

# database-10

☰ 태그	
📅 날짜	@2023년 6월 13일

계산 용도의 목적으로 RDMBS 내부에 function 형태로 저장해서 쓸 수 있는 SQL 문법을 설명

## 내용

- stored function의 개념과 여러 예제
- stored function 삭제하기
- 이미 저장되어 있는 stored function 파악하기
- stored function은 언제 써야할까?

## stored function

- 사용자가 정의한 함수
- DBMS에 저장되고 사용되는 함수
- SQL의 select, insert, update, delete statement에서 사용할 수 있다.

```
stored function
- 임직원의 id를 열자리 정수로 랜덤하게 발급하고 싶다.
- ID의 맨앞자리는 1로 고정이다.

delimiter $$
CREATE FUNCTION id_generator()
  RETURNS int
  NO SQL
  BEGIN
    RETURN (1000000000 + floor(rand() * 1000000000));
  END
```

```
$$  
delimiter;
```

기본적으로 sql에서 사용되는 delimiter는 세미콜론(;)이다.  
function 내부에서 어떤일을 할지 정의할때 ;을 사용하게 되는데 만약 delimiter를 바꿔주지않으면 함수 생성을 하는 것이 중간에 끝나는걸로 인식을 하게 된다.  
그렇기 때문에 delimiter를 바꿔주어야 한다.

```
INSERT INTO employee  
VALUES (id_generator(), 'JEHN', '1991-08-04', 'F', 'PO', 1000000000, 1005);
```

```
mysql> SELECT * FROM employee WHERE name = 'JEHN';  
+-----+-----+-----+-----+-----+-----+-----+  
| id      | name  | birth_date | sex | position | salary      | dept_id |  
+-----+-----+-----+-----+-----+-----+-----+  
| 1961029958 | JEHN  | 1991-08-04 | F   | PO       | 1000000000  | 1005    |  
+-----+-----+-----+-----+-----+-----+-----+
```

부서의 id를 파라미터로 받으면 해당 부서의 평균 연봉을 알려주는 함수를 작성하자

```
delimiter $$  
CREATE FUNCTION dept_avg_salary(d_id int)  
RETURN int // return type  
READS SQL DATA  
BEGIN  
    DECLARE avg_sal int; // 변수선언 declare 변수명 타입  
    select avg(salary) into avg_sal  
        from employee  
        where dept_id = d_id;  
    RETURN avg_sal;  
END  
$$  
delimiter;
```

- 이외에도 loop를 돌면서 반복적인 작업을 수행
- case 키워드를 사용해서 값에 따라 분기 처리 하거나 에러를 핸들링하거나 에러를 일으키는

- 등의 다양한 동작을 정의

```
stored function 삭제하기
- DROP FUNCTION stored_function_name;
```

등록된 stored function 파악하기

```
SHOW FUNCTION STATUS where DB = 'company';
```

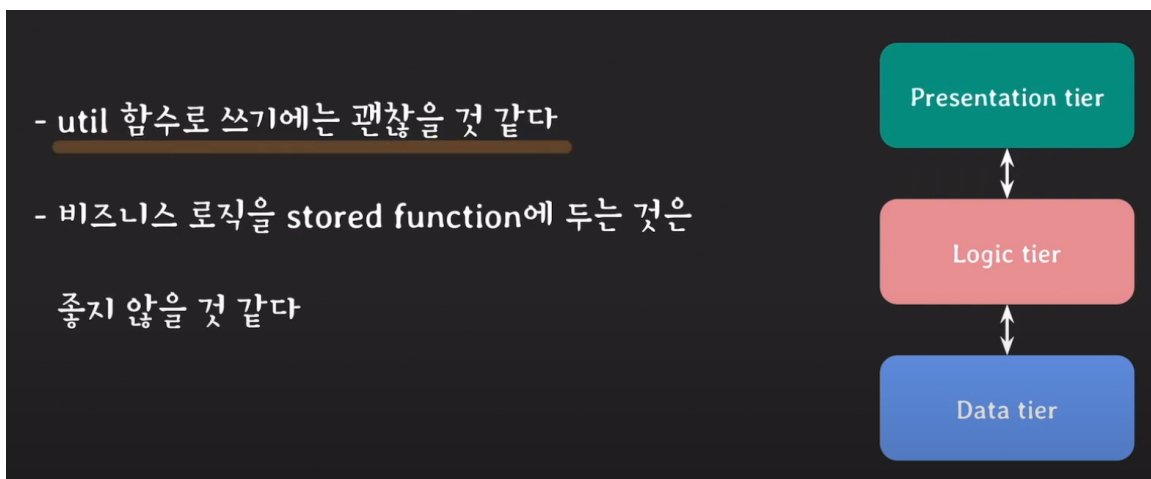
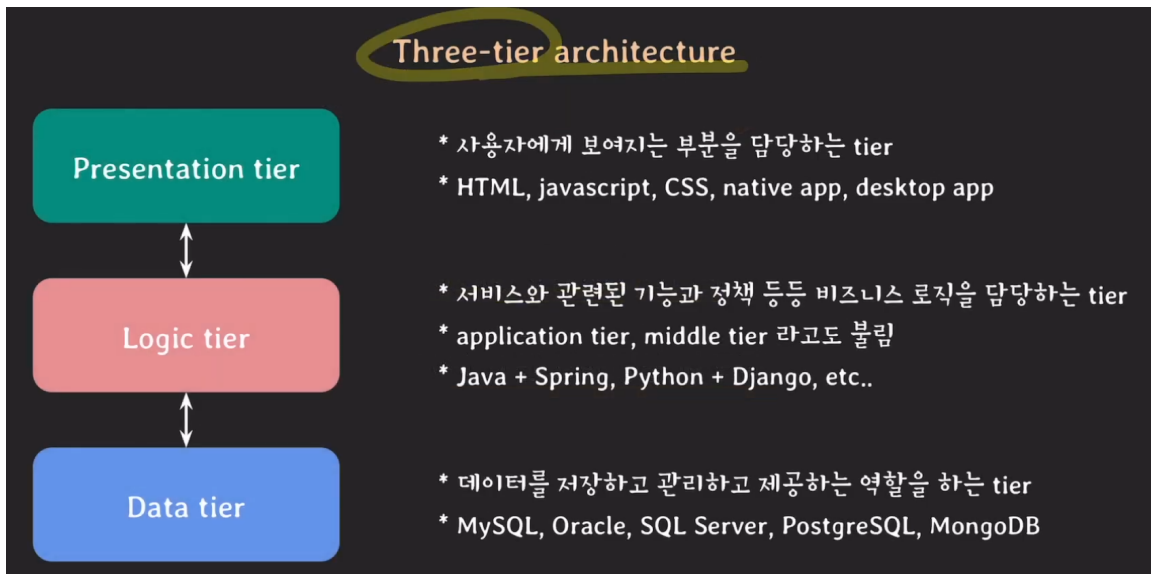
Db	Name	...
company	dept_avg_salary	...
company	id_generator	...
company	toeic_pass_fail	...

