SQL class 2

▼ 비교연산자

= 같다

!= 다르다(<>)

> 크다

< 작다

>= 크거나 같다

<= 작거나 같다

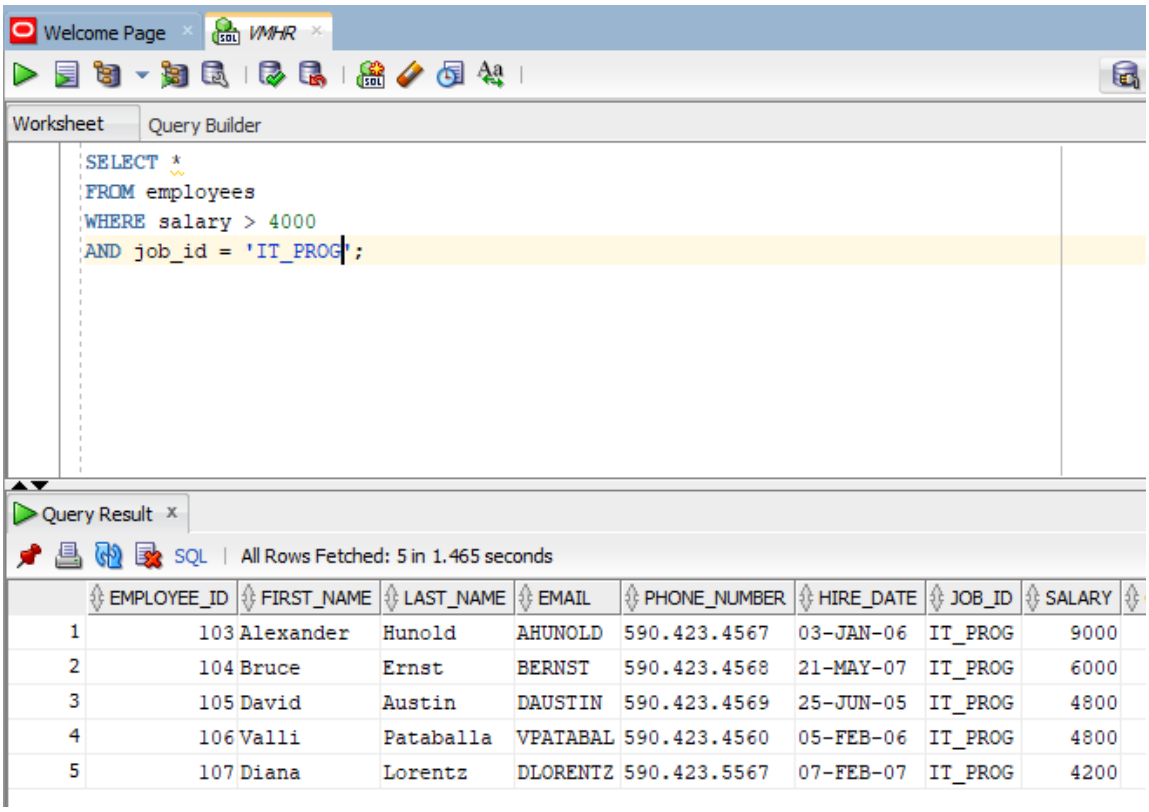
▼ 논리연산자

- AND 모든 조건을 동시에 다 만족할 때만 TRUE
- OR 조건 중 하나만 만족해도 TRUE
- NOT 조건의 반대 결과를 반환

AND

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

▼ 실행

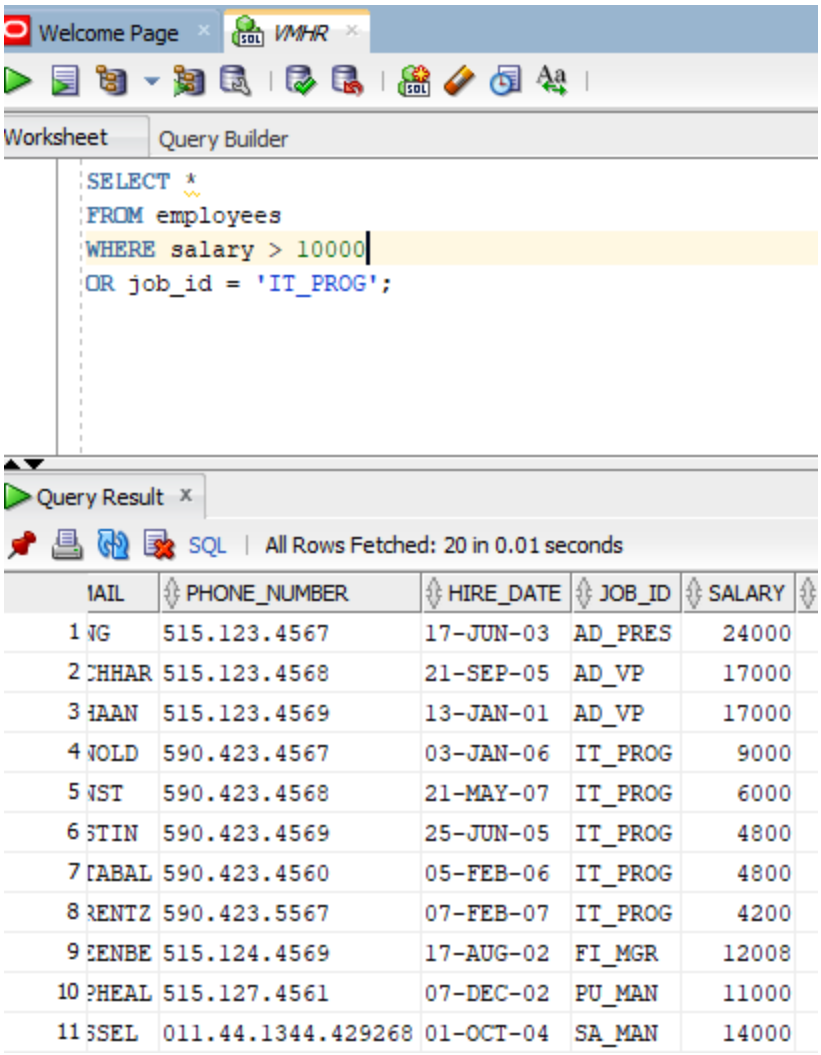


둘 다 만족해야 나

OR

SELECT *
FROM 테이블
WHERE 조건1
OR 조건2

▼ 실행



둘 중 하나만 만족해도 나옴

AND와 OR를 섞어서 사용도 가능하다.

▼ 실행

Worksheet | Query Builder

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

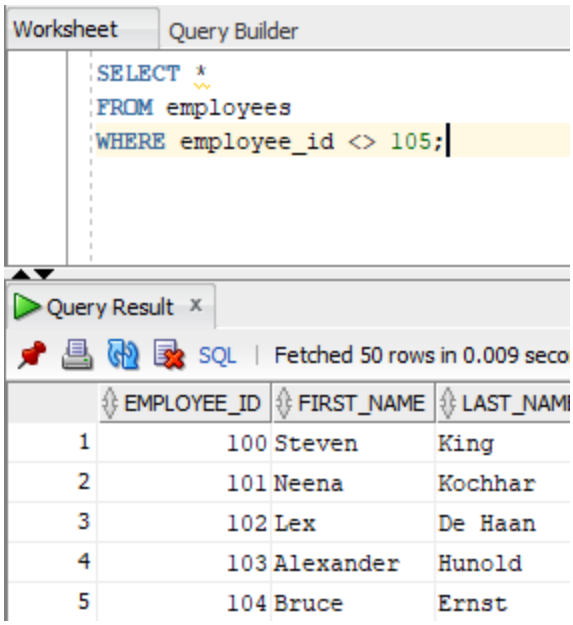
Query Result

SQL | All Rows Fetched: 10 in 0.165 seconds

	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
1	UNOLD	590.423.4567	03-JAN-06	IT_PROG	9000
2	RNST	590.423.4568	21-MAY-07	IT_PROG	6000
3	USTIN	590.423.4569	25-JUN-05	IT_PROG	4800
4	ATABAL	590.423.4560	05-FEB-06	IT_PROG	4800
5	ORENTZ	590.423.5567	07-FEB-07	IT_PROG	4200
6	AVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000
7	HEN	515.124.4269	28-SEP-05	FI_ACCOUNT	8200
8	CIARRA	515.124.4369	30-SEP-05	FI_ACCOUNT	7700
9	URMAN	515.124.4469	07-MAR-06	FI_ACCOUNT	7800
10	OPP	515.124.4567	07-DEC-07	FI_ACCOUNT	6900

NOT

▼ 실행



105가 제외되었다.

▼ 문자 관련 함수들

- 단일 행 함수 : 특정 행에 적용 , 데이터 값을 하나씩 계산

대문자 / 소문자 / 첫 글자만 대문자

SUN / sun / Sun

LOWER ⇒ 전부 소문자

UPPER ⇒ 전부 대문자

INITCAP ⇒ 첫 글자만 대문자

▼ 실행

Worksheet | Query Builder

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

Query Result

SQL | Fetched 50 rows in 0.006 seconds

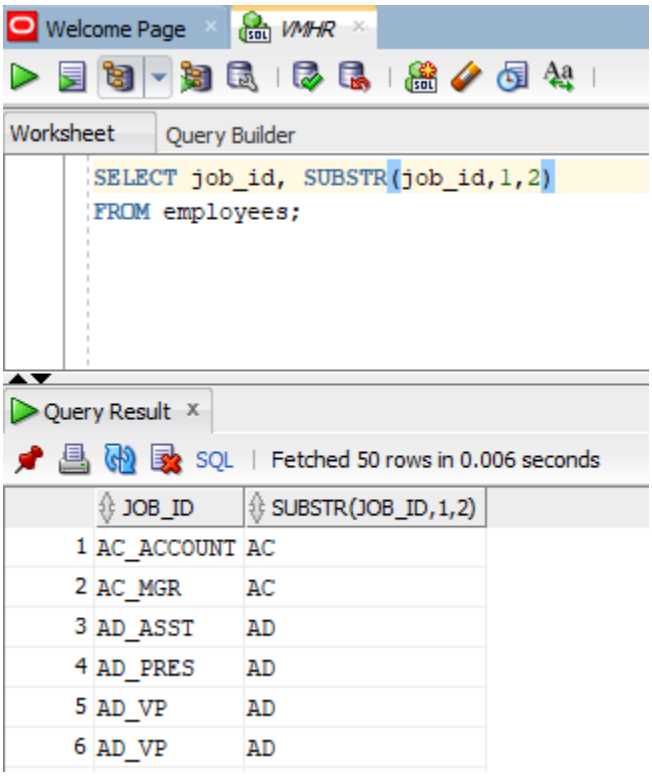
	LAST_NAME	LOWER(LAST_NAME)	UPPER(LAST_NAME)	EMAIL	INITCAP(EMAIL)
1	Abel	abel	ABEL	EABEL	Eabel
2	Ande	ande	ANDE	SANDE	Sande
3	Atkinson	atkinson	ATKINSON	MATKINSO	Matkinso
4	Austin	austin	AUSTIN	DAUSTIN	Daustin
5	Baer	baer	BAER	HBAER	Hbaer
6	Baida	baida	BAIDA	SBAIDA	Sbaida
7	Banda	banda	BANDA	ABANDA	Abanda
8	Bates	bates	BATES	EBATES	Ebates
9	Bell	bell	BELL	SBELL	Sbell
10	Bernstein	bernstein	BERNSTEIN	DBERNSTE	Dbernste

