

sql 독학 강의 # 단일행 함수 잘 사용 하기(문자 함수) 4편 -sTricky

sTricky 2020. 3. 23. 11:19

sql 독학 강의 # 단일행 함수 잘 사용 하기(문자 함수) 4편 -sTricky



컨텐츠 index

0. SQL 함수의 정의

1. lower/upper 함수 사용 하기

2. length 함수 사용 하기
3. concat 함수 사용 하기
4. substr/mid 함수 사용 하기
5. instr 함수 사용 하기
6. lpad/rpad 함수 사용 하기
7. trim/ltrim/rtrim 함수 사용 하기
8. replace 함수 사용 하기

안녕하세요.

이번 SQL 독학 강의의 주제는 <단일행 함수 잘 사용 하기, 문자 함수>입니다.

함수만 잘 사용하더라도 SQL 활용능력을 많이 끌어올릴 수 있습니다.

이번 강의를 통해서 여러분들의 SQL 활용 능력이 많이 좋아지기를 기대하며, SQL 공부 강의를 시작해 보겠습니다.

#지난 독학 강의 보러 가기#

[2020/03/19 - \[Database/sql 강의\] - sql 공부 강의 # select를 잘 이용하는 방법\(2\), 3편 -sTricky](#)

	<p>sql 공부 강의 # select를 잘 이용하는 방법(2), 3편 -sTricky</p> <p>stricky.tistory.com</p>
--	---

0. SQL 함수의 정의

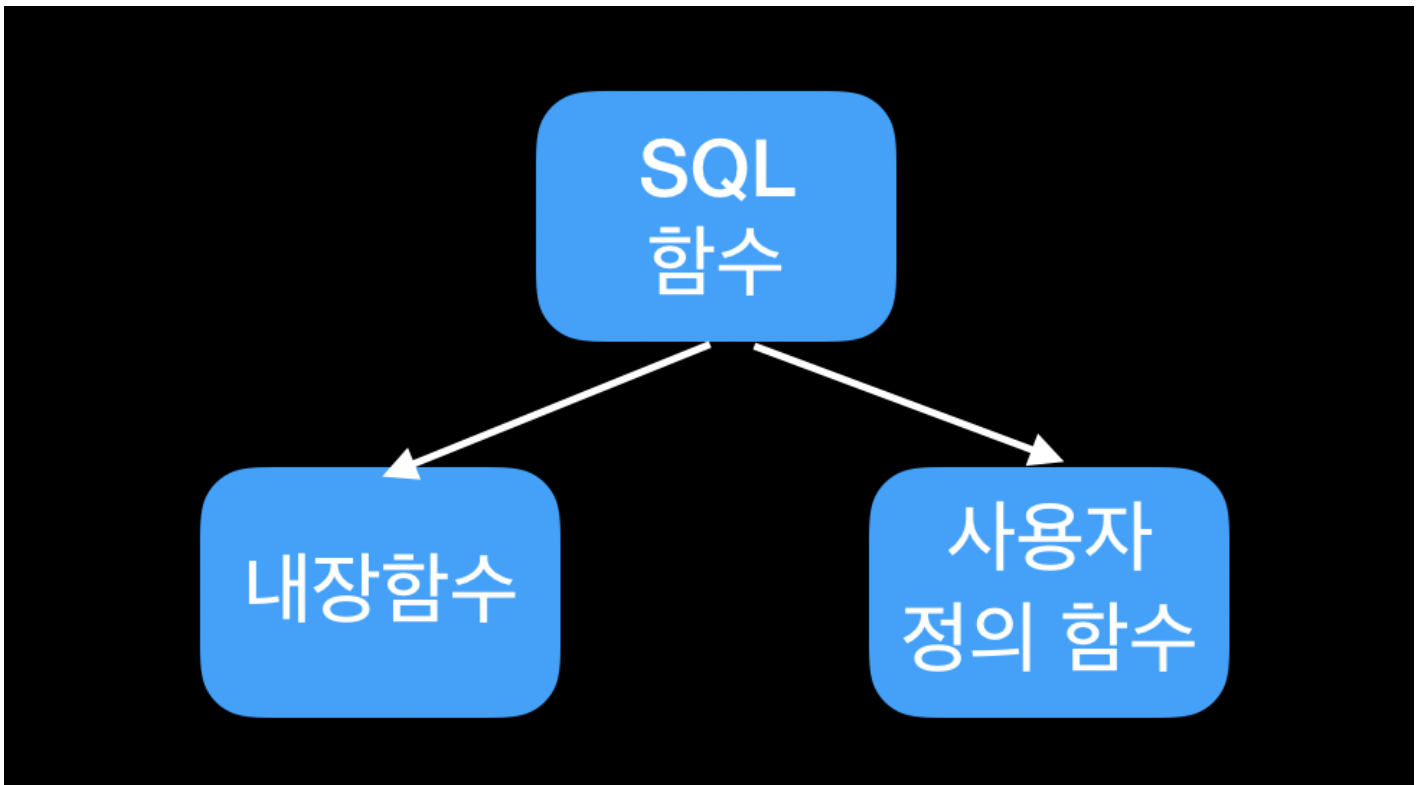
SQL은 여러 가지 함수를 이용해서 출력 값을 변환할 수 있습니다.

