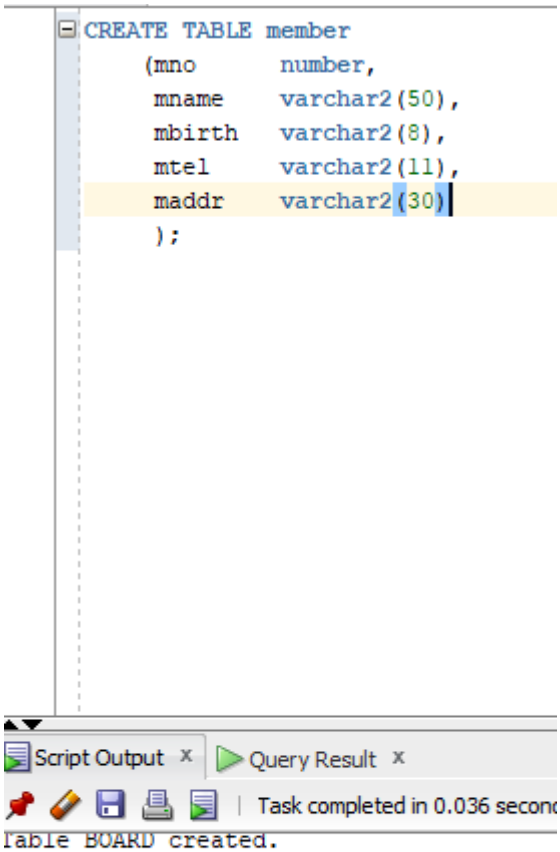
Script Output | Query Result | Task completed in 0.05 se

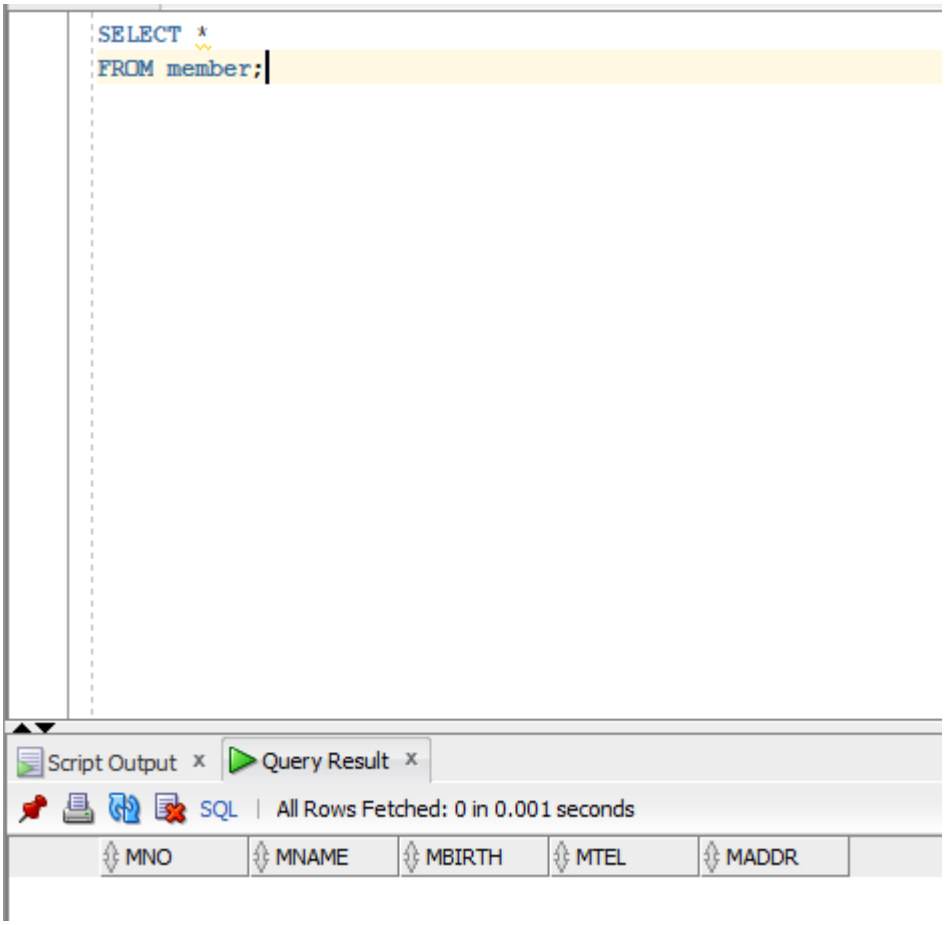
```
Table BOARD created.
```

```
SELECT *
FROM board;
```