SUBSTR 글자 자르기

SUBSTR('원본글자', 시작위치, 자를 개수)

*1부터 시작한다

▼ 실행

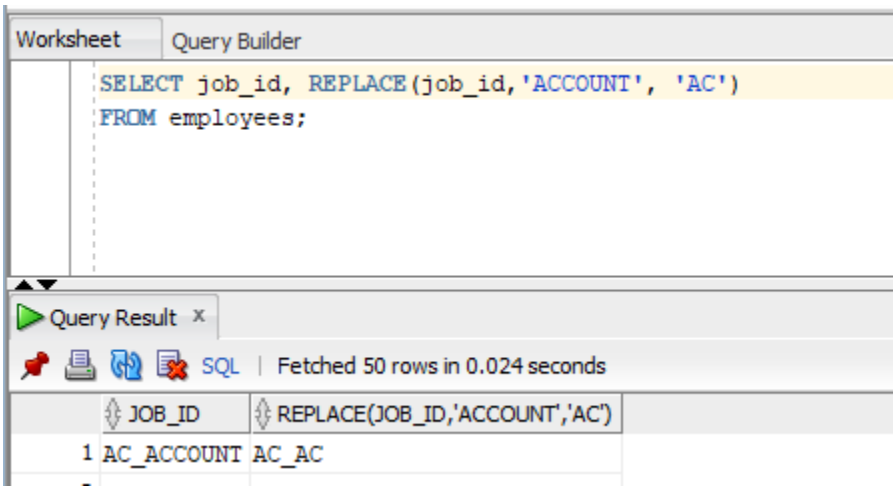


JOB_ID에서 앞에 두 글자만 잘라서 가져왔다

REPLACE 글자 바꾸기 - 특정 문자를 찾아서 변경

REPLACE('문자열', '찾을 문자', '바꿀 문자')

▼ 실행



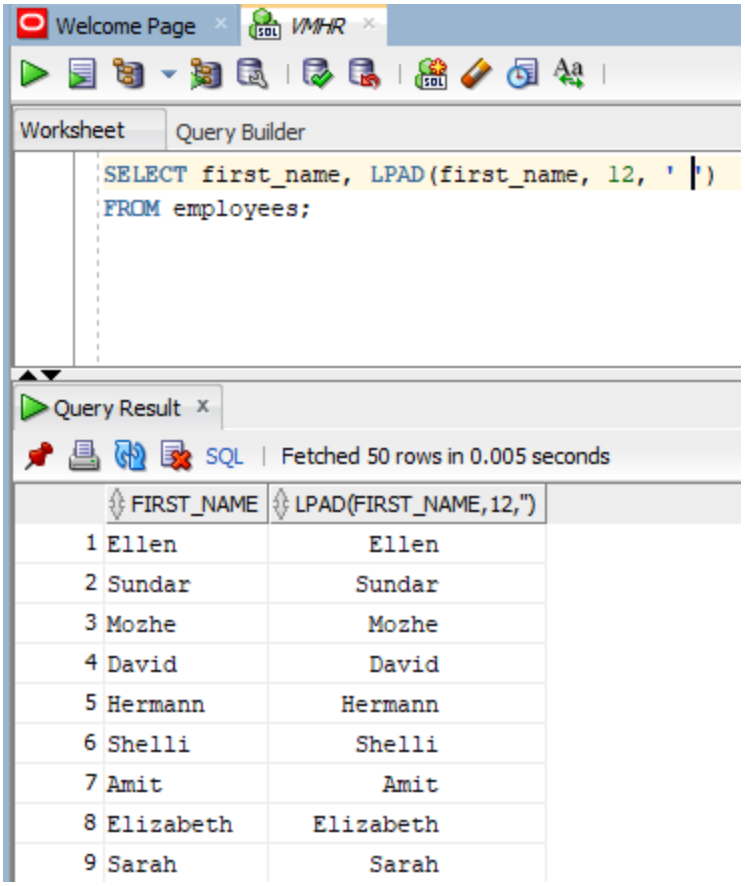
LPAD, RPAD - 특정 문자로 자리 채우기

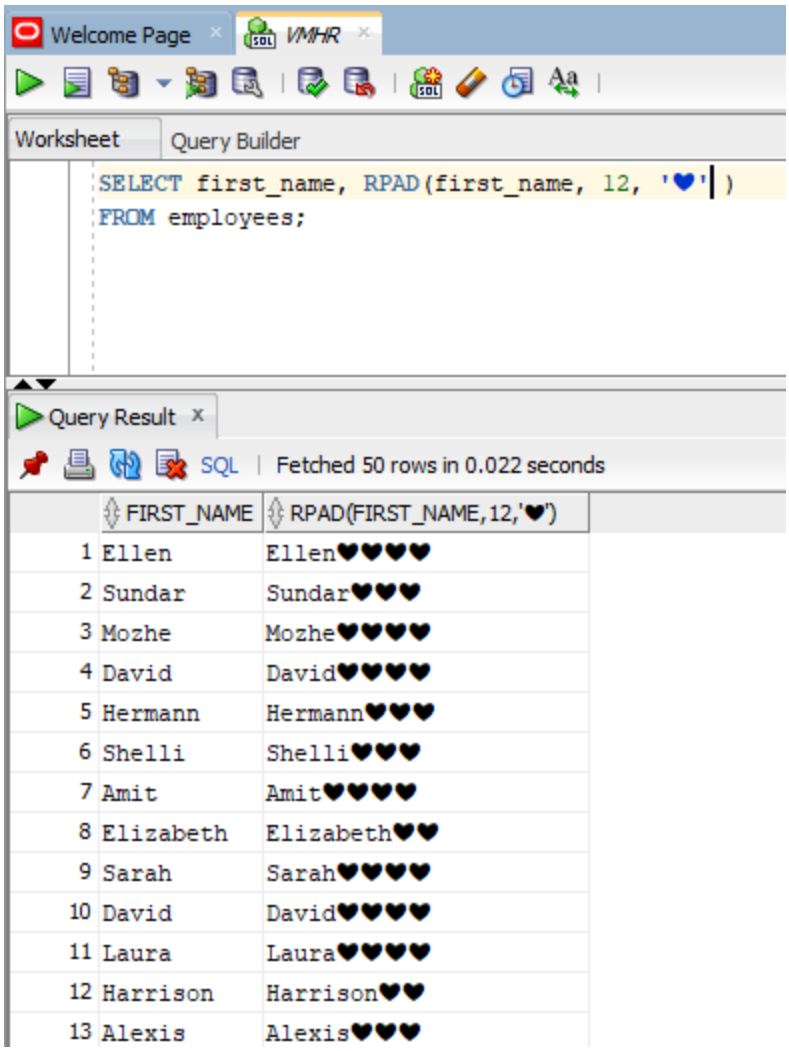
LPAD → 왼쪽 채우기

RPAD → 오른쪽 채우기

LPAD('문자열', 만들어질 자릿수, '채울문자')

▼ 실행





LTRIM, RTRIM - 삭제하기

LTRIM('문자열'/열이름, '자를문자')

▼ 실행

Welcome Page | VMHR

Worksheet | Query Builder

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

Query Result

SQL | Fetched 50 rows in 0.013 seconds

	JOB_ID	LTRIM(JOB_ID,'F')
7	FI_ACCOUNT	I_ACCOUNT
8	FI_ACCOUNT	I_ACCOUNT
9	FI_ACCOUNT	I_ACCOUNT
10	FI_ACCOUNT	I_ACCOUNT
11	FI_ACCOUNT	I_ACCOUNT

Worksheet | Query Builder

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

Query Result

SQL | Fetched 50 rows in 0.002 second

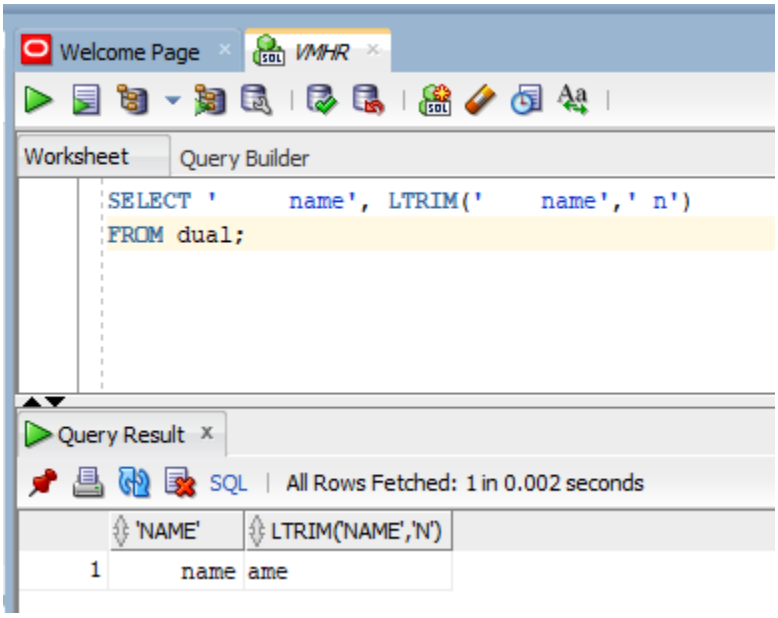
	JOB_ID	RTRIM(JOB_ID,'T')
1	AC_ACCOUNT	AC_ACCOUN
2	AC_MGR	AC_MGR
3	AD_ASST	AD_ASS
4	AD_PRES	AD_PRES
5	AD_VP	AD_VP
6	AD_VP	AD_VP
7	FI_ACCOUNT	FI_ACCOUN
8	FI_ACCOUNT	FI_ACCOUN
9	FI_ACCOUNT	FI_ACCOUN
10	FI_ACCOUNT	FI_ACCOUN
11	FI_ACCOUNT	FI_ACCOUN

dual table (dummy table)