여기서 함수란 무엇을 말하는 것일까요? 함수란 어떤 값을 받아서 그 값을 어떠한 정해진 정의에 의해 변환시켜 변환된 값을 출력하는 것을 말합니다. 예를 들어서 우리가 평상시에 사용하는 자판기 역시 동전이라는 값을 받아서 미리 정의된 규칙에 의해 특정한 음료나 서비스를 출력하는 것처럼 SQL의 함수 역시 마찬가지로의 개념으로 이해하시면 됩니다.

DBMS에서 함수를 분류하는 기준이 몇 가지 있습니다.

우선은 **내장 함수**와 **사용자 정의 함수**로 나눌 수 있습니다. 내장 함수란 우리가 사용하는 각각의 RDBMS에 이미 내장된 함수를 뜻 합니다. 그리고 사용자 정의 함수란 내장 함수를 제외하고 'create function' 문을 사용해서 자신이 필요한 변환 규칙을 적용해 개개인의 유저 혹은 DBA, 개발자들이 만든 함수를 뜻 합니다.





그리고 다른 분류로는 **단일행 함수**, **복수행 함수**로 구분할 수 있습니다. 단일행 함수란 한 행(row)의 값을 받아서 특정 규칙과 정의를 통해 변환시키는 함수이고, 복수행 함수란 여러 행의 값을 한꺼번에 받아서 하나의 행(row)의 결과 값으로 되 돌려주는 함수를 뜻합니다. 이미 알고 계실지도 모르겠지만 가장 보편적인 복수행 함수로는 'count()'가 있습니다.

그리고 또 분류하자면 **문자 함수**와 **숫자 함수**, **날짜 함수**, **형 변환 함수**, **일반 함수** 등으로도 분류할 수 있습니다. 여기에 대해서는 천천히 알아보도록 하겠습니다.

이번 포스팅에서 공부할 내용은 내장 함수이자 단일행 함수이며 문자 함수에 속하는 내용에 대해서 공부하도록 하겠습니다.

#문자 함수 실습을 위한 데이터 입력

```
create table kmong.country
(
    country_name varchar(100),
    capital_city varchar(100),
    continent varchar(100)
) character set utf8;

INSERT INTO kmong.country (country_name, capital_city, continent) VALUES ('USA', 'Washington', 'America');
```

```

INSERT INTO kmong.country (country_name, capital_city, continent) VALUES ('England', 'London', 'Europe');
INSERT INTO kmong.country (country_name, capital_city, continent) VALUES ('S.Korea', 'Seoul', 'Asia');
INSERT INTO kmong.country (country_name, capital_city, continent) VALUES ('Australia', 'Canberra', 'Oceania');
INSERT INTO kmong.country (country_name, capital_city, continent) VALUES ('Ghana', 'Accra', 'Africa');
INSERT INTO kmong.country (country_name, capital_city, continent) VALUES ('Argentina', 'Buenos aires', 'America');