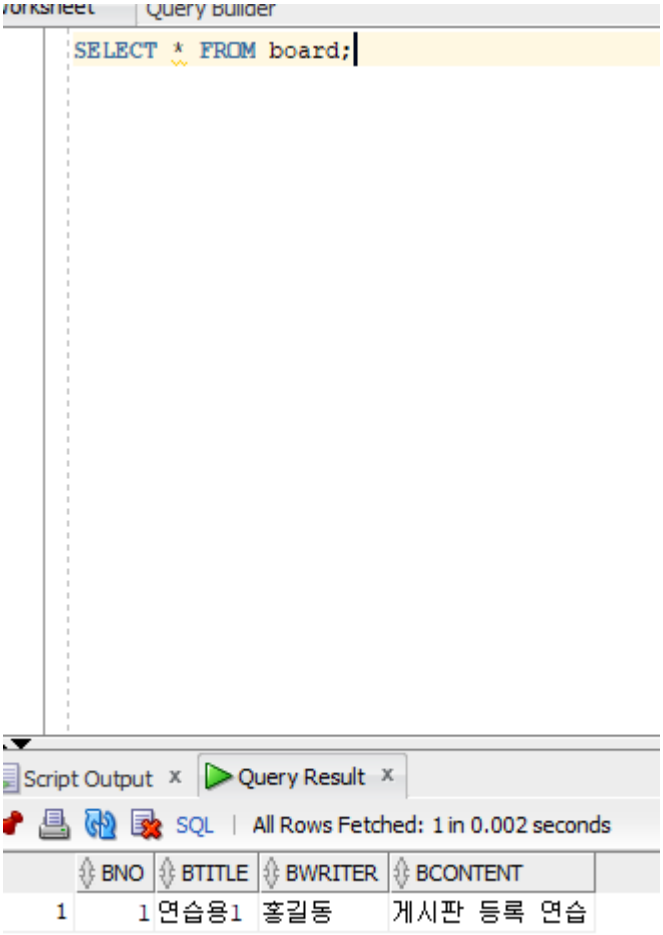
Script Output | Query Result | SQL | All Rows Fetched: 0 in 0.001 seconds

BNO	BTITLE	BWRITER	BCONTENT


CREATE TABLE member
(mno number,
mname varchar2(50),
mbirth varchar2(8),
mtel varchar2(11),
maddr varchar2(30)
);
Script Output
Query Result
Task completed in 0.036 second
Table BOARD created.
Table MEMBER created.



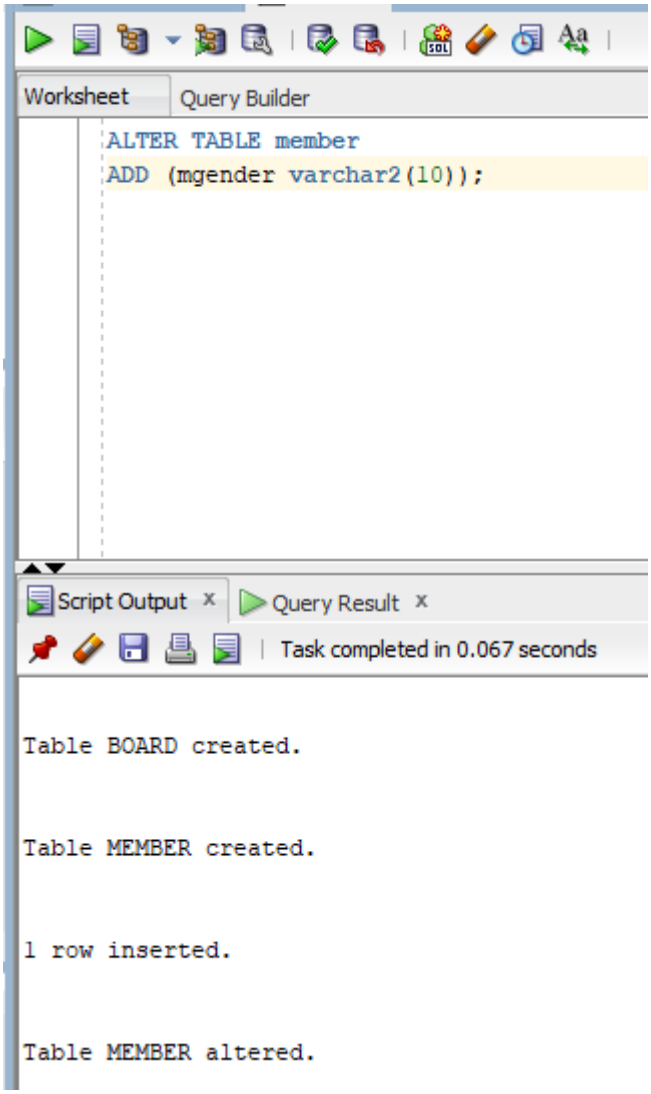

```
INSERT INTO board
VALUES(1, '연습용1','홍길동','게시판 등록 연습');
```