코드 확인

```
SHOW CREATE FUNCTION id_generator;
```

```
+-----+-----+
| ... | Create Function | ... |
+-----+-----+
| ... | CREATE DEFINER=`root`@`localhost` FUNCTION `id_generator`()
RETURNS int NO SQL begin return (1000000000 + floor(rand() * 1000000000));
end | ... |
+-----+-----+
```

stored function은 언제 써야할까



## stored procedure

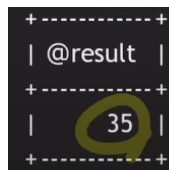
- 사용자가 정의한 프로시저
- RDBMS에 저장되고 사용되는 프로시저
- 구체적인 하나의 태스크(task)를 수행

두 정수의 곱셈 결과를 가져오는 프로시저를 작성하자

```
delimiter $$                                // 파라미터
CREATE PROCEDURE product(IN a int, IN b int, OUT result int)
BEGIN
    SET result = a * b;
END
$$
delimiter

call product(5,7, @result);
select @result;

// IN = input OUT = output , in out 키워드를 적어주지 않으면 in으로 인식
// out 키워드는 생략하면 안된다.
```

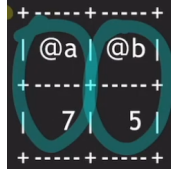


두 정수를 맞바꾸는 프로시저를 작성하자

```
delimiter $$
CREATE PROCEDURE swap(INOUT a int, INOUT b int)
BEGIN
    set @temp = a;
    set a = b;
    set b = @temp;
END
$$
delimiter;

set @a = 5, @b = 7;
call swap(@a, @b);
select @a, @b;

// inout 키워드를 통해 a와 b는 값을 전달받을 수 있으면서도 동시에 값을 바꿔서 저장할 수 있는 것
```



각 부서별 평균 연봉을 가져오는 프로시저를 작성

```
delimiter $$
CREATE PROCEDURE get_dept_avg_salary()
BEGIN
    select dept_id, avg(salary)
    from employee
    group by dept_id;
END
$$
delimiter;

call get_dept_avg_salary();
```

dept_id	avg(salary)
NULL	125000000.0000
1001	120000000.0000
1002	101000000.0000
1003	82500000.0000
1004	85000000.0000
1005	87500000.0000

6 rows in set (0.00 sec)

- 이외에도 조건문을 통해 분기처리를 하거나 반복문을 수행하거나 에러를 핸들링하거나 에러를 일으키는 등의 다양한 로직을 정의할 수 있다.

## stored procedure 와 stored function 의 차이

	STORED PROCEDURE	STORED FUNCTION
create 문법	CREATE PROCEDURE ...	CREATE FUNCTION ...
return 키워드로 값 반환	불가능 (SQL server는 상태코드 반환용으로만 사용 가능)	가능 (MySQL, SQL server는 값 반환하려면 필수)
파라미터로 값(들) 반환	가능 (값(들)을 반환하려면 필수)	일부 가능 (oracle 가능하나 권장 안함, postgresSQL 가능)
값을 꼭 반환해야 하나?	필수 아님	필수
SQL statement에서 호출	불가능	가능
transaction 사용	가능	대부분 불가능 (oracle의 경우 가능)
주된 사용 목적	business logic	computation

## 이외에도

- 다른 function/procedure를 호출할 수 있는지
- resultset(= table)을 반환할 수 있는지
- precompiled execution plan을 만드는지
- try-catch를 사용할 수 있는지