```

우선 위 데이터를 출력해 보겠습니다.

```
select * from kmong.country;
```

The screenshot shows a SQL IDE interface. At the top, the query `select * from kmong.country;` is entered in the editor. Below the editor, the 'Output' tab is active, displaying the results of the query. The results are shown in a table with 6 rows and 3 columns: `country_name`, `capital_city`, and `continent`. The data is as follows:

	country_name	capital_city	continent
1	USA	Washington	America
2	England	London	Europe
3	S.Korea	Seoul	Asia
4	Australia	Canberra	Oceania
5	Ghana	Accra	Africa
6	Argentina	Buenos aires	America

이 모양을 아래 문자 함수를 공부하면서 잘 기억해두고 비교하시기 바랍니다.

1. lower/upper 함수 사용 하기

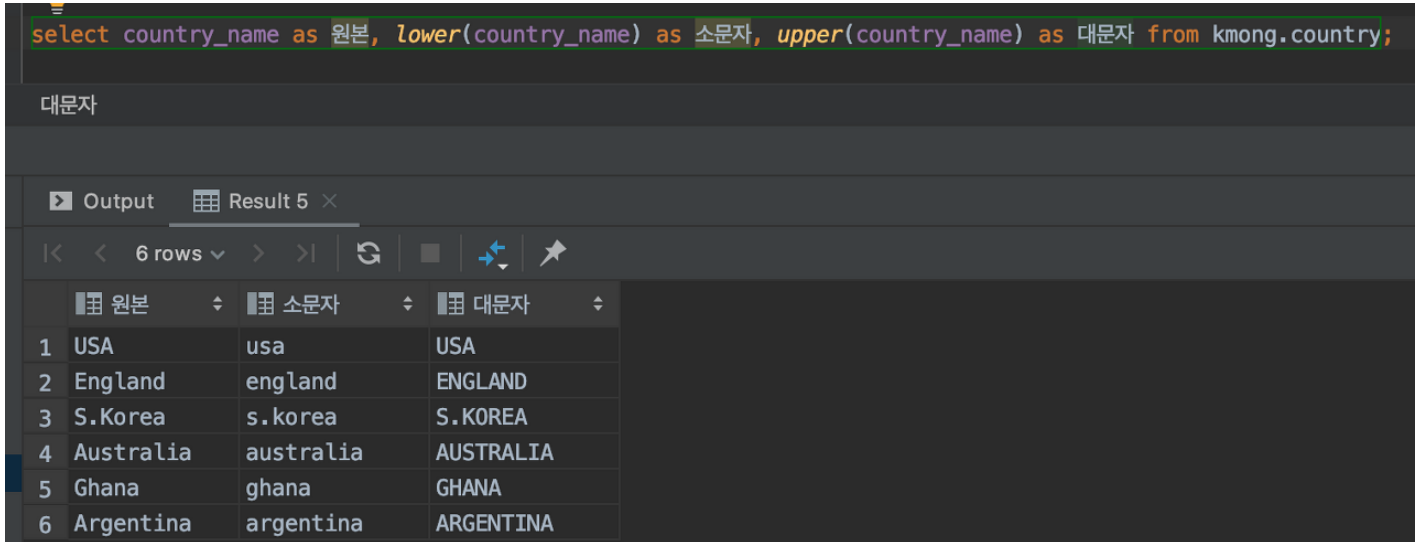
lower와 upper는 각 입력된 문자를 소문자 / 대문자로 변경시키는 함수입니다.

예를 들어서 lower 함수는 'KOREA'라는 문자를 'korea'로 변환하여 출력을 해주고, upper 함수는 반대가 되겠죠? 'korea'라는 문자를 'KOREA'로 변환하여 출력합니다.

