SQL CLASS 1

▼ DBMS

데이터베이스란 데이터를 저장, 검색, 관리하고 조직화하는데 사용되는 컴퓨터시스템

데이터베이스는 데이터베이스 관리 시스템(Database Management System)에 의해 관리

사용자나 프로그램이 데이터에 접근 가능하게 함

1. **데이터베이스 유형**

   1) 관계형 데이터베이스(Relational Database)

   2) NoSQL(Not Only SQL)

   - 관계형 데이터베이스 :

   테이블(행과 열) 형태로 데이터를 저장, SQL을 사용하여 데이터에 접근

   ACID(Atomicity, Consistency, Isolation, Durability)의 속성을 유지

   ex - MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server etc..

   - NoSQL :

   관계형데이터베이스와는 달리, 여러 유형의 데이터 구조를 허용

   대용량의 분산 데이터 처리에 유용, 확장성 높음

   분류 - 문서형(Document), 키값(Key-value), 컬럼패밀리, 그래프

   ex - MongoDB, Redis, Cassandra, Neo4j

2. **데이터베이스 구성 요소**

   - 스키마(Schema): 데이터베이스의 구조와 제약 조건을 정의. 테이블, 인덱스, 뷰, 프로시저 등의 객체를 포함.
   - 테이블(Table): 행과 열로 이루어진 데이터 저장 구조,

     각 열⇒ 데이터의 속성, 각 행⇒ 데이터의 레코드를 나타냄.

- 인덱스(Index): 데이터 검색 속도를 높이기 위해 사용되는 데이터 구조. 일종의 데이터베이스 '색인'으로, 특정 열의 값에 따라 레코드를 빠르게 찾을 수 있음.
- 뷰(View): 데이터베이스의 가상 테이블, 실제로 데이터를 저장하지 않지만 기본 테이블로부터 데이터를 참조하고 질의를 수행. 뷰는 데이터 접근을 단순화하거나, 보안 목적으로 특정 데이터에 대한 접근을 제한함.
- 트랜잭션(Transaction): 데이터베이스에서 실행되는 하나의 논리적 작업 단위. 데이터의 일관성과 동시성을 유지하기 위해 ACID 속성 지님.
- 저장 프로시저(Stored Procedure): 데이터베이스 서버에서 실행되는 일련의 SQL 명령어로 구성된 코드 블록. 저장 프로시저는 코드 재사용, 성능 향상 및 보안 향상과 같은 이점을 제공함.
- 트리거(Trigger): 특정 이벤트(INSERT, UPDATE, DELETE 등)가 발생할 때 자동으로 실행되는 사용자 정의 코드. 트리거는 데이터의 일관성을 유지하거나, 추가 로직을 적용하는 데 사용.
- 제약 조건(Constraint): 데이터의 무결성을 유지하기 위해 테이블에 적용되는 규칙. NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK 등이 있음.
- 데이터베이스 관리 시스템(DBMS): 데이터베이스를 생성하고, 조작하며, 관리하는 소프트웨어입니다. 사용자와 프로그램이 데이터베이스와 상호 작용할 수 있게 합니다.

  구분 - 관계형 DBMS / No SQL DBMS

▼ SQL

- SQL(Structured Query Language) : 관계형 DBMS에서 데이터를 검색, 수정, 추가 및 삭제하기 위해 사용되는 표준 프로그래밍 언어

  관계형 데이터베이스에서 데이터를 조작하고 관리하기 위한 명령어와 구문 제공

  다양한 RDBMS 제품 (ex - Oracle, MySQL, PostgreSQL, Microsoft SQL Server) 및 ISO에 의해 표준화
- SQL의 작업

  1) 데이터 검색(조회) : SELECT문 사용, 데이터베이스의 특정 테이블에서 원하는 데이터를 검색 또는 조회 가능. 필터링, 정렬, 그룹화, 결합 등의 다양한 옵션 사용 가능

  2) 데이터 삽입 : INSERT문 사용, 테이블의 기존 레코드(행)을 추가 가능. 각 열에 대한 값을 지정하거나 다른 테이블로부터 값 복사 가능