이름 정의 방법

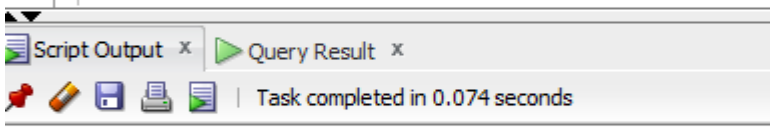
- 동일한 이름의 테이블이 여러 개 존재할 수 없다 .
- 예약어 즉 이미 사용 중인 명령어 등으로는 이름을 사용할수 없다.
- 반드시 문자로 시작해야 한다. 한글/특수문자도 쓸수는 있지만 절대 사용하지 말자
- 가능하면 의미있는 단어를 사용하

테이블 수정 (항목 추가)


Worksheet
Query Builder
ALTER TABLE member
ADD (mgender varchar2(10));
Script Output
Query Result
Task completed in 0.067 seconds
Table BOARD created.
Table MEMBER created.
1 row inserted.
Table MEMBER altered.

테이블 수정 (항목 수)

```
ALTER TABLE member MODIFY (MTEL varchar2(20));
```




```
Table MEMBER created.

1 row inserted.

Table MEMBER altered.

Table MEMBER altered.
```

```
ALTER TABLE member RENAME COLUMN MTEL to MPHONE;
```

Script Output | Query Result

Task completed in 0.04 seconds

```
1 row inserted.

Table MEMBER altered.

Table MEMBER altered.

Table MEMBER altered.
```



```
SELECT * FROM member;
```

