

## ★ 조인(JOIN)과 SQL 프로그래밍 ★

### - Stored Procedure, IF ~ ELSE ~ END IF, CASE 문, WHILE 문 -

#### ■ 저장 프로시저(Stored Procedure)

☞ 쿼리의 집합으로서 어떠한 동작을 일괄 처리하는 데 사용

- MySQL에서 제공하는 프로그래밍 기능
- 자주 사용되는 일반적인 쿼리를 하나하나 실행하는 것이 아니라 모듈화하여 필요할 때마다 호출하기 때문에 MySQL을 한층 더 편리하게 운영할 수 있음
- 작성 형식

형식
<pre> DELIMITER \$\$ CREATE PROCEDURE 스토어드프로시저이름( ) BEGIN     -- 이곳에 SQL 프로그래밍 코딩 END \$\$ DELIMITER ; CALL 스토어드프로시저이름( );         </pre>

#### ■ IF ~ ELSE ~ END IF 문

☞ 조건에 따라 분기하는 명령

형식
<pre> DELIMITER \$\$ IF &lt;불 표현식&gt; THEN     -- SQL문장들1 ... ELSE     -- SQL문장들2 ... END IF;         </pre>

- 명령을 실행한 결과 한 문장 이상 처리해야 할 때는 BEGIN ... END 문으로 묶어야 하는데, 실행할 문장을 습관적으로 BEGIN ... END 문으로 묶는 것이 좋음.

```

1 • USE cookDB;
2 -- 기존에 스토어드 프로시저를 만든 적이 있다면 삭제 DELIMITER $$
3 • DROP PROCEDURE IF EXISTS ifProc;
4
5 DELIMITER $$
6 • CREATE PROCEDURE ifProc()
7 BEGIN
8     DECLARE var1 INT; -- var1 변수 선언
9     SET var1 = 100; -- 변수에 값 대입
10
11 IF var1 = 100 THEN -- 만약 @var1이 100이라면
12     SELECT '100입니다.';
13 ELSE
14     SELECT '100이 아닙니다.';
15 END IF;
16 END $$
17
18 DELIMITER ;
19 • CALL ifProc();
    
```

100입니다.
100입니다.

```

MySQL 5.7 Command Line Client - Unicode
mysql> USE employees;
Database changed
mysql>
mysql> show tables;
+-----+
| Tables_in_employees |
+-----+
| departments          |
| dept_emp              |
| dept_manager          |
| employees             |
| salaries              |
| titles                |
+-----+
6 rows in set (0.00 sec)

mysql> desc employees;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| emp_no     | int(11)       | NO   | PRI | NULL    |       |
| birth_date | date          | NO   |     | NULL    |       |
| first_name  | varchar(14)   | NO   |     | NULL    |       |
| last_name   | varchar(16)   | NO   |     | NULL    |       |
| gender      | enum('M','F') | NO   |     | NULL    |       |
| hire_date   | date          | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.02 sec)

mysql>