사용법은 `lower(칼럼명)`, `upper(칼럼명)` 이렇게 사용하시면 됩니다.

아래 select 예문과 결과를 확인해보겠습니다.

```
select country_name as 원본, lower(country_name) as 소문자, upper(country_name) as 대문자
from kmong.country;
```



대문자

Output Result 5 ×

6 rows

	원본	소문자	대문자
1	USA	usa	USA
2	England	england	ENGLAND
3	S.Korea	s.korea	S.KOREA
4	Australia	australia	AUSTRALIA
5	Ghana	ghana	GHANA
6	Argentina	argentina	ARGENTINA

첫 번째 '원본'이 원본 데이터입니다. 두 번째 '소문자'가 lower 함수, 마지막 '대문자'가 upper 함수의 결과 출력 값을 나타내고 있습니다.

이처럼 SQL에서는 한 테이블의 한 칼럼을 다양하게 가공하여 여러 가지 모습으로 보이게 하는 것이 가능합니다.

함수 테스트를 할 때 유용하게 쓰일 수 있습니다.

2. length 함수 사용 하기

length 함수는 데이터의 길이를 세어 숫자로 리턴을 해주는 함수입니다. 거두절미하고 예제를 보면 이해가 되실 겁니다.

사용법은 select 절에 **length(칼럼명)** 이렇게 아래와 같이 사용하시면 됩니다.

```
select country_name, length(country_name) as 길이
from kmong.country;
```

```
select country_name, length(country_name) as 길이
from kmong.country;
```

Output

Result 6

×

⏪

⏴

6 rows

⏵

⏩

↺

■

↻

⚡

	<div><div></div>country_name</div>	<div><div></div>길이</div>
1	USA	3
2	England	7
3	S.Korea	7
4	Australia	9
5	Ghana	5
6	Argentina	9

'USA'는 세글자닌깐 3을 리턴했습니다.

그리고 'S.Korea'는 점까지 세면 모두 일곱 글 자이닌깐 7을 리턴했습니다.

간단하죠?

다음으로 넘어가겠습니다.

3. concat 함수 사용 하기

concat은 앞서 한번 이야기드렸던 내용 입니다만 문자 함수에 속하니 간단하게 설명하고 넘어가겠습니다.

오라클의 경우 '||' 연산자로 문자열이나 칼럼값을 붙여서 사용 합니다. 하지만 mysql에서는 그것을 지원하지 않고, concat 이라는 함수를 사용하여 문자나, 컬럼 값을 붙여서 출력시킬 수 있습니다.

간단한 사용법을 설명드리자면 `concat(칼럼 값, 칼럼 값, '문자열', '문자열')` 이런 식으로 칼럼 값 고 문자열을 원하는 데로 넣고 그 사이는 ','로 구분하여 작성하면 됩니다.

예문을 보겠습니다.

```
select concat(country_name, '의 수도는 ', capital_city, '입니다!') as 수도소개
from kmong.country;
```

```
select concat(country_name, '의 수도는 ', capital_city, '입니다!') as 수도소개
from kmong.country;
```

	수도소개
1	USA 의 수도는 Washington 입니다!
2	England 의 수도는 London 입니다!
3	S.Korea 의 수도는 Seoul 입니다!
4	Australia 의 수도는 Canberra 입니다!
5	Ghana 의 수도는 Accra 입니다!
6	Argentina 의 수도는 Buenos aires 입니다!

위 예문을 보고 눈치채셨는지 모르겠지만 칼럼명을 쓸때는 그냥 컬럼명을 쓰고 ','를 붙이면 됩니다. 하지만 문자열을 입력할 때는 양 옆에 '문자열' 이렇게 따옴표로 감싸 주어야 합니다.

concat으로 연결하여 '수도소개'라는 별칭을 단 하나의 칼럼으로 표현하는 방법을 배웠습니다.

다음으로 넘어가겠습니다.