특정 테이블을 사용하지 않고 문법적으로 오류를 회피하고자 할 때 사용하는

일종의 가상 테이블

▼ 실행



▼ 숫자 관련 함수들

ROUND - 반올림

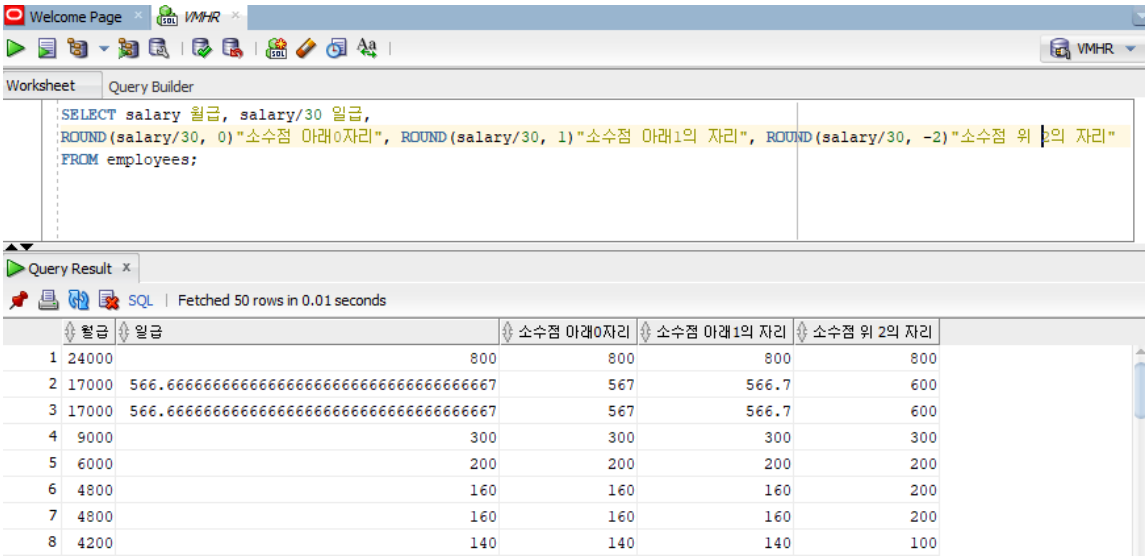
ROUND('열이름', 반올림할 위치)

위치는 소수점을 기준으로 정해진다

0, 1, 2, 3 은 소수점 아래 자리를 지칭하고

-1, -2, -3은 소수점 위 자리를 지칭

▼ 실행

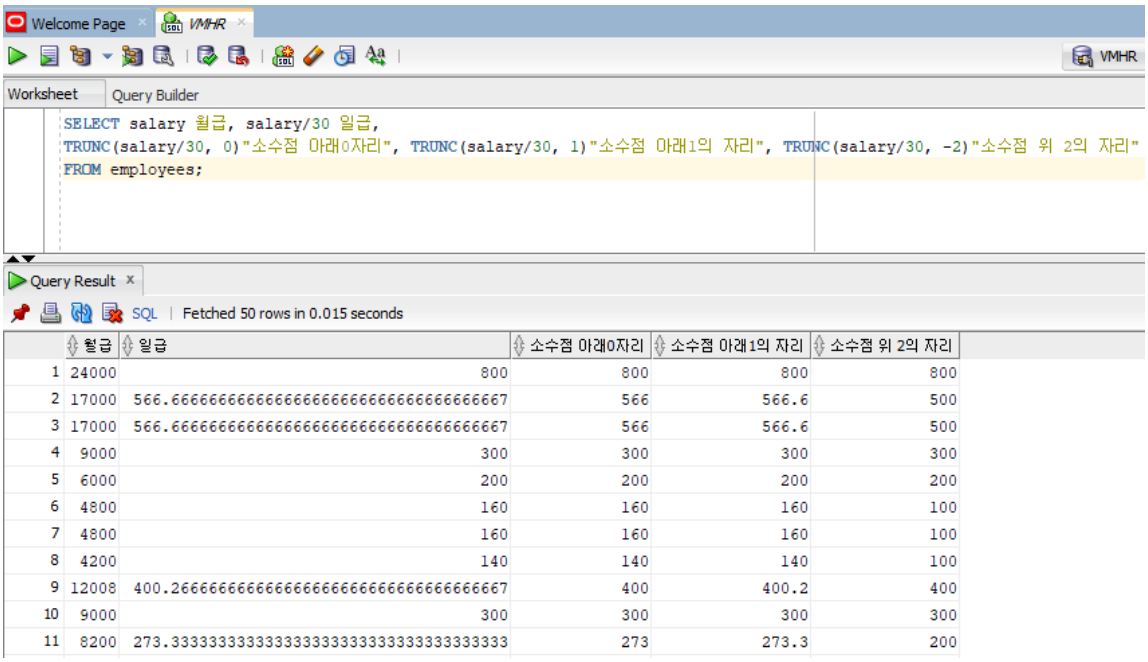


	월급	일급	소수점 아래0자리	소수점 아래1의 자리	소수점 위 2의 자리
1	24000	800	800	800	800
2	17000	566.6666666666666666666666666666666667	567	566.7	600
3	17000	566.6666666666666666666666666666666667	567	566.7	600
4	9000	300	300	300	300
5	6000	200	200	200	200
6	4800	160	160	160	200
7	4800	160	160	160	200
8	4200	140	140	140	100

TRUNC 버림/절삭

TRUNC('열이름', 위치)

▼ 실행



	월급	일급	소수점 아래0자리	소수점 아래1의 자리	소수점 위 2의 자리
1	24000	800	800	800	800
2	17000	566.6666666666666666666666666666666667	566	566.6	500
3	17000	566.6666666666666666666666666666666667	566	566.6	500
4	9000	300	300	300	300
5	6000	200	200	200	200
6	4800	160	160	160	100
7	4800	160	160	160	100
8	4200	140	140	140	100
9	12008	400.2666666666666666666666666666666667	400	400.2	400
10	9000	300	300	300	300
11	8200	273.3333333333333333333333333333333333	273	273.3	200

▼ Casting 데이터 형변환

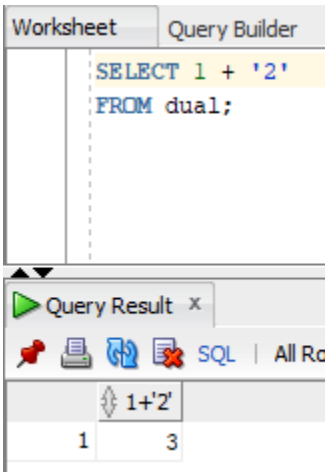
- 자동 형변환[묵시적 형변환

필요시 데이터 형을 자동으로 변환한다.

숫자 + 문자열 일지라도 '숫자'가 들어있을 때 시스템이 자동으로 형변환을 통해 정상 계산이 됨

하지만 항상 의도대로 완벽하게 이뤄지지 않기 때문에 의도적으로 명시해서 수동 형변환을 해주는 것이 바람직함

▼ 실행



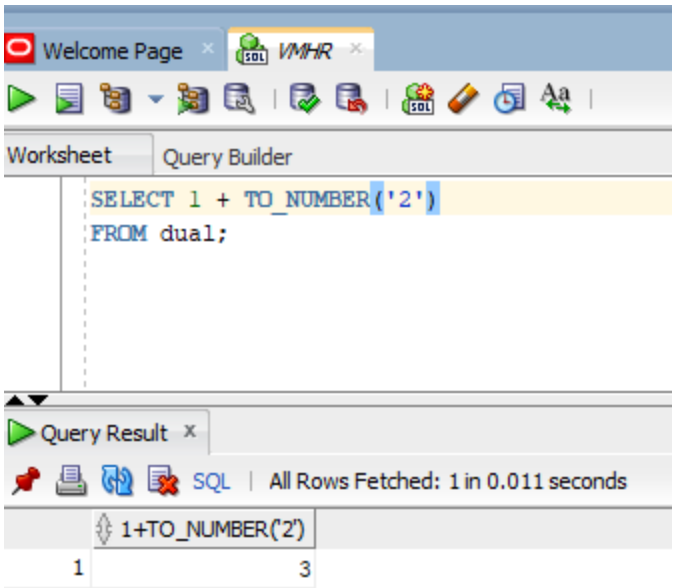
- 수동 형변환[명시적 형변환]

수동으로 데이터 형변환한다.

TO_CHAR 문자로 형변환

TO_NUMBER 숫자로 형변환

▼ 실행

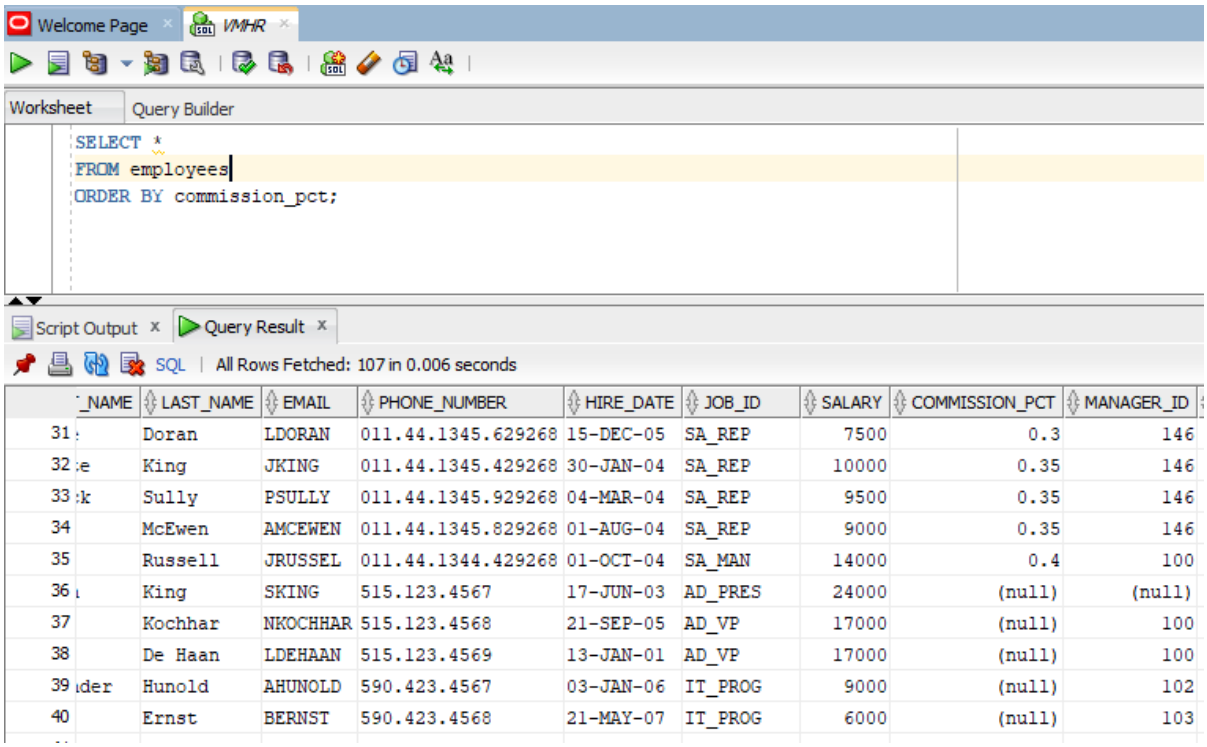


TO_DATE 날짜로 형변환

VARCHAR2 (varchar) —→ NUMBER(integer)

NUMBER(integer) —→ VARCHAR2(varchar)