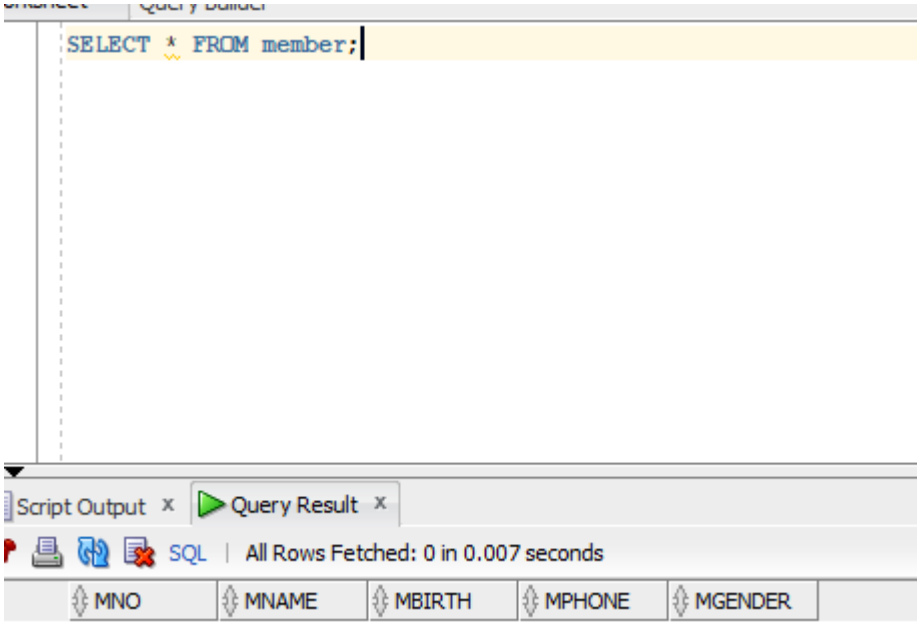
Script Output | Query Result

SQL | All Rows Fetched: 0 in 0.018 seconds

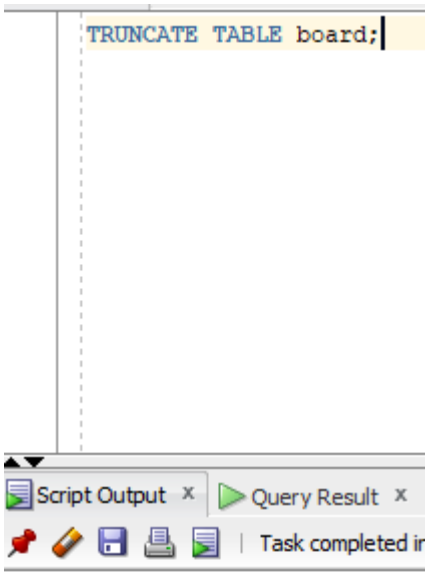
MNO	MNAME	MBIRTH	MPHONE	MADDR	MGENDER

heet | Query Builder

```
ALTER TABLE member DROP COLUMN MADDR ;
```



테이블에서 전체 데이터 삭제 (구조는 그대로)

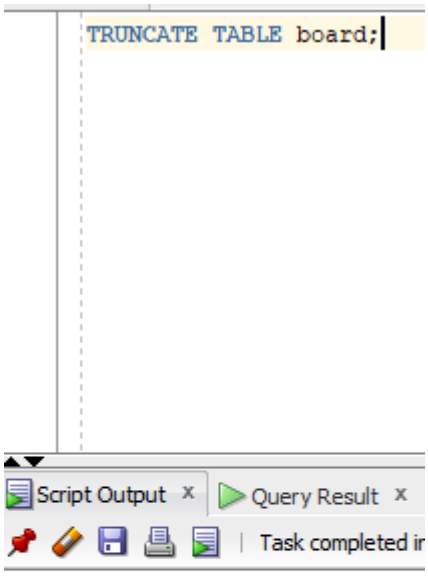



```
Table MEMBER altered.

Table MEMBER altered.

Table MEMBER altered.

Table BOARD truncated.
```