3) 데이터 수정 : UPDATE문 사용, 테이블의 기존 레코드(행)을 수정 가능. 조건을 지정함으로써 특정 레코드만 수정 가능

4) 데이터 삭제 : DELETE문 사용, 테이블에서 레코드(행)을 삭제 가능. 조건을 지정함으로써 특정 레코드만 삭제 가능

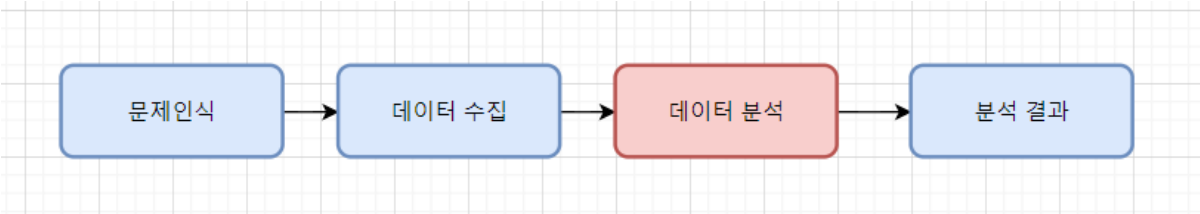
5) 테이블 및 데이터베이스 구조 조작 : CREATE, ALTER, DROP문(DDL, 데이터정의어)을 사용. 데이터베이스, 테이블, 인덱스, 뷰 등의 구조를 생성, 수정, 삭제 기능을 가짐

6) 트랜잭션 제어 : COMMIT, ROLLBACK, SAVEPOINT문(TCL, 트랜잭션 제어어)을 사용. 트랜잭션을 제어하고 데이터의 일관성 유지

7) 데이터베이스 접근 제어 : GRANT, REVOKE문(DCL, 데이터제어어)을 사용. 사용자에게 데이터베이스 객체에 대한 권한을 부여 또는 취소 가능. ⇒ 데이터의 보안 유지

Database

데이터 분석과정



- 전처리 : 데이터를 수집하고 가공하는 과정, 데이터 전체 분석 과정의 약 80% 차지
- 정형 데이터 : 틀이 잡힌 데이터 ex) excel etc..
- 비정형 데이터 : 틀이 없고 다양하고 방대한 데이터 ex) 트위터, 음악, 그림 etc..

▼ SQL 활용

SELECT (데이터 조회) : 가장 많이 사용되는 구문

- 기본형

SELECT 열이름 FROM 테이블이름;

SELECT 열이름1, 열이름2

FROM 테이블이름;

※ SQL 구문은 대문자로 사용 (관용적 표현, 소문자도 가능)

※ 줄바꿈, 들여쓰기는 특별히 제한하지 않음

※ 세미콜론(;)으로 마쳐야만 함

```
1  SELECT first_name, last_name FROM hr.employees;
2
```

FIRST_NAME	LAST_NAME
Ellen	Abel
Sundar	Ande
Mozhe	Atkinson
David	Austin
Hermann	Baer
Shelli	Baida
Amit	Banda
Elizabeth	Bates

```
SELECT first_name, last_name
    FROM hr.employees;
```

줄바꿈, 들여쓰기 상관x

```
SELECT *
FROM hr.employees;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	03/06/17	AD_PRES	24000	-	-	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	05/09/21	AD_VP	17000	-	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	01/01/13	AD_VP	17000	-	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	06/01/03	IT_PROG	9000	-	102	60

전체 데이터 조회 (SELECT * FROM table;)

- 정렬 조회 ORDER BY

SELECT * FROM 테이블이름

ORDER BY 열이름 오름(내림)차순;

※ 오름차순 ⇒ ASC or 내림차순 ⇒ DESC