▼ NVL - NULL값 처리



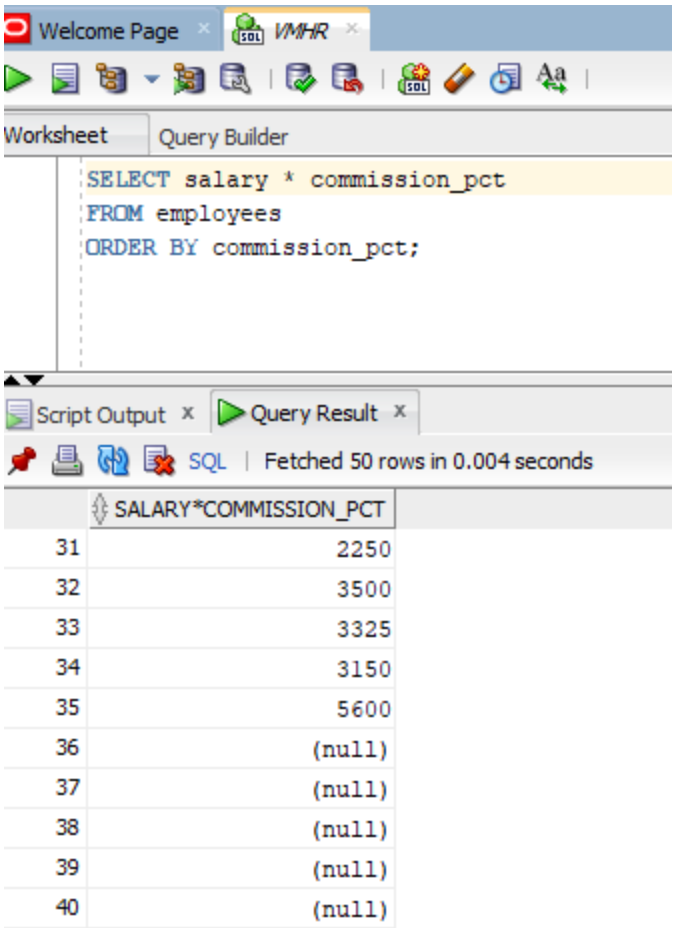
	_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID
31	:	Doran	LDORAN	011.44.1345.629268	15-DEC-05	SA_REP	7500	0.3	146
32	:e	King	JKING	011.44.1345.429268	30-JAN-04	SA_REP	10000	0.35	146
33	:k	Sully	PSULLY	011.44.1345.929268	04-MAR-04	SA_REP	9500	0.35	146
34		McEwen	AMCEWEN	011.44.1345.829268	01-AUG-04	SA_REP	9000	0.35	146
35		Russell	JRUSSEL	011.44.1344.429268	01-OCT-04	SA_MAN	14000	0.4	100
36	ı	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	(null)	(null)
37		Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	(null)	100
38		De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	(null)	100
39	ıder	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000	(null)	102
40		Ernst	BERNST	590.423.4568	21-MAY-07	IT_PROG	6000	(null)	103

salary 와 commission_pct 값을 계산하려고 하면

cummission_pct에 **null값이 있기 때문에 오류발생**

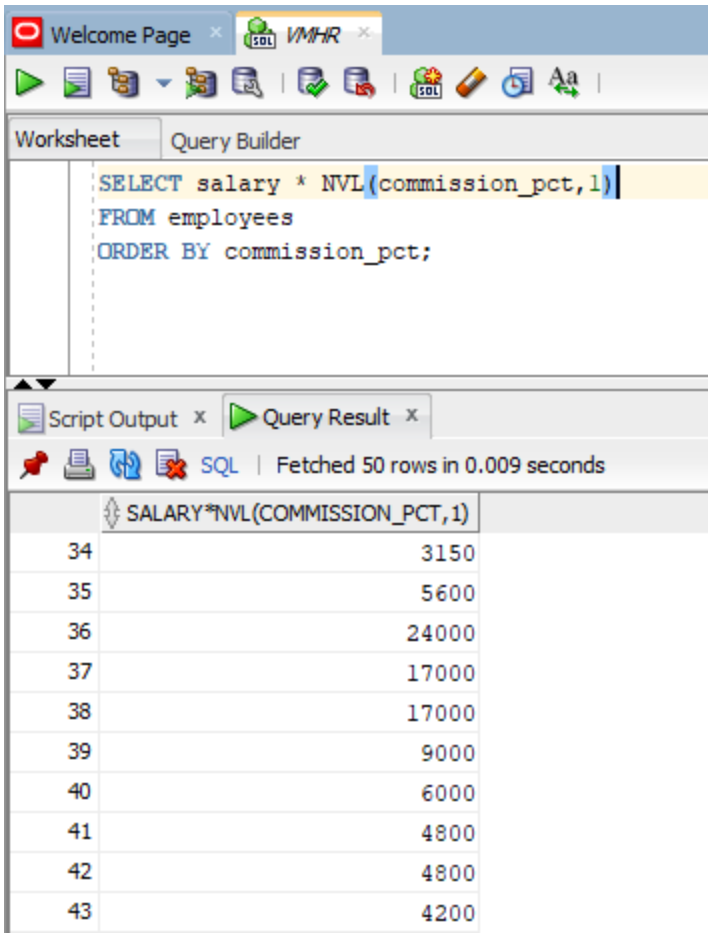
⇒ null로 계산하면 결과가 null이 됨.

▼ 실행



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

▼ 실행



▼ 조건 처리

DECODE - 조건 처리하기

DECODE(열이름, 조건값, 치환값, 기본값)

치환값 - 조건을 만족할 경우

기본값 - 조건을 만족하지 않을 경우

▼ 실행

Worksheet | Query Builder

```
SELECT department_id,
       employee_id,
       first_name,
       salary 원래급여,
       DECODE(department_id, 60, salary*1.1, salary) 지급급여,
       DECODE(department_id, 60, '급여인상' , ' ') 확인

FROM employees;
```

Script Output | Query Result

SQL | Fetched 50 rows in 0.015 seconds

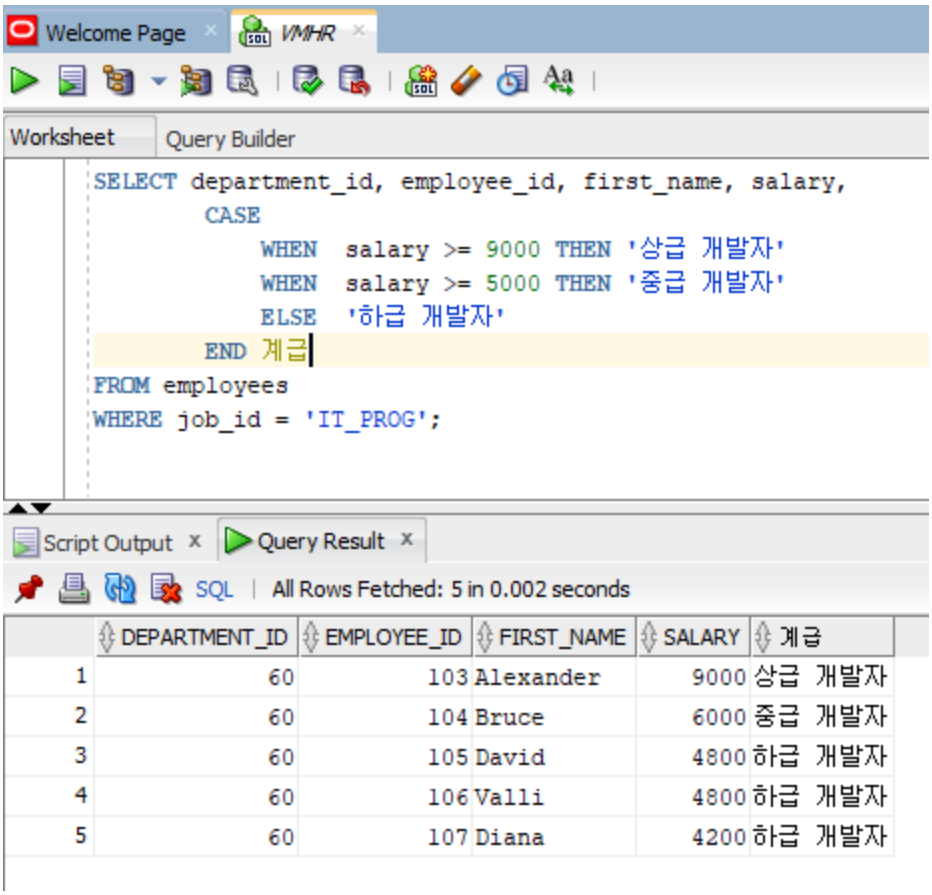
	DEPARTMENT_ID	EMPLOYEE_ID	FIRST_NAME	원래급여	지급급여	확인
1	90	100	Steven	24000	24000	
2	90	101	Neena	17000	17000	
3	90	102	Lex	17000	17000	
4	60	103	Alexander	9000	9900	급여인상
5	60	104	Bruce	6000	6600	급여인상
6	60	105	David	4800	5280	급여인상
7	60	106	Valli	4800	5280	급여인상
8	60	107	Diana	4200	4620	급여인상