```

- 직원 번호가 10001번인 직원의 입사일이 5년이 넘었는지 확인

```

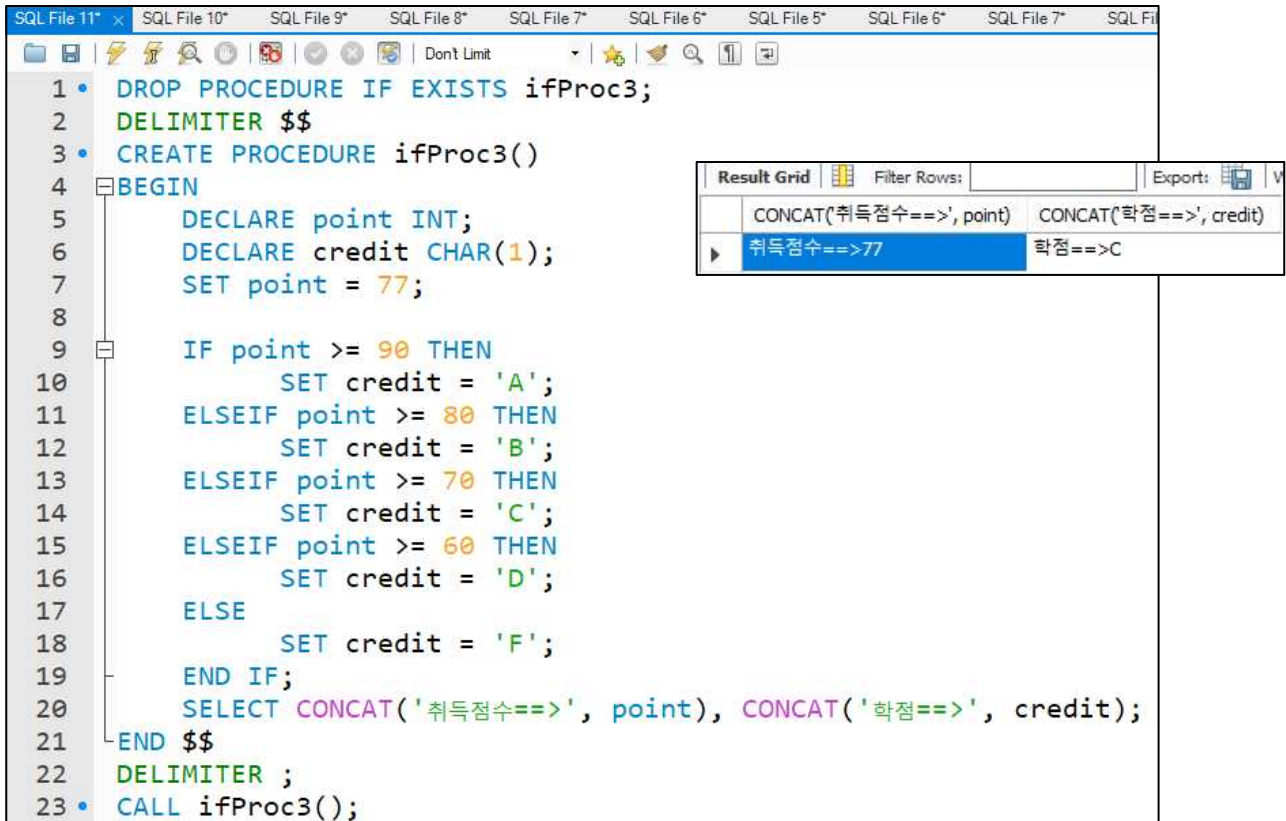
SQL File 9*  SQL File 10*  SQL File 8*  SQL File 7*  SQL File 6*  SQL File 5*  SQL File 6*  SQL File 7*  SQL File 8*
1  -- 기존에 스토어드 프로시저를 만든 적이 있다면 삭제 DELIMITER $$
2  • DROP PROCEDURE IF EXISTS ifProc2;
3  • USE employees;
4
5  DELIMITER $$
6  • CREATE PROCEDURE ifProc2()
7  BEGIN
8      DECLARE hireDATE DATE; -- 입사일
9      DECLARE curDATE DATE; -- 오늘
10     DECLARE days INT; -- 근무한 일수
11
12     SELECT hire_date INTO hireDATE -- hire_date 열의 결과를 hireDATE에 대입
13     FROM employees.employees
14     WHERE emp_no = 10001;
15
16     SET curDATE = CURRENT_DATE(); -- 현재 날짜
17     SET days = DATEDIFF(curDATE, hireDATE); -- 날짜의 차이, 일 단위
18
19     IF (days/365) >= 5 THEN -- 5년이 지났다면
20         SELECT CONCAT('입사한지 ', days, '일이나 지났습니다. 축하합니다!') AS '메시지';
21     ELSE
22         SELECT '입사한지 ' + days + '일밖에 안되었네요. 열심히 일하세요.' AS '메시지';
23     END IF;
24 END $$
25 DELIMITER ;
26 • CALL ifProc2();

```

Result Grid | Filter Rows:

메시지
입사한지 12564일이나 지났습니다. 축하합니다!