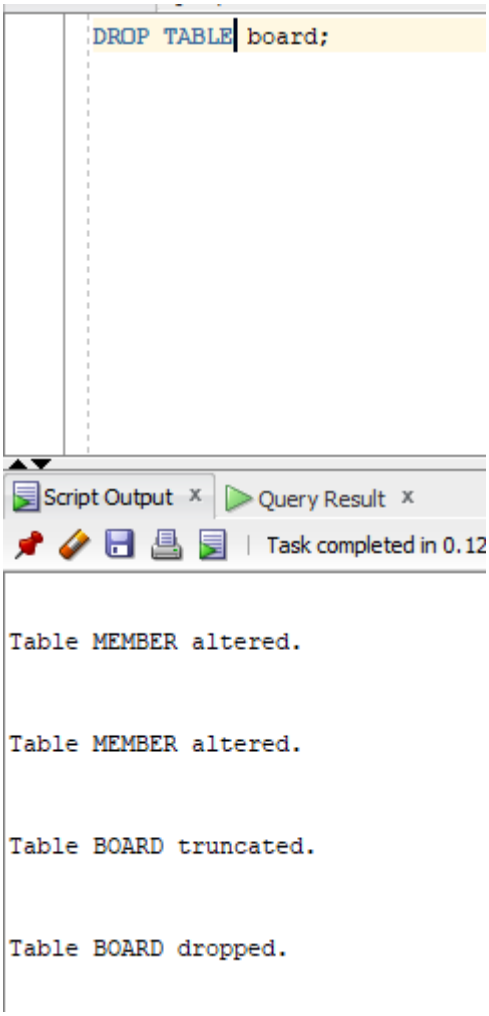


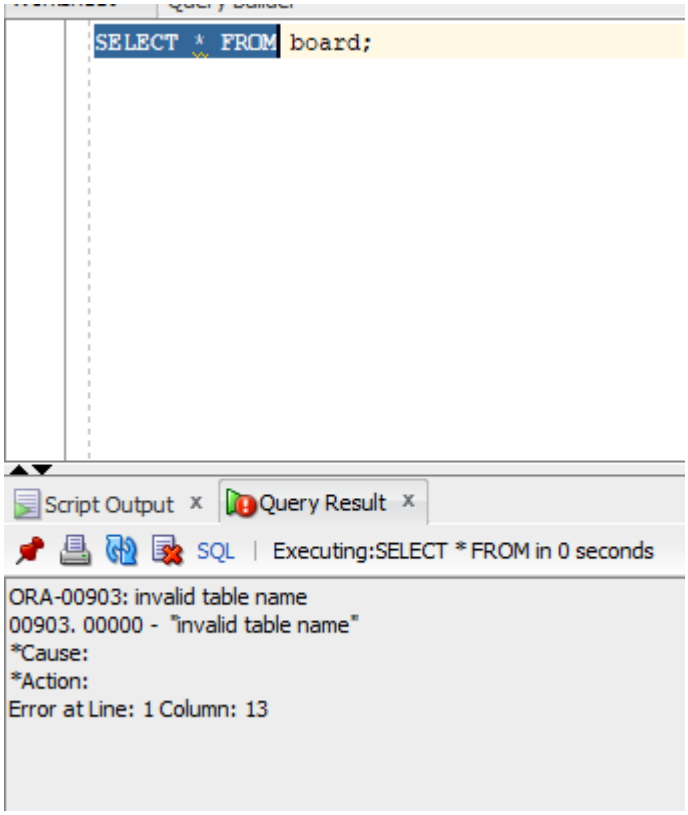

```
Table MEMBER altered.


Table MEMBER altered.


Table MEMBER altered.


Table BOARD truncated.
```


DROP TABLE board;
Script Output
Query Result
Task completed in 0.12
Table MEMBER altered.
Table MEMBER altered.
Table BOARD truncated.
Table BOARD dropped.



DML delete 데이터만 삭제

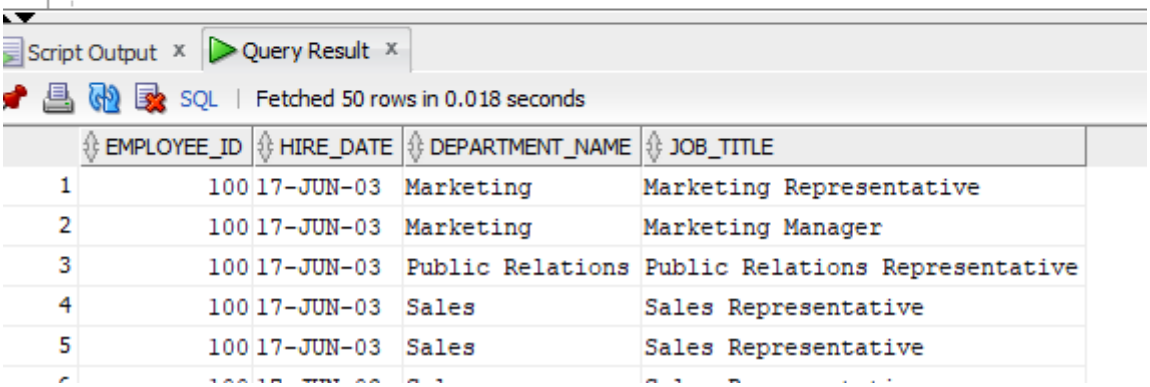
DDL truncate 구조를 남기고 데이터만 전체 삭제

DDL drop 구조 포함 데이터 전체 완전히 삭제

View - 가상의 테이블

View는 데이터베이스에서 가상의 테이블이다. 실제로 테이블에 저장되어 있는 데이터를 그대로 사용하는 것이 아니라, 필요한 데이터만 추출하여 새로운 가상 테이블을 만들어서 사용한다. 뷰는 테이블과 동일하게 사용자에 의해 생성되고, SQL문으로 조작이 가능하지만, 데이터는 뷰가 참조한 원본 테이블에 저장되어 있다. 뷰는 데이터를 중복해서 저장하지 않아도 되므로 데이터 정규화에 도움을 준다. 또한, 특정 사용자가 필요한 데이터만 조회하도록 제한하거나, 여러 개의 테이블에서 데이터를 조합하여 표시할 수 있다.