department_id 가 60이면 → 치환값 / 아니면 → 기본값

CASE - 조건 여러개

CASE
WHEN 조건 THEN 출력값
WHEN 조건 THEN 출력값
ELSE 그 외 출력값
END

▼ 실행



▼ NUMBERING 순위 매기기 3가지

RANK

공통 순위 만큼 건너 뛰어 순위 매기기

1, 2, 2 → 4

DENSE_RANK

공통 순위 건너뛰지 않고 순위 매기기

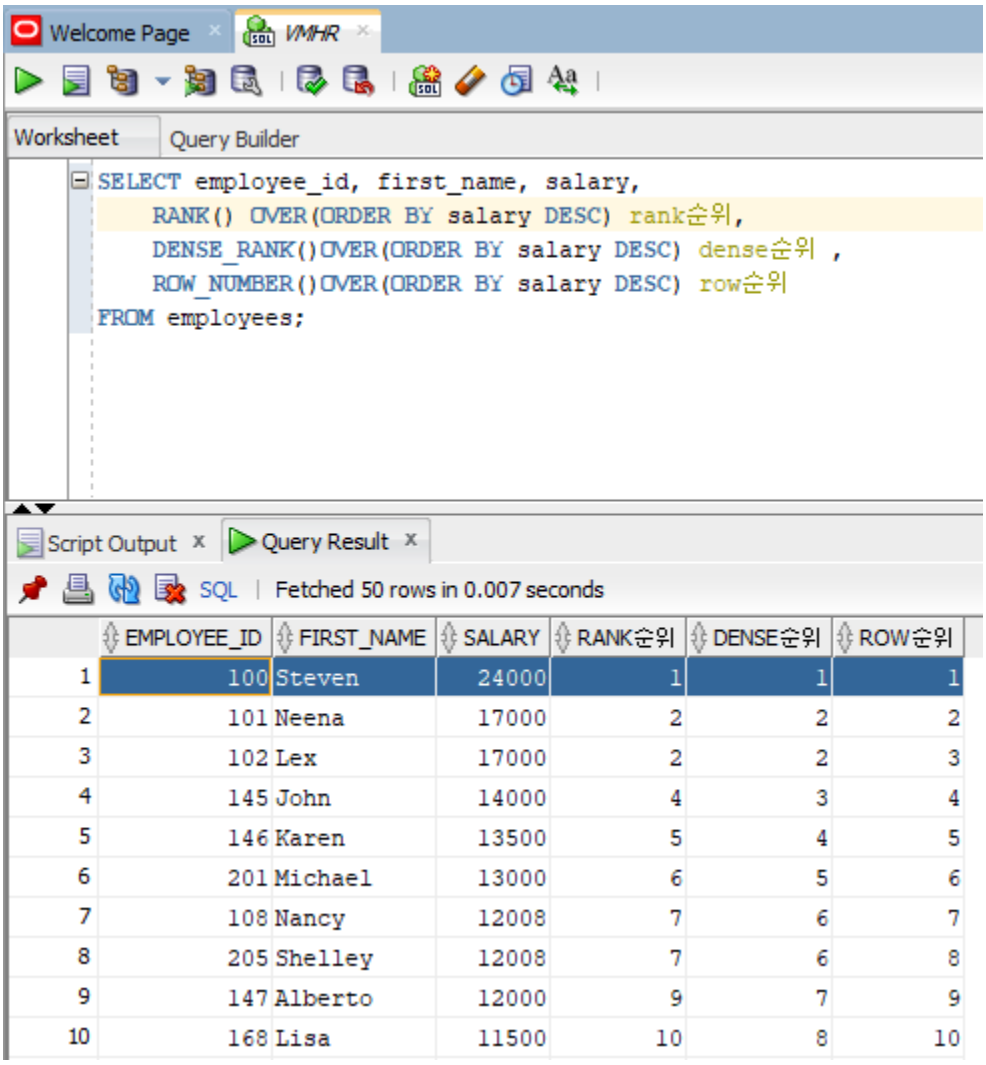
1, 2, 2 → 3

ROW_NUMBER

공통 순위 없이 출력

1, 2, 3, 4

▼ 실행



	EMPLOYEE_ID	FIRST_NAME	SALARY	RANK순위	DENSE순위	ROW순위
1	100	Steven	24000	1	1	1
2	101	Neena	17000	2	2	2
3	102	Lex	17000	2	2	3
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

▼ 그룹 함수[수 계산]

여러행에 함수가 적용이 되어 하나의 결과를 나타냄

null값 포함 계산

COUNT 갯수

NULL값을 포함ㄱ해서 수를 세므로

어떤 열로 세도 갯수가 동일하기 때문에 COUNT(*)로 많이 표현한다.