4. substr/mid/substring 함수 사용 하기

substr과 mid, substring 함수는 똑같은 함수이다. 왜 똑같은 게 이렇게 있는지... 나도 잘 모르겠지만 아무튼 사용법이 똑같고 리턴해주는 값 역시 같습니다.

사용법은 셋다 동일합니다. **substr(칼럼명, 시작할 문자열의 위치 값, 리턴 시킬 값의 길이)** 잘 이해가 안 되시면 예문을 보겠습니다.

```
select continent as 원본, substr(continent,2,2) as substr, mid(continent,2,2) as mid, substring(continent,2,2) as substring
from kmong.country;
```

	원본	substr	mid	substring
1	America	me	me	me
2	Europe	ur	ur	ur
3	Asia	si	si	si
4	Oceania	ce	ce	ce
5	Africa	fr	fr	fr
6	America	me	me	me

substr, mid, substring, 세 가지 함수를 원본과 함께 똑같은 파라미터를 주고 실행을 한 결과입니다.

우선, 세 함수의 리턴 값이 같다는 것을 확인할 수 있습니다. 함수 안에 칼럼명이라던지, 숫자 등을 입력할 때 이것을 우리는 파라미터라고 부르는데, 똑같이 **(continent,2,2)**라고 파라미터를 입력했습니다.

우선 첫 번째 row에 있는 'America' 값을 보면서 설명드릴게요.

continent 칼럼 값인 'America'의 두 번째 글자인 'm'에서 2자리를 리턴하라는 파라미터 값입니다.

그러니깐 'me'라는 값이 리턴되고, 그 아래 'Europe'을 봐도 두 번째 글자인 'u'에서 두 글자를 리턴 하니까 'ur'이 리턴된 것입니다.

그럼 파라미터 값을 조금 바꿔서 실행해보겠습니다.

```
select continent as 원본, substr(continent,3,1) as substr, mid(continent,3,3) as mid, substring(continent,3,5) as substring
from kmong.country;
```

```
select continent as 원본, substr(continent,3,1) as substr, mid(continent,3,3) as mid, substring(continent,3,5) as substring
from kmong.country;
```

	원본	substr	mid	substring
1	America	e	eri	erica
2	Europe	r	rop	rope
3	Asia	i	ia	ia
4	Oceania	e	ean	eania
5	Africa	r	ric	rica
6	America	e	eri	erica

이번엔 substr, mid, substring, 세 가지 함수의 파라미터 값을 조금씩 바꾸어 봤습니다.

이번 결과를 보신 칸 조금 이해가 되시나요?

세 번째 row의 값인 'Asia'를 보겠습니다. 여기서 첫 번째 함수에서는 `substr(continent,3,1)`이라고 파라미터를 주니 세 번째 글자인 'i'에서 한자리 리턴, 'i'가 리턴되었고, 두 번째 함수인 `mid(continent,3,3)`이라고 실행하니, 'ia'만 나왔습니다. 분명 3자리를 리턴하라고 했는데..

맞습니다! 원문 자체가 짧기 때문에 그것만 리턴이 된 것이지요.

마지막에 있는 substring도 이해가 되시죠?