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



Script Output | Query Result

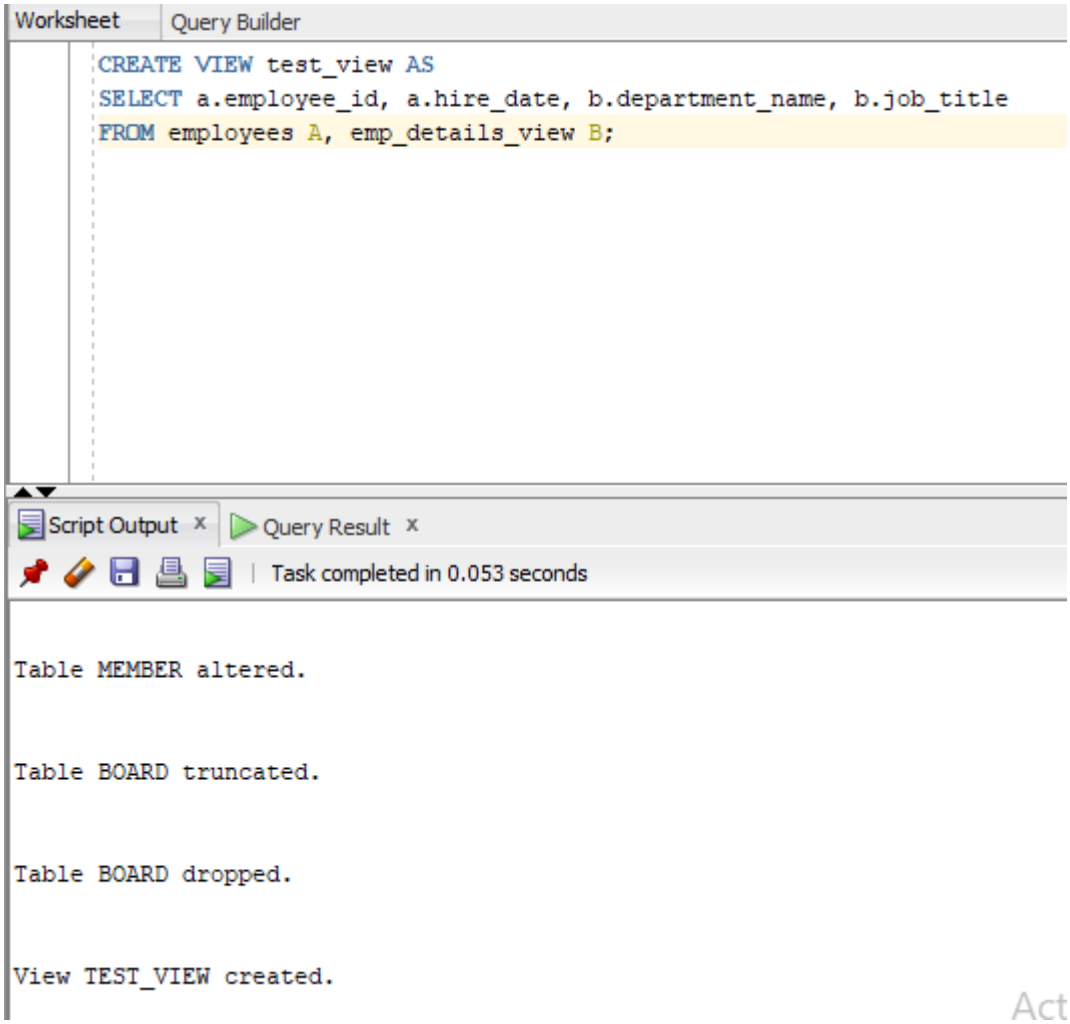
SQL | Fetched 50 rows in 0.018 seconds

	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

```
CREATE VIEW employee_view AS
SELECT employee_id, first_name, last_name, salary, department_name
FROM employees
JOIN departments ON employees.department_id = departments.department_id;


뷰의 장점
- 뷰는 보안을 제공한다. 예를 들면, 보안 등급이 낮은 직원은 VIP 회원 테이블의 보든 정보를 다 볼수 없도록 일부 항목만을 볼수 있게 일부 항목들만 추출하여  뷰를 만들어 보안 등급이 낮은 직원 해당 뷰만 볼수 있도록 하면  보안상 잇점을 확보할수 있다.
```

뷰는 데이터베이스에서 가상의 테이블이기 때문에, 보안상의 이점을 제공합니다. 예를 들어, 보안 등급이 낮은 직원은 VIP 회원 테이블의 모든 정보를 볼 수 없도록, 일부 항목들만 추출하여 새로운 뷰를 만들고 해당 뷰만 볼 수 있도록 하면, 보안성이 보장됩니다. 이를 통해, 뷰를 통해서만 필요한 정보에 접근하도록 설정하여, 데이터베이스 전체의 보안성을 높일 수 있습니다.


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
Query Result
Task completed in 0.053 seconds
Table MEMBER altered.
Table BOARD truncated.
Table BOARD dropped.
View TEST_VIEW created.

```
SELECT *
FROM test_view;
```

Script Output | Query Result

SQL | Fetched 50 rows in 0.011 seconds

	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