null값 제외 계산

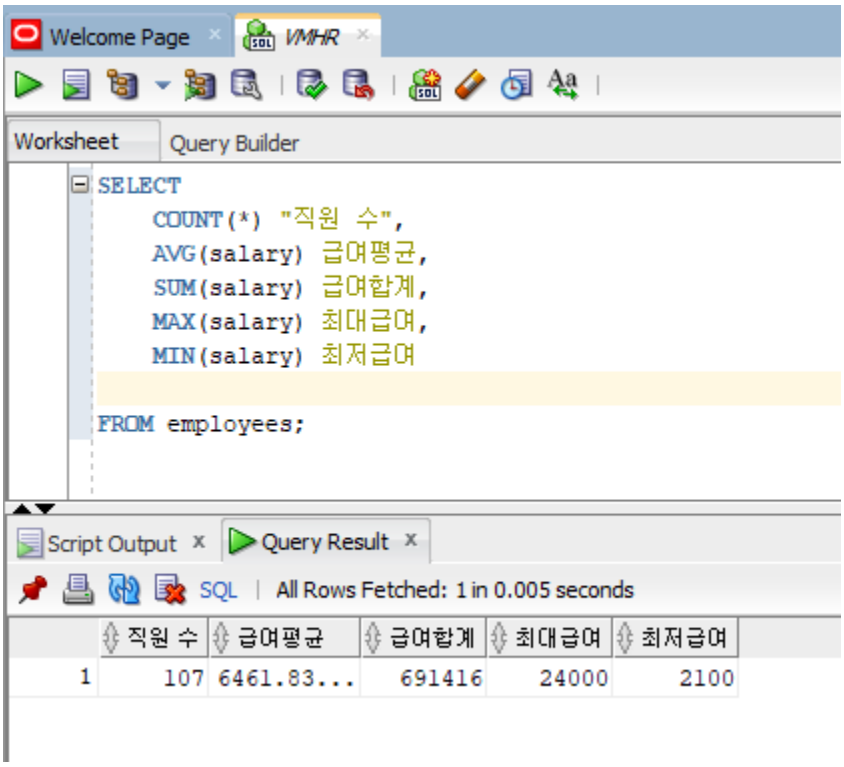
SUM 합계

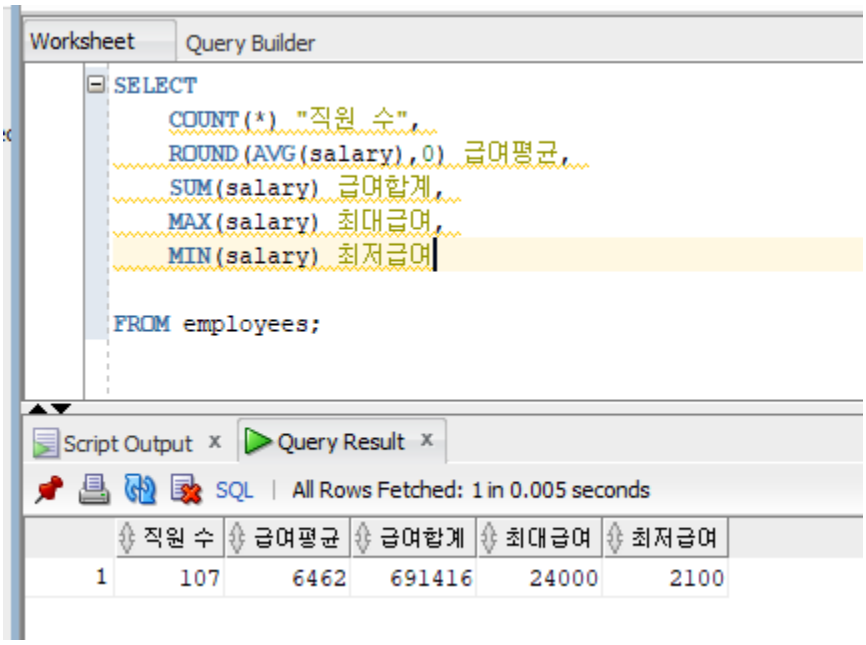
AVG 평균

MAX 최대값

MIN 최소

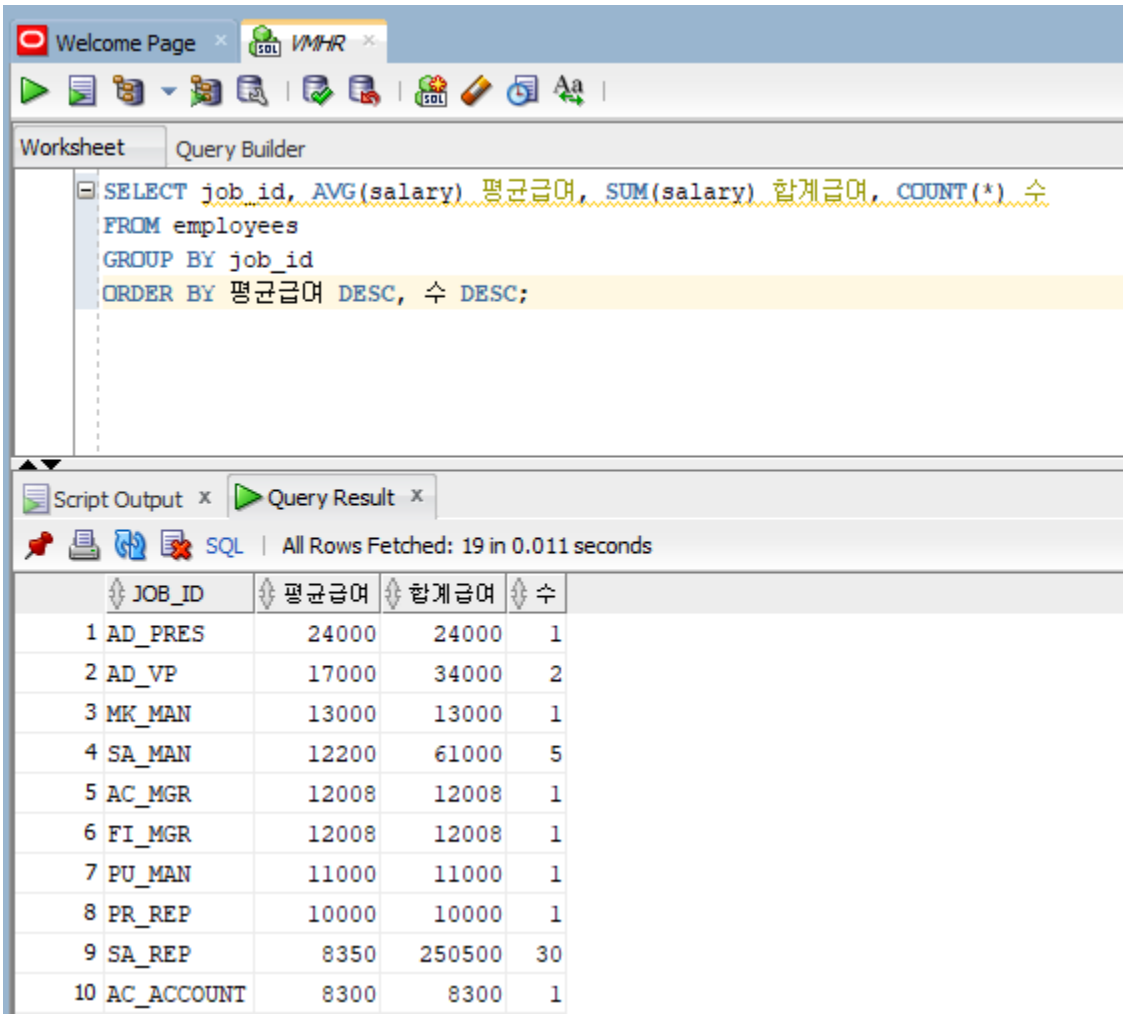
▼ 실행





평균을 반올림해서 깔끔하게 정리

▼ Group By 그룹으로 묶기



	JOB_ID	평균급여	합계급여	수
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

▼ DML, DDL, DCL

- **DML [Data Manipulation Language] / 데이터 조작 언어**

테이블에 있는 행과 열을 조작하는 언어

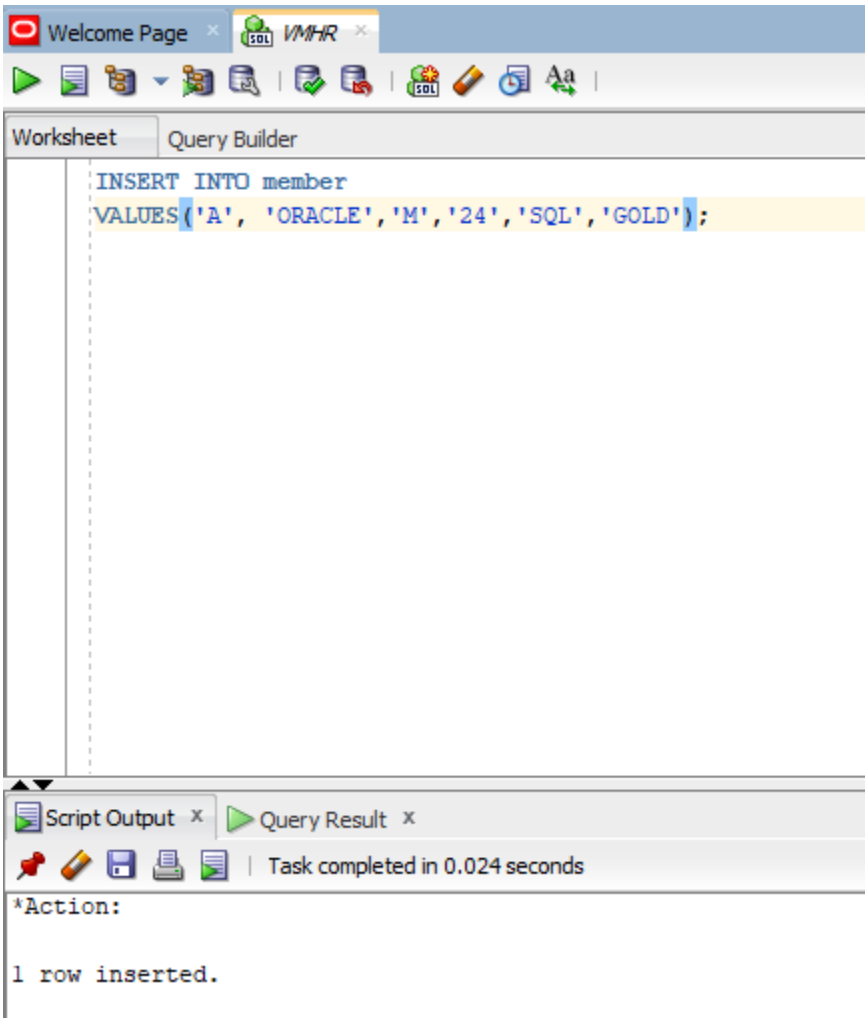
SELECT 조회[read]

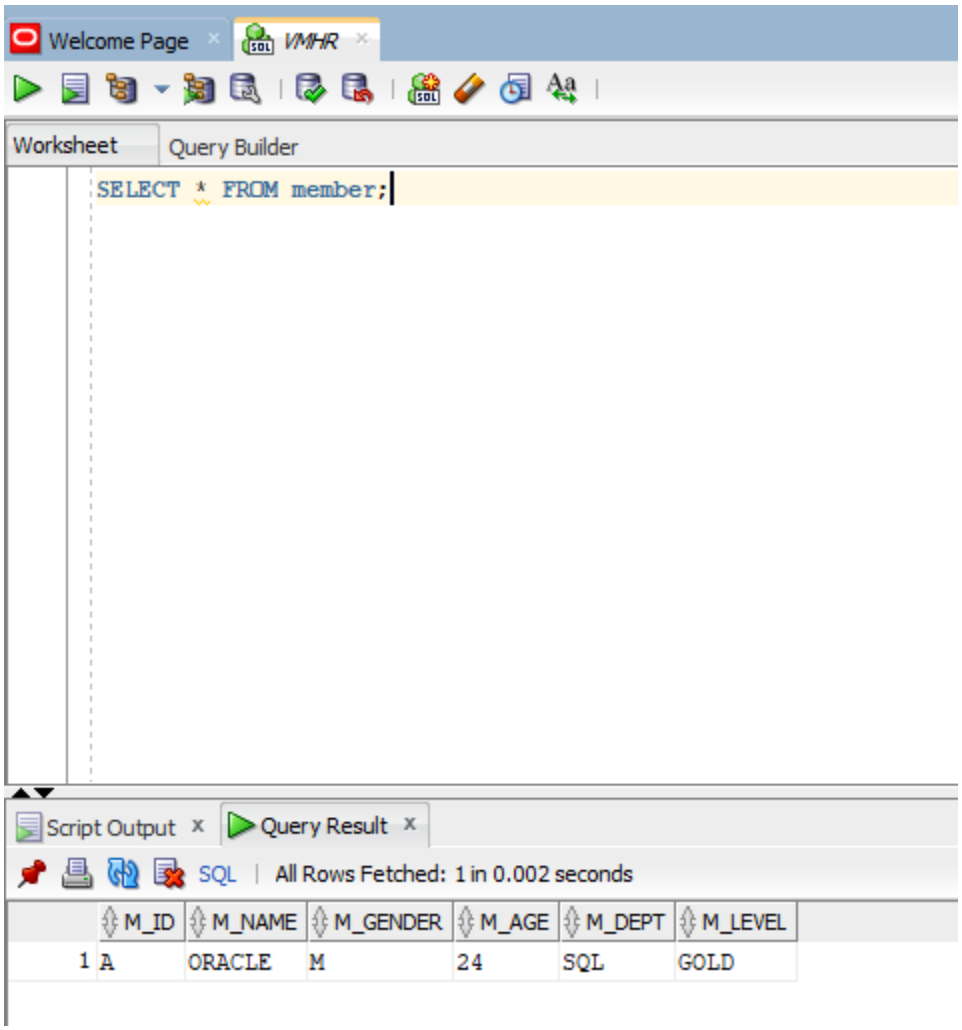
INSERT 삽입[crete]

UPDATE 수정[update]

DELETE 제거[delete]

▼ INSERT


Welcome Page
VMHR
Worksheet
Query Builder
INSERT INTO member
VALUES('A', 'ORACLE','M','24','SQL','GOLD');
Script Output
Query Result
Task completed in 0.024 seconds
*Action:
1 row inserted.



- **DDL [Data Definition Language] / 데이터 정의 언어**

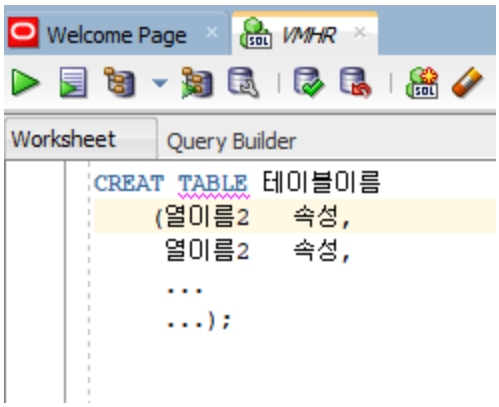
CREATE DB 생성, Table 생성

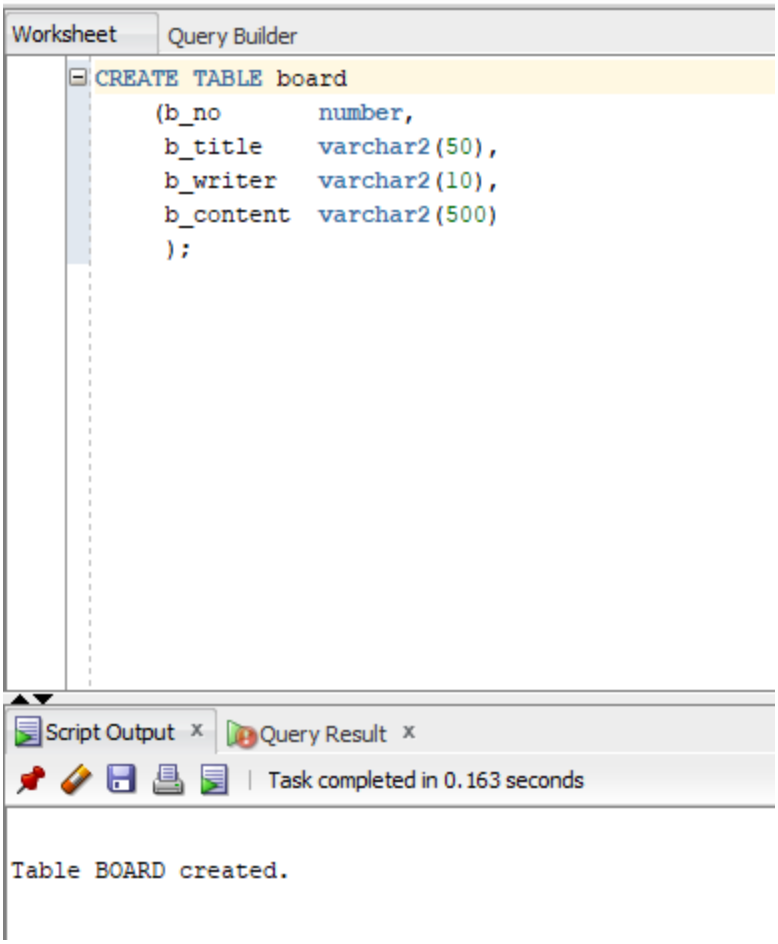
ALTER 수정

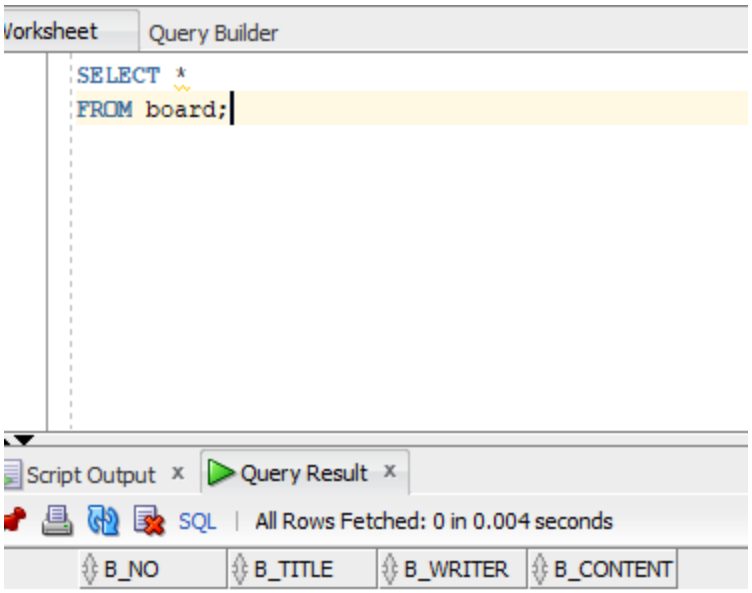
TRUNCATE 초기화

DROP 삭제

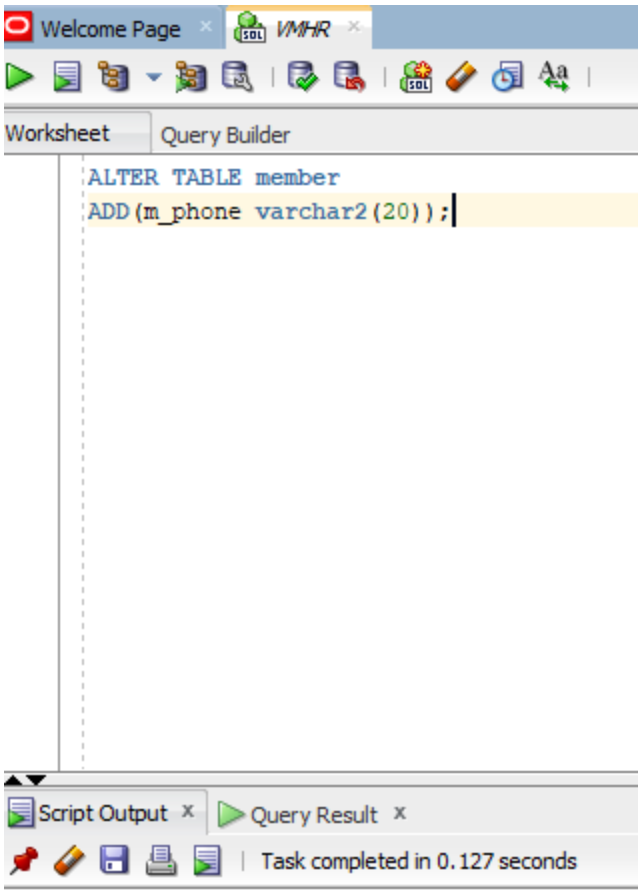
▼ CREATE


Welcome Page
VMHR
Worksheet
Query Builder
CREAT TABLE 테이블이름
(열이름2 속성,
열이름2 속성,
...
...);