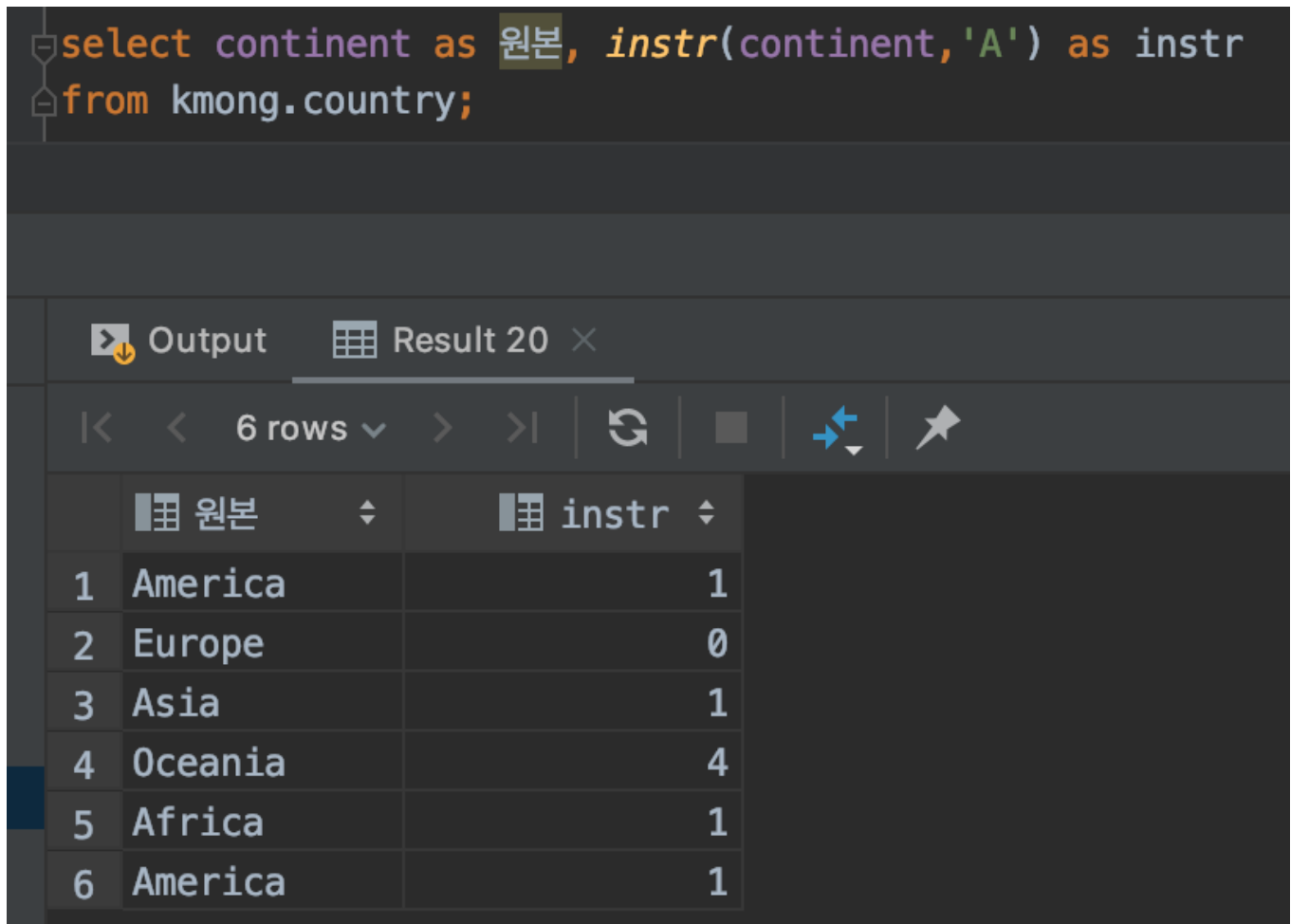
그럼 다음으로 넘어가겠습니다.

5. instr 함수 사용 하기

instr 함수는 특정 문자열의 위치를 숫자로 리턴해주는 함수입니다.

사용법은 **instr(칼럼 값, '찾는 문자')**입니다. 바로 예문을 보겠습니다.

```
select continent as 원본, instr(continent, 'A') as instr  
from kmong.country;
```



	원본	instr
1	America	1
2	Europe	0
3	Asia	1
4	Oceania	4
5	Africa	1
6	America	1

왼쪽 원본 데이터에서 'A'라는 문자가 어디에 있는지 찾아서 해당 위치를 숫자로 리턴해줍니다.

'America'를 보면 가장 앞에 'A'가 있습니다. 그러니깐 1이라는 숫자를 리턴했구요.