- 다중 분기를 IF문으로 작성



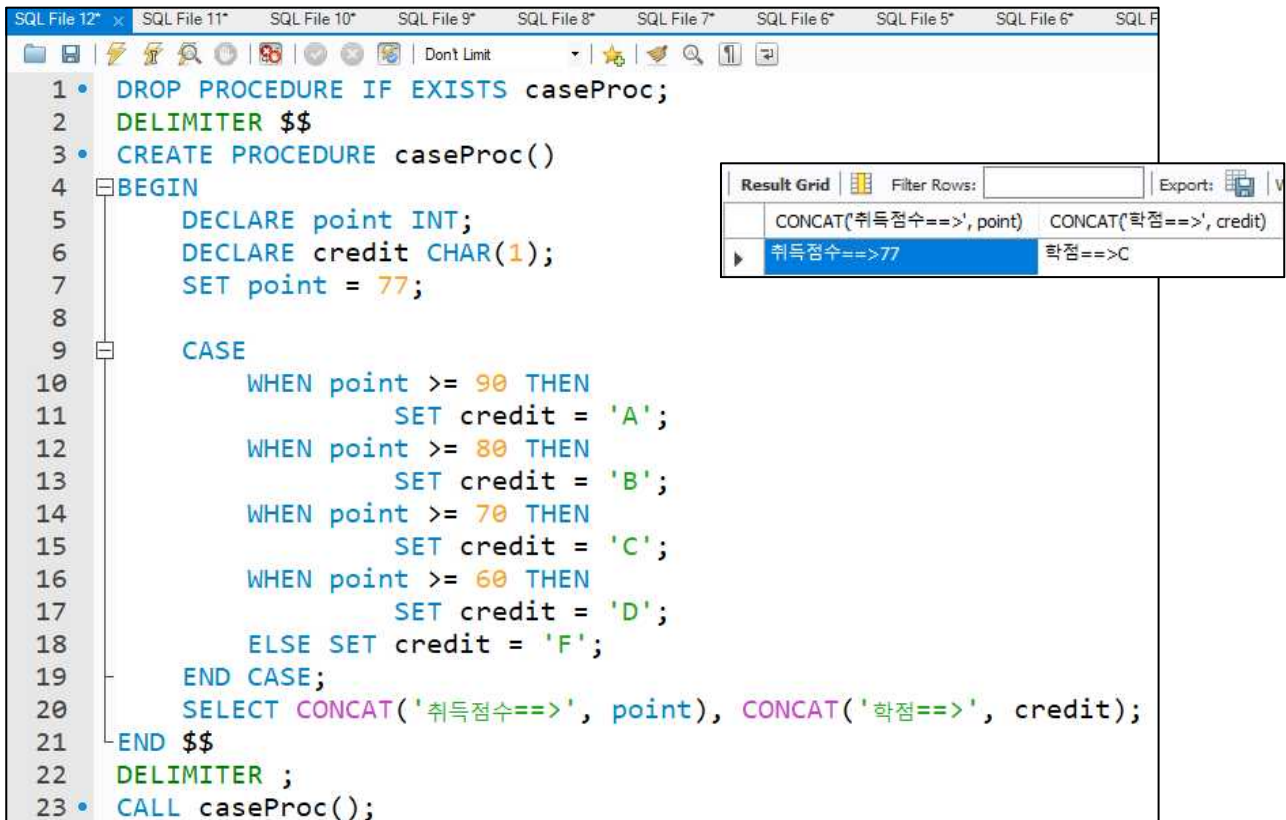
```

1 • DROP PROCEDURE IF EXISTS ifProc3;
2 • DELIMITER $$
3 • CREATE PROCEDURE ifProc3()
4 • BEGIN
5 •     DECLARE point INT;
6 •     DECLARE credit CHAR(1);
7 •     SET point = 77;
8 •
9 •     IF point >= 90 THEN
10 •         SET credit = 'A';
11 •     ELSEIF point >= 80 THEN
12 •         SET credit = 'B';
13 •     ELSEIF point >= 70 THEN
14 •         SET credit = 'C';
15 •     ELSEIF point >= 60 THEN
16 •         SET credit = 'D';
17 •     ELSE
18 •         SET credit = 'F';
19 •     END IF;
20 •     SELECT CONCAT('취득점수==>', point), CONCAT('학점==>', credit);
21 • END $$
22 • DELIMITER ;
23 • CALL ifProc3();
  