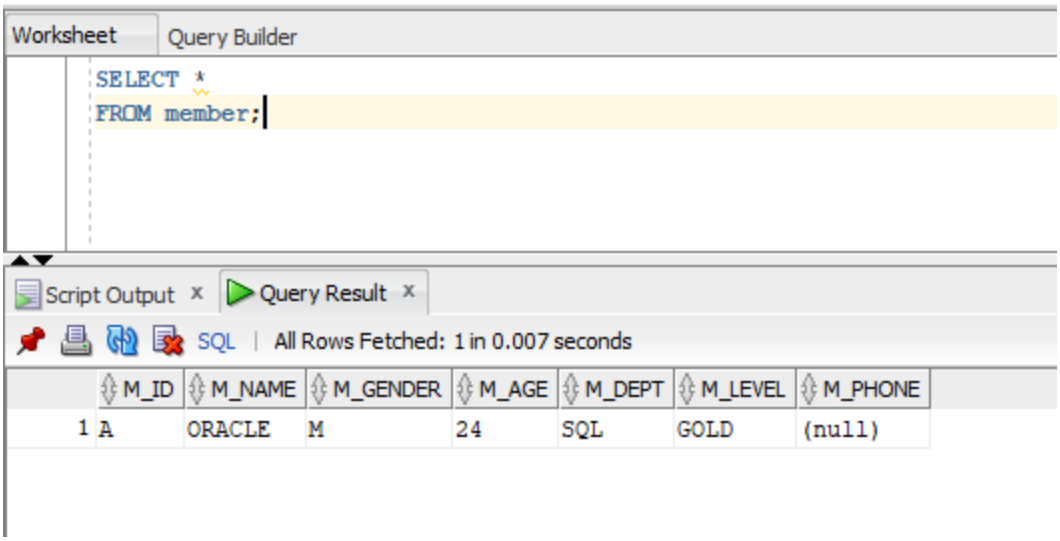

Worksheet
Query Builder
CREATE TABLE board
(b_no number,
b_title varchar2(50),
b_writer varchar2(10),
b_content varchar2(500)
);
Script Output
Query Result
Task completed in 0.163 seconds
Table BOARD created.


Worksheet
Query Builder
SELECT *
FROM board;
Script Output
Query Result
SQL
All Rows Fetched: 0 in 0.004 seconds
B_NO
B_TITLE
B_WRITER
B_CONTENT

▼ ALTER[항목 추가]

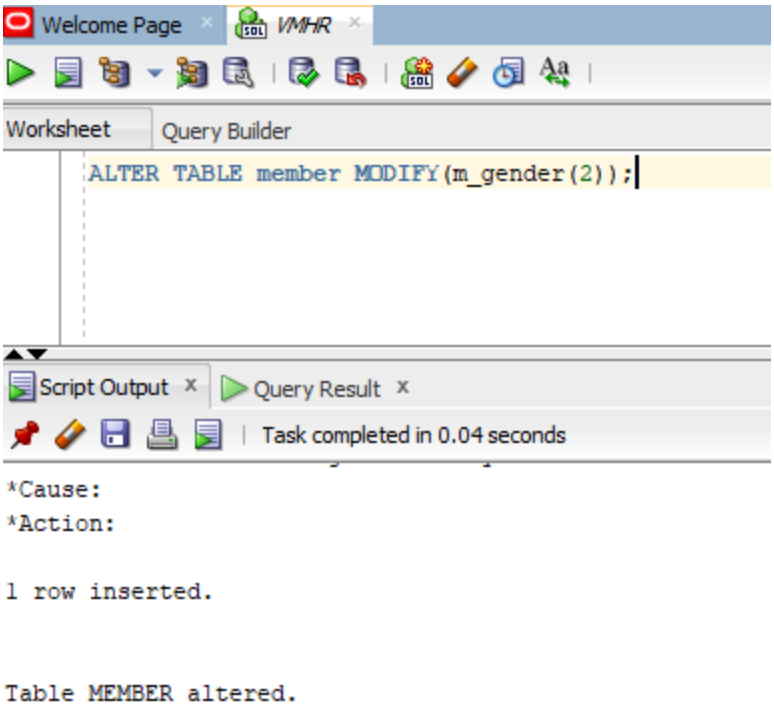



```
Table MEMBER altered.
```

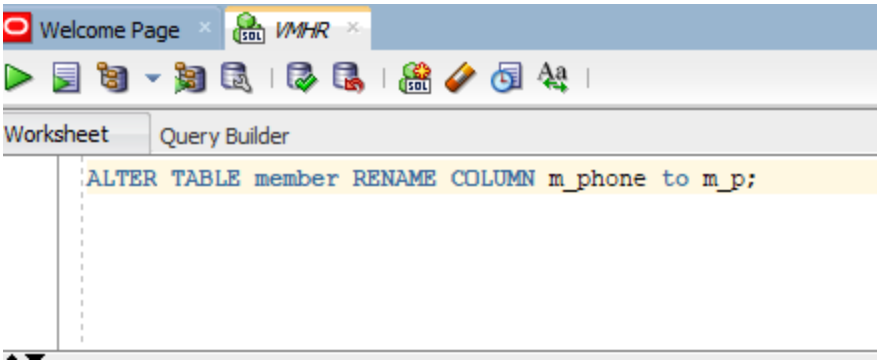


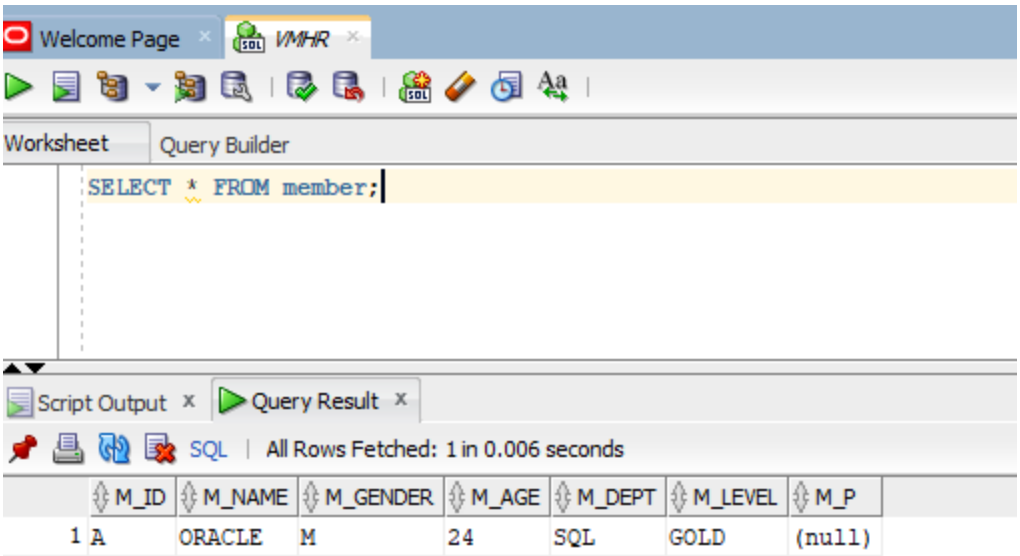
▼ ALTER[항목 변경]

MODIFTY 속성 변경[크기변경]

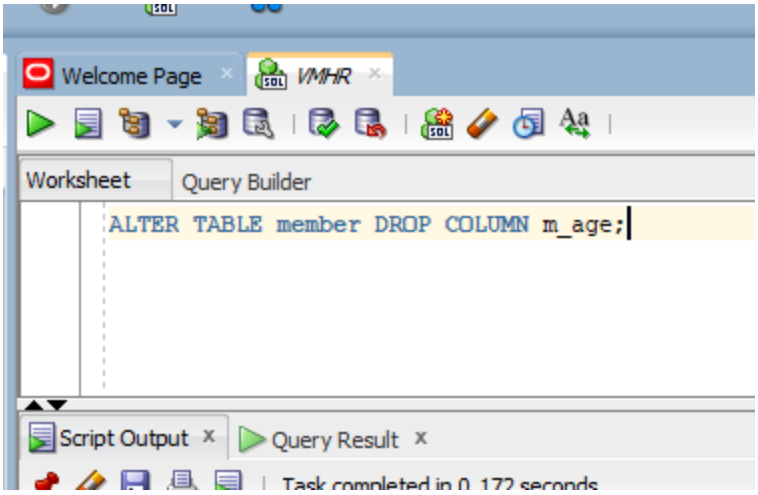


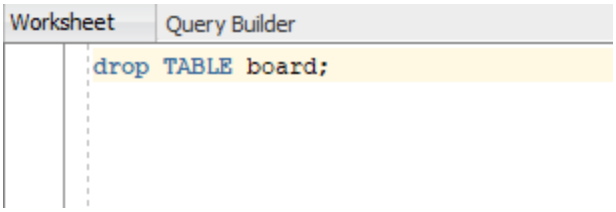
RENAME [이름변경]

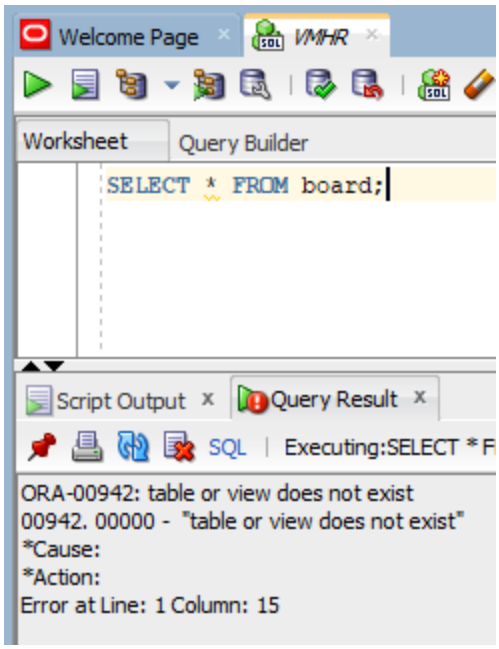



Welcome Page
VMHR
Worksheet
Query Builder
SELECT * FROM member;
Script Output
Query Result
SQL
All Rows Fetched: 1 in 0.006 seconds
M_ID M_NAME M_GENDER M_AGE M_DEPT M_LEVEL M_P
1 A ORACLE M 24 SQL GOLD (null)