```sql
SELECT *
FROM hr.employees
ORDER BY employee_id DESC;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER
206	William	Gietz	WGIETZ	515.123.8181
205	Shelley	Higgins	SHIGGINS	515.123.8080
204	Hermann	Baer	HBAER	515.123.8888
203	Susan	Mavris	SMAVRIS	515.123.7777
202	Pat	Fay	PFAY	603.123.6666
201	Michael	Hartstein	MHARTSTE	515.123.5555
200	Jennifer	Whalen	JWHALEN	515.123.4444
199	Douglas	Grant	DGRANT	650.507.9844

employee_id 기준으로 내림차순 정렬

- 중복 값 제거 DISTINCT

SELECT **DISTINCT** 열이름

FROM 테이블이름;

```
1 SELECT DISTINCT job_id
2 FROM hr.employees;
3
```

JOB_ID
AC_ACCOUNT
AC_MGR
AD_ASST
AD_PRES
AD_VP

중복 값 제거, 고유 값만 출력 DISTINCT

- 별명(별칭) 사용 ALIAS / AS

※ 별명은 열이름을 임시로 보여줄 때만 바꿔서 표현

※ 원래 열이름 자체는 바뀌지 않음

※ 이름을 바꿀 때 AS를 사용하며 생략도 가능 (반드시 넣을 때도 있음)

※ 빈칸, 특수기호, 대소문자 등을 넣을 때는 반드시 “” 사용

```
SELECT first_name AS 성, last_name AS 이름
FROM hr.employees;
```

성	이름
Ellen	Abel
Sundar	Ande
Mozhe	Atkinson

AS로 별명 설정

- 열 값들을 하나로 출력 || (버티컬 바 / 연결연산자)

```
SELECT first_name||last_name
FROM hr.employees;
```

FIRST_NAME||LAST_NAME
EllenAbel
SundarAnde
MozheAtkinson

```sql
SELECT first_name||' '||last_name
FROM hr.employees;
```

FIRST_NAME||''||LAST_NAME
Ellen Abel
Sundar Ande
Mozhe Atkinson

```sql
SELECT first_name||' '||last_name,
       email||'@naver.com'
FROM hr.employees;
```

FIRST_NAME||''||LAST_NAME	EMAIL||'@NAVER.COM'
Ellen Abel	EABEL@naver.com
Sundar Ande	SANDE@naver.com
Mozhe Atkinson	MATKINSO@naver.com

문자열 연결 ||(버티컬 바)

- 열값 계산하여 출력(산술연산 +-*/)

```sql
SELECT employee_id, salary, salary+500, salary-100,salary*1.1
FROM hr.employees;
```

EMPLOYEE_ID	SALARY	SALARY+500	SALARY-100	SALARY*1.1
100	24000	24500	23900	26400
101	17000	17500	16900	18700
102	17000	17500	16900	18700

사칙연산 계산 but 웹개발에서는 SQL보다 JAVA에서 계산

```sql
SELECT employee_id 사원번호,
       salary 급여,
       salary+500 추가급여,
       salary-100 급여인하,
       salary*1.1 보너스
FROM hr.employees;
```

사원번호	급여	추가급여	급여인하	보너스
100	24000	24500	23900	26400
101	17000	17500	16900	18700
102	17000	17500	16900	18700

