/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

오라클 정규식(정규표현식) 함수 사용

- 정규표현식(regular expression, regexp):

- 패턴을 가진 문자열을 표현할때 사용하는 형식 언어.

- 문자열 내에서 특정 패턴의 문자열을 찾을 때 주로 사용

- 표현식에서 사용하는 문자

- 메타(Meta)문자: 패턴을 표현하는 문자.

- 리터럴(Literal) 문자: 표현식 자체가 값.

- 오라클 10g 부터 정규식을 지원하는 함수를 제공.

- regexp\_like()

- regexp\_substr()

- regexp\_replace()

- regexp\_count()

- 패턴에서 사용하는 메타문자 들

- 일치하는 문자 반복 개수 관련

\* : 앞의 문자(패턴)과 일치하는 문자가 0개 이상인 경우. ('Ab\*') >>A, Ab, Abb, Abb...

+ : 앞의 문자(패턴)과 일치하는 문자가 1개이상인 경우. ('Ab+') >>Ab, Abb, Abb....

? : 앞의 문자(패턴)과 일치하는 문자가 한개 있거나 없는 경우. ('A?b') >>b, Ab

. : 어떤 글자던지 한글자 와야한다. ('A.B') >>AaB, AbB, ...

{m} : 앞의 문자(패턴)가 m개. ('a{3}') >>aaa

{m, } : 앞의 문자(패턴)이 m개 이상. ('a{3, }') >>aaa, aaaa, aaaaa...

{m, n} : 앞의 문자(패턴)이 m개이상 n개 이하. ('a{2, 5}') >>aa, aaa, aaaa, aaaaa

- 문장의 시작과 끝 표현

^ : 문자열의 시작을 의미. ('^abc') >>abc로 시작

$ : 문자열의 끝을 의미 ('abc$') >>abc로 끝

- 특정 클래스 지정

- [ 패턴 ] : 클래스

- [ ] 사이의 문자들중 하나와 일치하는 것.

- "-"와 같이 쓰면 범위로 지정 가능

- [^ ] : 문자 클래스가 "^" 로 시작 하면 일치하지 않는 것이 된다.

- A[abc]>>Aa, Ab, Ac

A[abc]\*>>Aaa, Aabba, Aabc

[a-zA-Z]>>aZ 소문자 하나 대문자 하나

[가-갛]>>ㄱ이 들어가는 모든 글자

[가-힣]>>한글전체 중 아무거나

[0-9]>>0에서 9사이 숫자

- [: 클래스명 :] : 미리 지정된 클래스 사용

- [:alpha:] 또는 \a : 알파벳 문자 (\A는 알파벳를 제외한 모든 문자 하나)

- [:digit:] 또는 \d : 숫자 (\D는 숫자를 제외한 모든문자)

- [:lower:] : 소문자 알파벳 문자 [:upper:] : 대문자 알파벳 문자

- [:alnum:] : 또는 \w : 알파벳문자와 숫자 (\W : 숫자와 문자를 제외한 모든 문자(특수문자, 공백 등))

- [:space:] : 공백 문자 또는 \s. \S: 공백을 제외한 모든 문자

- [:punct:] : 특수 문자들 ($,\...)

- 기타

- | : 선택 (010|011) or의 의미

-\메타문자 : 메타문자를 패턴에 사용할 경우 앞에 \를 붙인다. (\. ) ex)\\d

-( ) : 식을 묶을때 사용.(패턴내에 하위패턴을 만든다) ex) a(b|c)d

- ( ) 로 묶인 패턴은 \순번 으로 호출 할 수 있다.

- ex: 'a(b|c)d\1' 는 'a(b|c)d(b|c)' 와 동일

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

--예제테이블

drop table regexp\_exam;

create table regexp\_exam(

no number primary key,

phone varchar(30),

email varchar(50)

);

insert into regexp\_exam values(1, '010-352-1029', 'aaa@daum.net');

insert into regexp\_exam values(2, '010-551-3011', 'bbbb@naver.net');

insert into regexp\_exam values(3, '02-8761-9200', 'ab.cde@naver.net');

insert into regexp\_exam values(4, '031-5432-5432', 'cccc@daum.net');

insert into regexp\_exam values(5, '010-5322-1321', 'dd@daum.net');

insert into regexp\_exam values(6, '019-3333-3333', 'test123@outook.kr');

insert into regexp\_exam values(7, '016-4631-2139', 'qwer@gmail.com');

insert into regexp\_exam values(8, '016-532-0100', 'oiuy@daum.net');

insert into regexp\_exam values(9, '051-3210-2011', 'ujnhy@outook.kr');

insert into regexp\_exam values(10, '010-4113-9300', 'ounae@daum.net');

insert into regexp\_exam values(11, '011-5131-8040', 'abc123@gmail.com');

insert into regexp\_exam values(12, '010-812-0012', 'my.email@gmail.com');

insert into regexp\_exam values(13, '011-912-2167', 'test.addr@daum.net');

commit;

select \* from regexp\_exam;

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

regexp\_like(조회문자열, 패턴 [, 옵션])

- like 연산자의 확장

- where 절에서 사용하며 패턴과 매칭 되는 문자열이 조회문자열안에 있는 지 확인.