▼ DROP 삭제

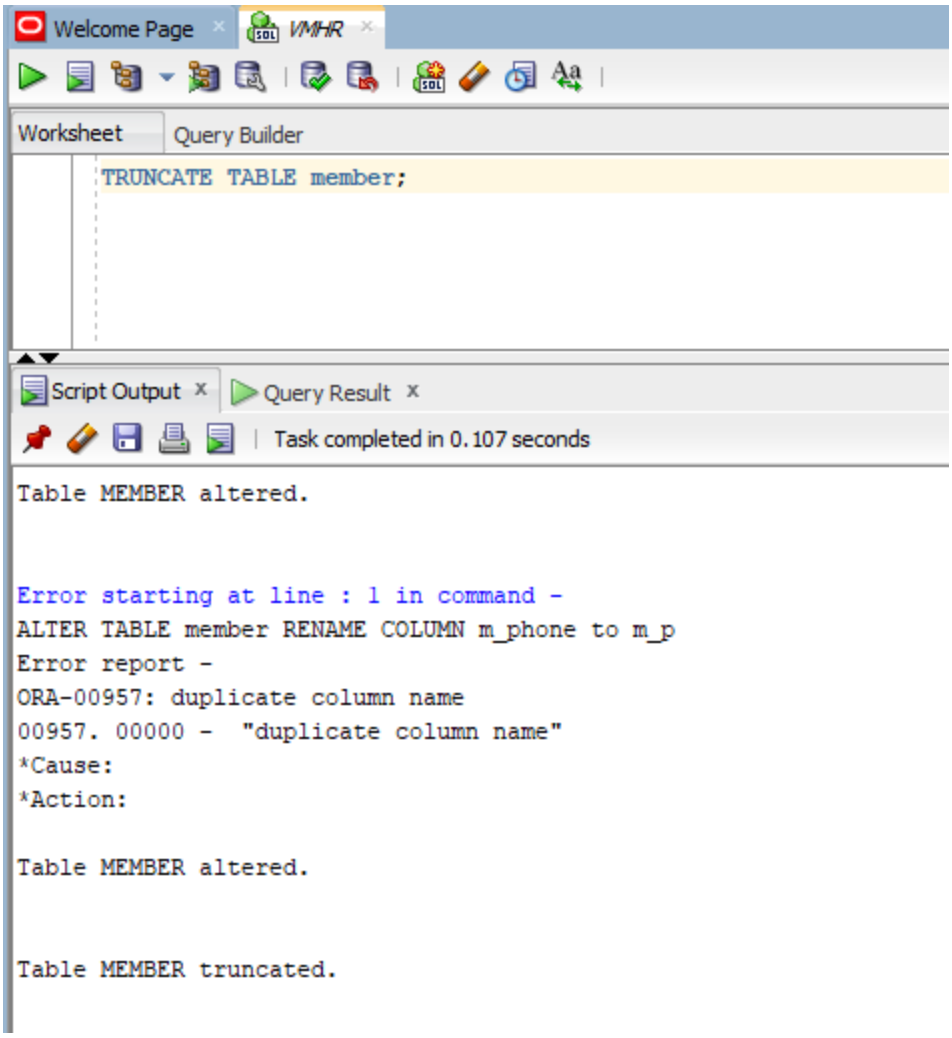

Welcome Page
VMHR
Worksheet
Query Builder
ALTER TABLE member DROP COLUMN m_age;
Script Output
Query Result

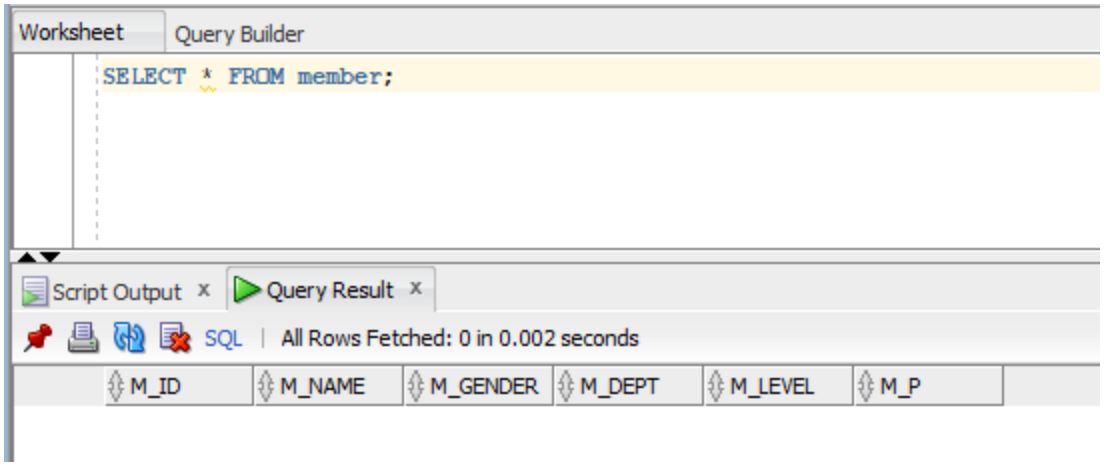

Worksheet
Query Builder
drop TABLE board;



▼ TRUNCATE 삭제

내용만 지움, 테이블(구조)는 남아있음





- **DCL [Data Control Language] / 데이터 제어 언어**

DB에 접근하거나 객체에 권한을 주는 등의 역할

GRANT 권한 부여

REVOKE 권한 박탈, 회수

COMMIT 완료 알림

ROLLBACK 비정상 종료 시 복구

TABLE이름 정의 방법

동일한 이름의 테이블이 존재할 수 없다.

예약어 즉 이미 사용중인 명령어 등으로는 이름을 사용할 수 없다.

반드시 문자로 시작(한글도 쓸 수는 있지만 사용하지 않는 것이 좋음)

가능하면 의미있는 단어를 사용하자

삭제

DML delete 데이터만 삭제

DDL truncate 구조를 남기고 데이터만 전체 삭제

DDL drop 구조 포함 데이터 전체 완전히 삭제

▼ View - 가상의 테이블

뷰의 장점

보안제공

→뷰를 사용하면 기본 테이블에서 필요한 데이터만 추출하여 보여줄 수 있어서 보호에 유용

복잡한 쿼리를 단순화, 데이터 조회 시간을 감소

→ 데이터베이스의 응답 시간을 단축시켜 더 높은 효율성

뷰는 여러 가지 사용방법

여러 테이블에서 데이터를 조합

특정 조건에 따라 데이터를 필터링

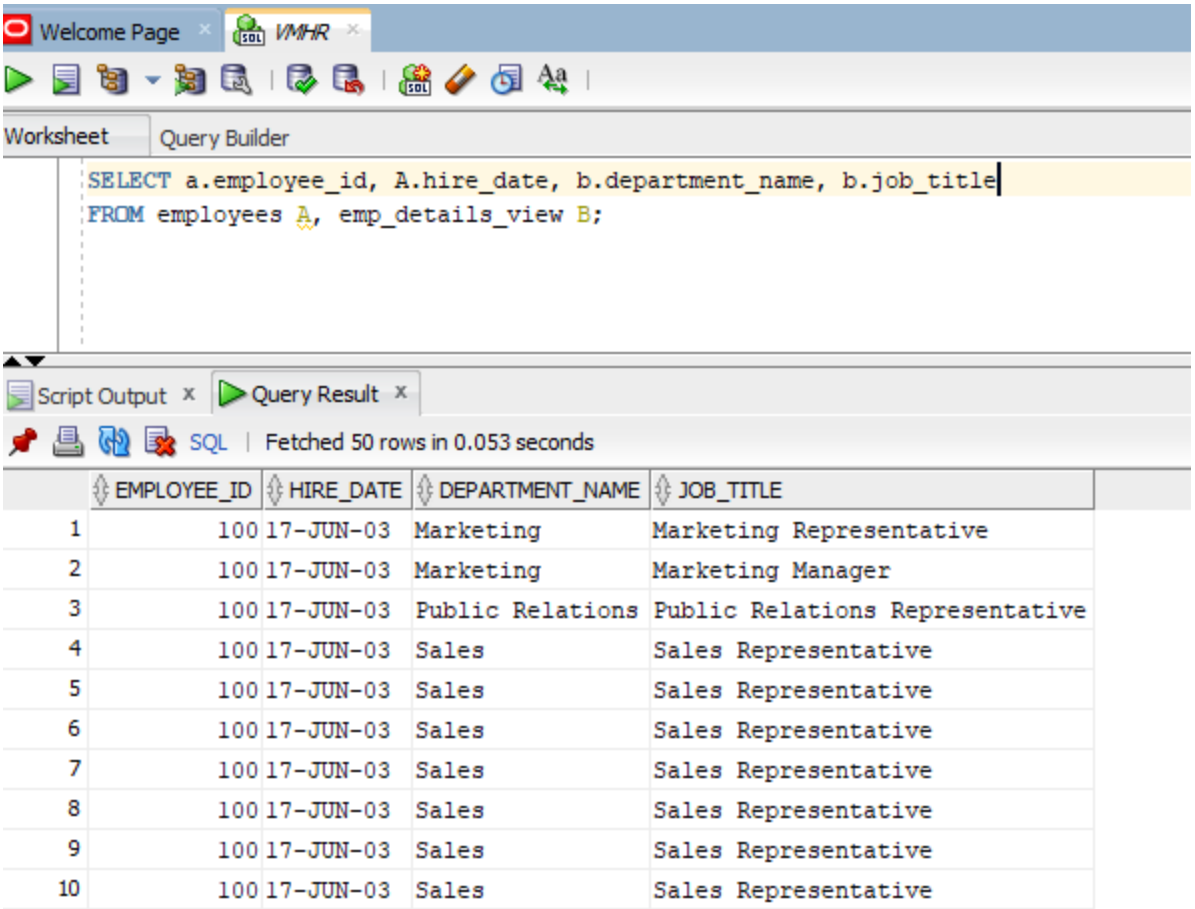
계산된 값을 생성 등

뷰의 단점

기본 테이블에서 변경된 내용이 반영되지 않을 수 있음

뷰를 사용하는 경우, 기본 테이블에서 변경된 내용이 뷰에 반영되지 않으면, 뷰에 저장된 데이터는 잘못된 결과를 반환할 수 있음

마지막으로, 뷰는 대규모 시스템에서 매우 유용하지만, 작은 규모의 시스템에서는 뷰를 사용하지 않는 것이 좋습니다. 작은 규모의 시스템에서는 뷰를 사용하지 않는 것이 더 효율적이며, 뷰를 사용하면 오히려 성능에 악영향을 미칠 수 있습니다.



	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
7	100	17-JUN-03	Sales	Sales Representative
8	100	17-JUN-03	Sales	Sales Representative
9	100	17-JUN-03	Sales	Sales Representative
10	100	17-JUN-03	Sales	Sales Representative