SQL 자체에서 계산하는 경우는 적지 않음

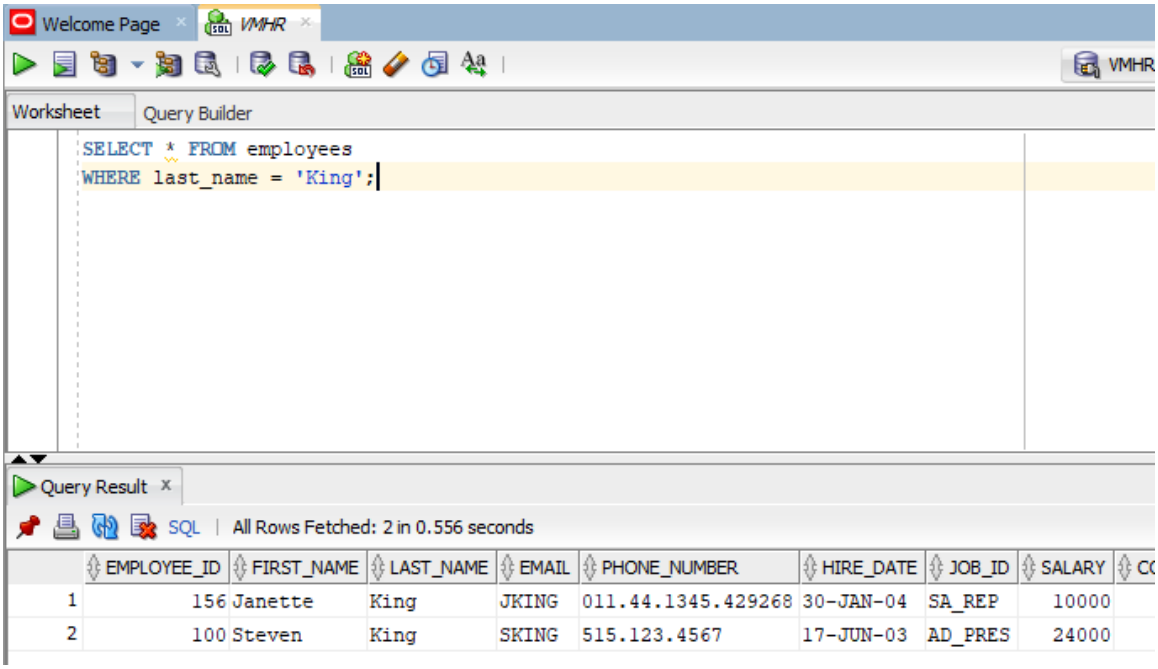
- WHERE 조건절

※반드시 from 다음에 사용

SELECT 열이름

FROM 테이블이름

WHERE 조건;