- 옵션: c (대소문자 구분)-기본값default, i (대소문자구분안함)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

--전화번호가 011 또는 010인 행을 조회

select \* from regexp\_exam

where regexp\_like(phone, '^01(1|0)'); --^: 시작, 01 1 또는 0

--where regexp\_like(phone, '^01[01]'); -- 01로 시작, 0또는 1

--핸드폰 번호인 행을 조회 (010, 011, 016, 017, 018, 019)로 시작하는 번호

select \* from regexp\_exam

where regexp\_like(phone, '^01[016789]'); -- 01로 시작, 0|1|6|7|8|9 중 하나

-- 국번이 3자리인 행을 조회

select \* from regexp\_exam

where regexp\_like(phone, '-\d{3}-'); --\d: 숫자 1개 , {3}: 3개, 양쪽에 -(대시)가 들어옴

--전화번호의 국번(2번째)과 번호(3번째)가 같은 것

select \* from regexp\_exam

where regexp\_like(phone, '-(\d+)-\1'); --(\d+): 패턴내의 하위그룹. \d{3,4}: 숫자가 3개나 4개온다. \1: 1번 패턴의 값과 동일하다.

--이메일주소 계정 부분(@ 앞)에 . 이 없는 행들 조회

select \* from regexp\_exam

where not regexp\_like(email,'\w+\.\w+@'); --\w:문자(숫자,알파벳) 하나 +1글자이상

--1글자이상의문자.1글자이상의문자@

--메타문자 .: 임의의 1글자

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

regexp\_substr(대상, 패턴 [, 찾기시작index, 몇번째것, 옵션])

- substr() 함수의 확장

- 대상문자열에서 패턴과 일치하는 문자열을 반환한다.

- 찾기 시작 index: 패턴과 일치하는 문자열을 찾기 시작하는 글자 위치 (기본값 : 1)

- 몇번째것: 패턴과 일치하는 문자열이 여러개일때 몇번째 것인지 지정 (기본값 : 1)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

--이메일 주소의 @ 앞의 부분/뒷부분만 출력

select regexp\_substr(email, '[^@]+') "@앞" ,regexp\_substr(email, '[^@]+',1,2)"@뒤" --패턴과 일치하는 것 중 2번째 것.

from regexp\_exam;

select regexp\_substr(email, '[^@]\*') "@앞" ,regexp\_substr(email, '[^@]\*',1,2) "@뒤", regexp\_substr(email, '[^@]\*',1,3)

from regexp\_exam;

select email from regexp\_exam;

--전화번호의 첫번째부분(통신망/지역번호) 만 추출해서 출력

select regexp\_substr(phone, '\d+')"첫번째부분", regexp\_substr(phone, '\d+',1,2)"두번째부분",regexp\_substr(phone, '\d+',1,3)"세번째부분"

--전화번호 앞자리, 중간자리, 마지막자리 순

from regexp\_exam;

--국번만 추출해서 출력

select regexp\_substr(phone, '\d+',1,2)"두번째부분"

from regexp\_exam;

--개별번호만 추출해서 출력

select regexp\_substr(phone, '\d+',1,3)"세번째부분"

from regexp\_exam;

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

regexp\_replace(대상, 패턴, 변경할문자열 [, 찾기시작idex, 몇번째것, 옵션])

- replace() 함수의 확장

- 대상문자열에서 패턴과 일치하는 문자열을 반환한다.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

-- 전화번호의 국번을 \* 로 바꿔서 출력

select phone, regexp\_replace(phone, '\d+', '\*\*\*\*', 1, 2)

from regexp\_exam;

-- 전화번호의 번호(마지막 부분)을 \*\*\*\*로 바꿔서 출력

select phone, regexp\_replace(phone, '\d+', '\*\*\*\*', 1, 3)

from regexp\_exam;

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

regexp\_count(대상, 패턴 [, 시작index] [, 옵션])

- 대상에서 패턴이 나타난 횟수를 반환한다.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

-- 전화번호에서 숫자들의 개수를 조회

select regexp\_count(phone, '\d')

from regexp\_exam;