네 번째 row에 있는 'Oceania'를 보면 'A' 문자가 4번째 위치해 있으니 4를 리턴합니다.

여기서 눈치채셨겠지만, instr는 대소문자를 구별하지 않습니다.

다음으로 넘어가겠습니다.

6. lpad/rpad 함수 사용 하기

lpad와 rpad는 간단하게 설명하자면 데이터가 있고, 해당 데이터가 어떤 기준보다 짧을 경우에 원하는 문자를 왼쪽이나 오른쪽으로 자릿수를 맞춰 채워 주는 함수입니다.

사용법은 lpad와 rpad 모두 동일합니다. **lpad(칼럼명, 기준 자릿수, 채워 넣을 숫자 or 문자)**입니다.

바로 예문을 확인해 보겠습니다.

```
select continent as 원본, lpad(continent,10,'A') as lpad, rpad(continent,10,'A') as rpad
from kmong.country;
```

	원본	lpad	rpadd
1	America	AAAAmerica	AmericaAAA
2	Europe	AAAAEurope	EuropeAAAA
3	Asia	AAAAAAAsia	AsiaAAAAAA
4	Oceania	AAA0ceania	OceaniaAAA
5	Africa	AAAAAfrica	AfricaAAAA
6	America	AAAAmerica	AmericaAAA

원본 데이터와 lpad, rpad 각각의 출력 결과 값입니다. **기준 자릿수를 둘 다 10으로 주었으니 10자리의 문자열을 만들고, lpad는 왼쪽, rpad는 오른쪽에 파라미터로 지정한 'A'라는 문자열을 칼럼 값이 들어가고 남은 빈자리에 채워 넣었습니다.**

데이터를 보닌 간 한 번에 이해가 되시죠?

다음으로 넘어가겠습니다.

7. trim/ltrim/rtrim 함수 사용 하기

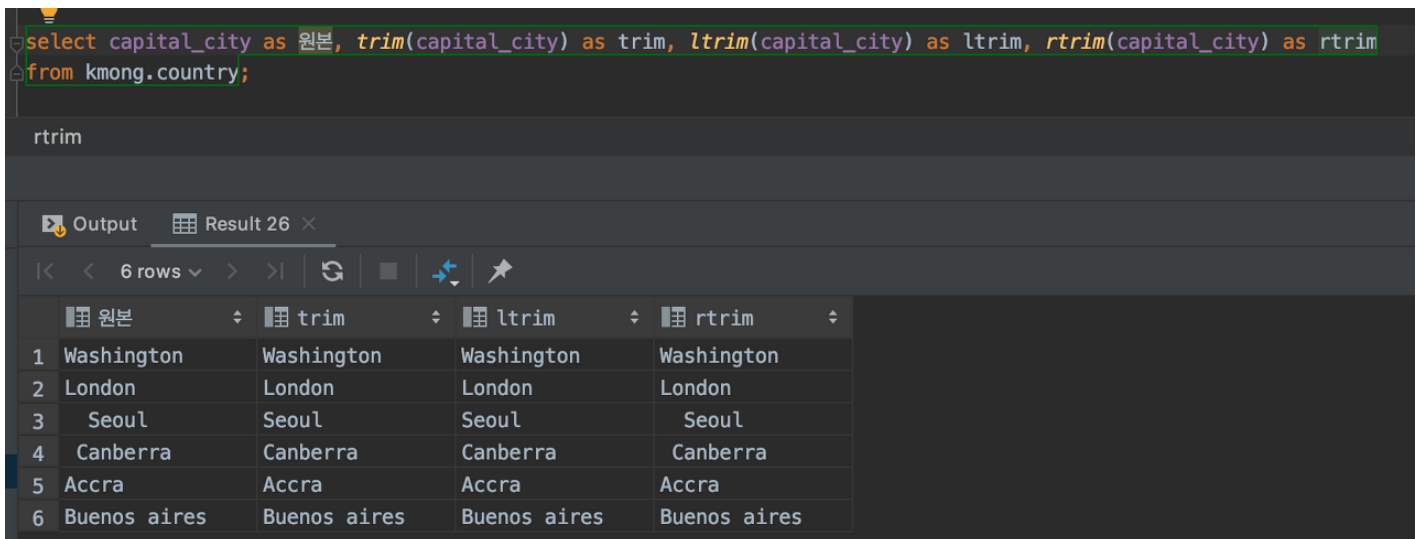
trim, ltrim과 rtrim 역시 비슷한 기능을 가진 함수들입니다.