조건 설정

SELECT는 select로 대소문자를 가리지는 않지만 찾을 값에서는 대소문자를 가림

→ ‘King’과 ‘king’은 다르게 취급, SELECET와 select는 같게 취급

- 복잡한 조건 적용

BETWEEN 1 AND 4 ⇒ 1과 4 사이 조회

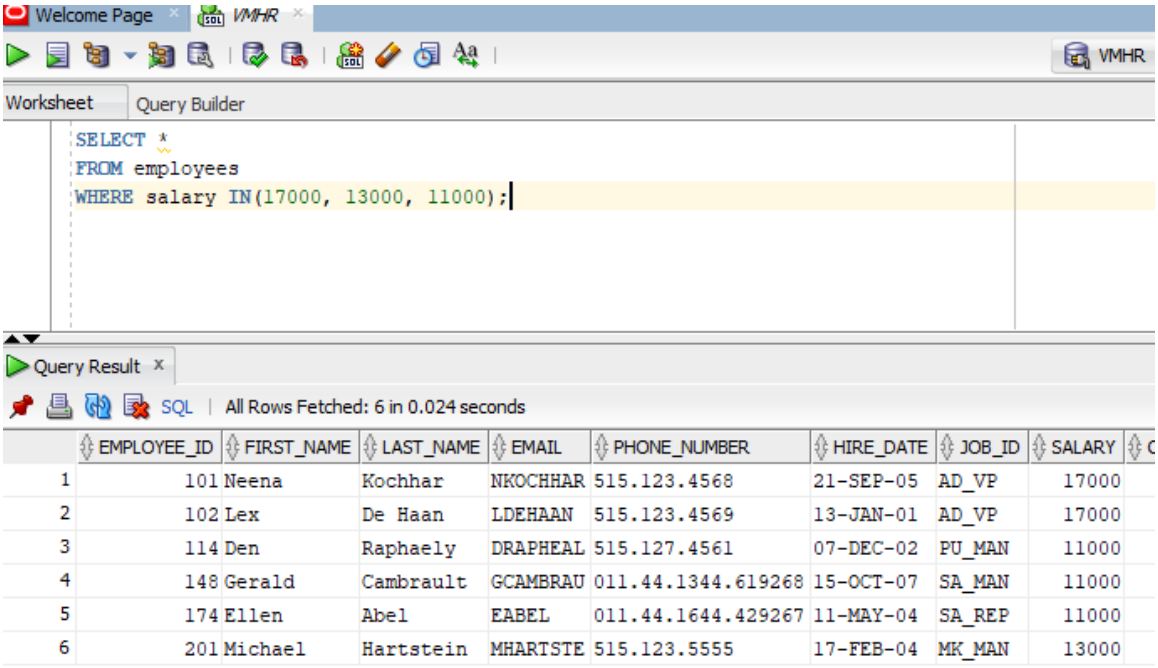
IN(1,2,3,4) ⇒ 1과 4 사이 조회

LIKE ‘%도’ ⇒ “도”로 끝나는 단어들 조회

LIKE ‘%@%’ ⇒ “@”가 포함하는 단어들 모두 조회

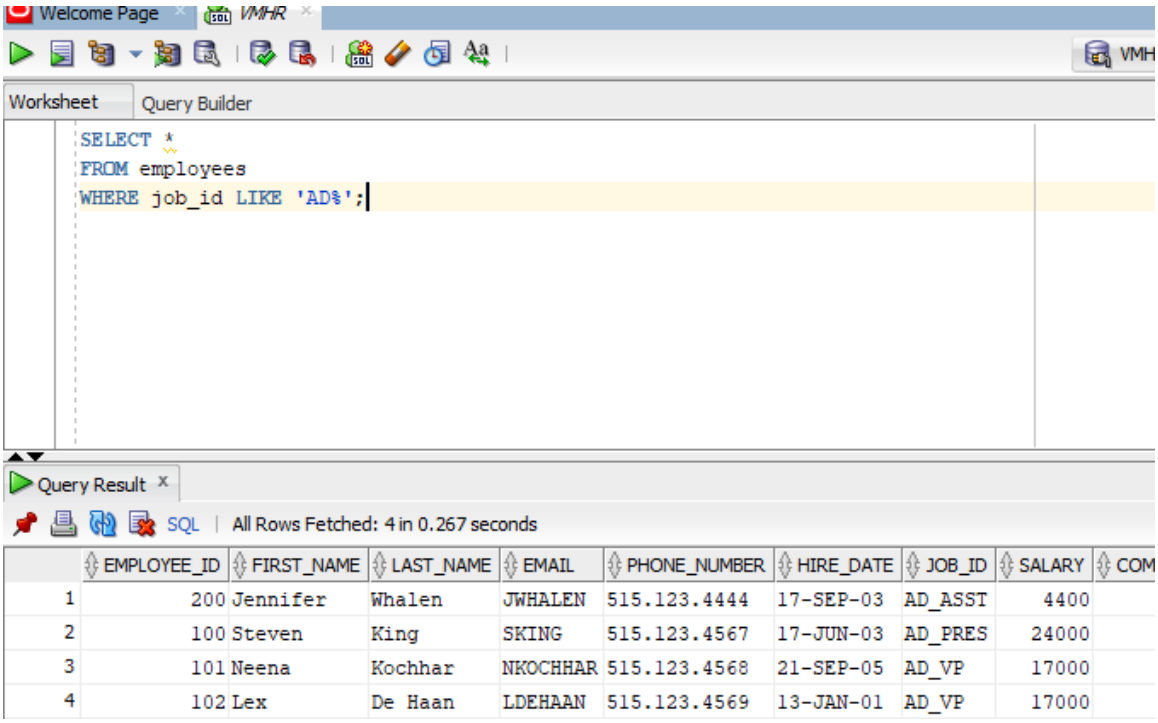
LIKE ‘서울%’ ⇒ “서울”로 시작하는 단어들만 모두 조회

IS NULL ⇒ null 찾기



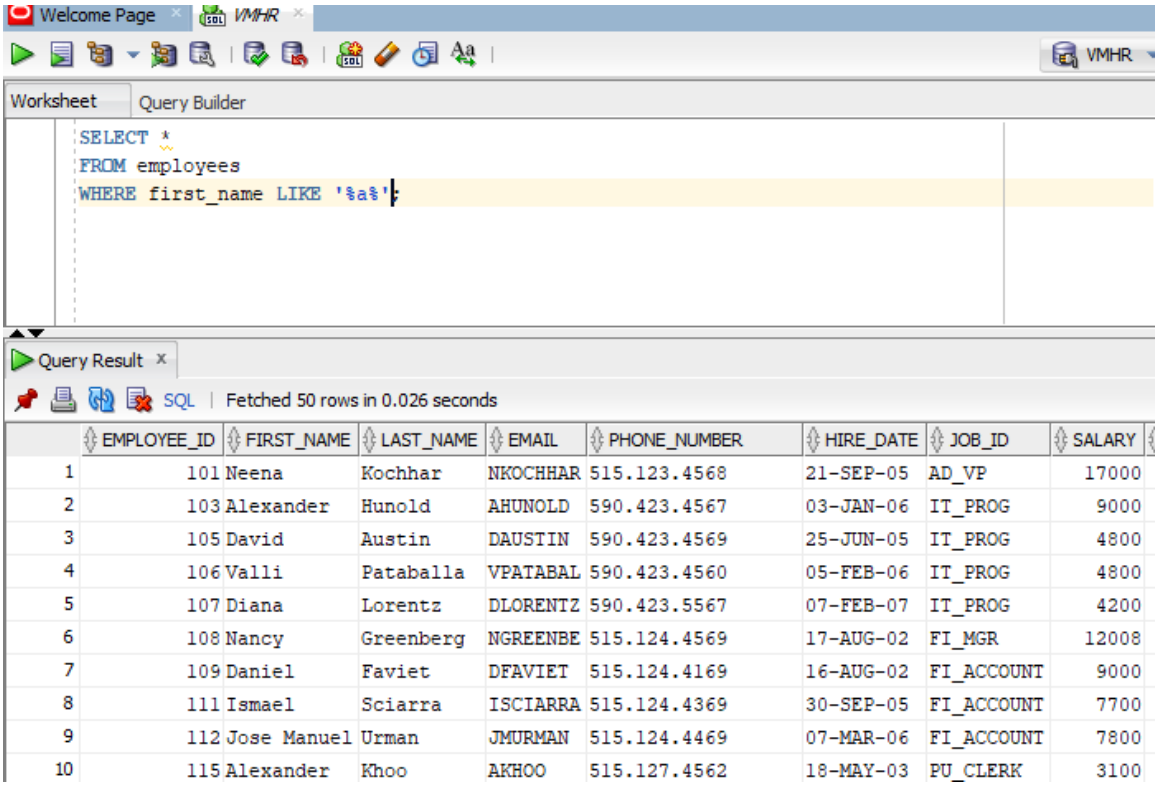
	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
1	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000
2	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000
3	114	Den	Raphaely	DRAPHEAL	515.127.4561	07-DEC-02	PU_MAN	11000
4	148	Gerald	Cambrault	GCAMBRAU	011.44.1344.619268	15-OCT-07	SA_MAN	11000
5	174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-04	SA_REP	11000
6	201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-04	MK_MAN	13000