```

Result Grid	Filter Rows:	Export:
CONCAT('취득점수==>', point)	CONCAT('학점==>', credit)	
취득점수==>77	학점==>C	

## ■ CASE 문

- IF문을 CASE 문으로 변경



```

1 • DROP PROCEDURE IF EXISTS caseProc;
2 • DELIMITER $$
3 • CREATE PROCEDURE caseProc()
4 • BEGIN
5 •     DECLARE point INT;
6 •     DECLARE credit CHAR(1);
7 •     SET point = 77;
8 •
9 •     CASE
10 •         WHEN point >= 90 THEN
11 •             SET credit = 'A';
12 •         WHEN point >= 80 THEN
13 •             SET credit = 'B';
14 •         WHEN point >= 70 THEN
15 •             SET credit = 'C';
16 •         WHEN point >= 60 THEN
17 •             SET credit = 'D';
18 •         ELSE SET credit = 'F';
19 •     END CASE;
20 •     SELECT CONCAT('취득점수==>', point), CONCAT('학점==>', credit);
21 • END $$
22 • DELIMITER ;
23 • CALL caseProc();
  
```

Result Grid	Filter Rows:	Export:
CONCAT('취득점수==>', point)	CONCAT('학점==>', credit)	
취득점수==>77	학점==>C	

## ■ WHILE 문, ITERATE / LEAVE 문

- ☞ 다른 프로그래밍 언어의 WHILE 문과 동일한 개념
- ☞ <불식>이 참인 동안 WHILE 문 내의 명령을 계속 반복

### 형식

```
WHILE <불식> DO
    -- SQL 명령문들
END WHILE;
```

- 1부터 100까지의 값을 모두 더하는 코드

```
1 • DROP PROCEDURE IF EXISTS whileProc;
2   DELIMITER $$
3 • CREATE PROCEDURE whileProc()
4   BEGIN
5       DECLARE i INT; -- 1부터 100까지 증가할 변수
6       DECLARE hap INT; -- 더한 값을 누적할 변수
7       SET i = 1;
8       SET hap = 0;
9
10      WHILE (i <= 100) DO
11          SET hap = hap + i; -- hap의 원래 값에 i를 더하여 hap에 넣으라는 의미
12          SET i = i + 1; -- i의 원래 값에 1을 더하여 i에 넣으라는 의미
13      END WHILE;
14
15      SELECT hap;
16  END $$
17  DELIMITER ;
18 • CALL whileProc();
```

- 1~100 중 7의 배수를 합계에서 제외 (ITERATE 문과 LEAVE 문 사용)

```
1 • DROP PROCEDURE IF EXISTS whileProc2;
2   DELIMITER $$
3 • CREATE PROCEDURE whileProc2()
4   BEGIN
5       DECLARE i INT; -- 1부터 100까지 증가할 변수
6       DECLARE hap INT; -- 더한 값을 누적할 변수
7       SET i = 1;
8       SET hap = 0;
9
10      myWhile: WHILE (i <= 100) DO -- While 문에 label을 지정
11          IF (i%7 = 0) THEN
12              SET i = i + 1;
13              ITERATE myWhile; -- 지정한 label 문으로 가서 계속 진행
14          END IF;
15
16          SET hap = hap + i;
17          IF (hap > 1000) THEN
18              LEAVE myWhile; -- 지정한 label 문을 떠남(While 종료)
19          END IF;
20          SET i = i + 1;
21      END WHILE;
22
23      SELECT hap;
24  END $$
25  DELIMITER ;
26 • CALL whileProc2();
```