trim은 어떤 문자열의 양쪽, 즉 왼쪽, 오른쪽의 공백을 없애는 함수이고, ltrim과 rtrim은 각각 왼쪽과 오른쪽 공백만 없애는 함수들입니다.

사용법은 세 함수 모두 같습니다. trim(칼럼명)입니다.

예문을 보시면 이해가 빠를 것 같습니다.

```
select capital_city as 원본, trim(capital_city) as trim, ltrim(capital_city) as ltrim, rtrim(capital_city) as rtrim
from kmong.country;
```



	원본	trim	ltrim	rtrim
1	Washington	Washington	Washington	Washington
2	London	London	London	London
3	Seoul	Seoul	Seoul	Seoul
4	Canberra	Canberra	Canberra	Canberra
5	Accra	Accra	Accra	Accra
6	Buenos aires	Buenos aires	Buenos aires	Buenos aires

위 사진을 보시면 원본, trim, ltrim, rtrim 각각의 결과 출력 값들입니다. 원본을 보시면 왼쪽 공백은 눈에 잘 보이나, 오른쪽 공백은 잘 안 보이는 점은 가만해 주시기 바랍니다.

trim 함수의 결괏값에선 양옆 모든 공백을 지웠고요. ltrim과 rtrim에서는 원본 데이터의 각각 왼쪽과 오른쪽 공백만 지운 것을 확인할 수 있습니다.

다음으로 넘어가겠습니다.

8. replace 함수 사용 하기

replace 함수는 특정 문자열을 찾아서 다른 문자열로 치환하는 함수입니다.

사용법은 **replace(칼럼명, '찾을 문자', '치환할 문자')**입니다.

예문을 보면서 설명드리겠습니다.

```
select continent as 원본, replace(continent, 'A', '@') as 'replace'
from kmong.country;
```

```
select continent as 원본, replace(continent, 'A', '@') as 'replace'
from kmong.country;
```

	원본	`replace`
1	America	@merica
2	Europe	Europe
3	Asia	@sia
4	Oceania	Oceania
5	Africa	@frica
6	America	@merica

위에서 쓴 replace 함수의 파라미터를 보면 **continent** 칼럼에서 'A' 값을 찾아 '@'로 치환하여 출력하라는 명령을 준 것을 확인할 수 있습니다.

아래 결과가 나온 것을 보니 잘 나오 것 같은데 위에서 우리가 공부한 instr 함수와는 다르게 대소문자 구분을 한다는 게 특징입니다. 이 부분을 유념해서 사용하시길 바라겠습니다.

자, 여기까지 일단 준비한 내용들은 공부를 했습니다.

이것 말고도 mysql에서 지원하는 문자 함수는 많이 있습니다.

저는 그중에서 특별히 사용을 많이 하는 것을 추려서 수업 강의 자료로 사용을 했고요. 나머지 문자 함수들에 대해서 더 많이 알고 싶으신 분들은 mysql에서 공식 문서로 제공하는 문자 함수 문서를 참조하시면 될 것 같습니다.

링크는 아래쪽입니다.

<https://dev.mysql.com/doc/refman/8.0/en/string-functions.html>

	MySQL :: MySQL 8.0 Reference Manual :: 12.7 String Functions... dev.mysql.com
--	--

이만 오늘 강의는 여기서 마치도록 하겠습니다.

여기까지 봐주셔서 너무 감사합니다!

다음에 또 뵙겠습니다.