IN 사용하여 11000, 13000, 17000인 값 추출



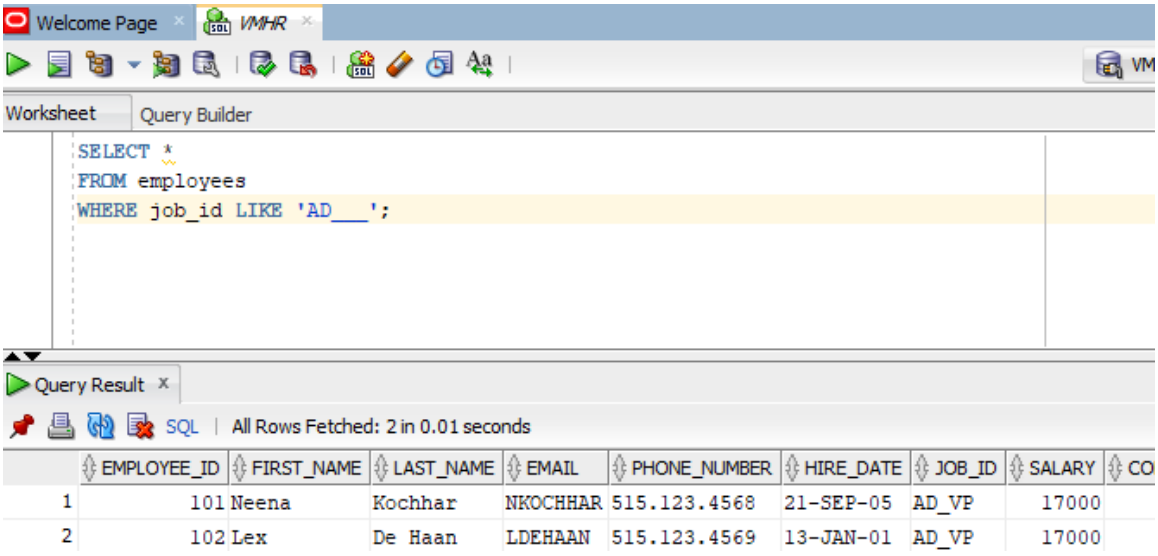
	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COM
1	200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-03	AD_ASST	4400	
2	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	
3	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	
4	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	

LIKE 사용하여 'AD'로 시작하는 job_id 추출



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
1	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000
2	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000
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	108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI_MGR	12008
7	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000
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	115	Alexander	Khoo	AKHOO	515.127.4562	18-MAY-03	PU_CLERK	3100

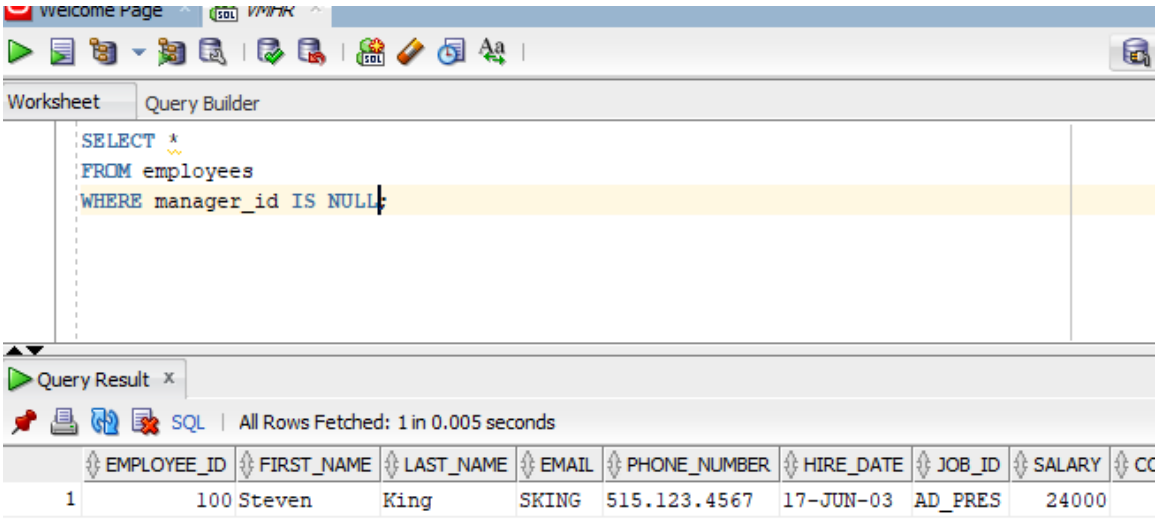
LIKE 사용하여 'a'가 포함된 first_name 추출



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	CO
1	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	
2	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	

'AD'로 시작하지만 뒤에 3글자의 공간을 갖고있는 job_id 추출

※ LIKE에서 '%' ⇒ 글자수 상관없음 / '_' ⇒ 글자수 카운트



IS NULL 이용하여 NULL값 찾기