## ★ SQL 프로그래밍 - 문제해결형 미션 1 ★

## ■ 데이터베이스(cookDB), 구매 테이블(buyTBL)

MySQL 5.7 Command Line Client - Unicode

```
mysql> USE cookDB;
Database changed
mysql>
mysql> SELECT * from buyTBL;
```

num	userID	prodName	groupName	price	amount
1	KHD	운동화	NULL	30	2
2	KHD	노트북	전자	1000	1
3	KYM	모니터	전자	200	1
4	PSH	모니터	전자	200	5
5	KHD	청바지	의류	50	3
6	PSH	메모리	전자	80	10
7	KJD	책	서적	15	5
8	LHJ	책	서적	15	2
9	LHJ	청바지	의류	50	1
10	PSH	운동화	NULL	30	2
11	LHJ	책	서적	15	1
12	PSH	운동화	NULL	30	2

12 rows in set (0.00 sec)

## ■ 데이터베이스(cookDB), 회원 테이블(userTBL)

MySQL 5.7 Command Line Client - Unicode

```
mysql> USE cookDB;
Database changed
mysql>
mysql> SELECT * FROM userTBL;
```

userID	userName	birthYear	addr	mobile1	mobile2	height	mDate
KHD	강호동	1970	경북	011	22222222	182	2007-07-07
KJD	김제동	1974	경남	NULL	NULL	173	2013-03-03
KKJ	김국진	1965	서울	019	33333333	171	2009-09-09
KYM	김용만	1967	서울	010	44444444	177	2015-05-05
LHJ	이회재	1972	경기	011	88888888	180	2006-04-04
LKK	이경규	1960	경남	018	99999999	170	2004-12-12
NHS	남희석	1971	충남	016	66666666	180	2017-04-04
PSH	박수홍	1970	서울	010	00000000	183	2012-05-05
SDY	신동엽	1971	경기	NULL	NULL	176	2008-10-10
YJS	유재석	1972	서울	010	11111111	178	2008-08-08

10 rows in set (0.00 sec)

1. CASE 구문을 활용하여 아래와 같은 결과가 나오도록 SQL문을 작성하시오.

- 총구매액이 1500이 넘으면 '최우수고객', 1000이 넘으면 '우수고객'
- 총구매액이 1보다 크면 '일반고객', NULL이면 '유령고객'으로 출력하시오.

userID	userName	총구매액	고객등급
PSH	박수홍	1920	최우수고객
KHD	강호동	1210	우수고객
KYM	김용만	200	일반고객
LHJ	이회재	95	일반고객
KJD	김제동	75	일반고객
KKJ	김국진	NULL	유령고객
NHS	남희석	NULL	유령고객
SDY	신동엽	NULL	유령고객
LKK	이경규	NULL	유령고객
YJS	유재석	NULL	유령고객

```
1 • USE cookDB;
2 • SELECT U.userID, U.userName, SUM( ) AS '총구매액',
3     CASE
4         [ ]
5         [ ]
6         [ ]
7         ELSE '유령고객'
8     END AS '고객등급'
9 FROM buyTBL B
10     [ ] userTBL U
11     ON B.userID = U.userID
12 GROUP BY U.userID, U.userName
13 ORDER BY sum( ) DESC;
```

2. 숫자를 음수와 양수로 구분하는 SQL 프로그램의 일부이다. 빈칸을 채우시오.

<pre> 1 • USE cookDB; 2 • DROP PROCEDURE IF EXISTS testProc; 3 4  DELIMITER \$\$ 5 • CREATE PROCEDURE testProc() 6 • BEGIN 7     [ ] myData SMALLINT; 8     [ ] myData = -1000; 9 10    [ ] myData &lt; 0 [ ] 11        SELECT '음수입니다.'; 12    [ ] 13        SELECT '양수입니다.'; 14    [ ]; 15 END \$\$ 16 DELIMITER ; 17 • CALL testProc();         </pre>	<table border="1"> <tr> <th colspan="2">Result Grid</th> </tr> <tr> <td></td> <td>음수입니다.</td> </tr> <tr> <td>▶</td> <td>음수입니다.</td> </tr> </table>	Result Grid			음수입니다.	▶	음수입니다.
Result Grid							
	음수입니다.						
▶	음수입니다.						

3. WHILE 문을 사용하여 12345부터 67890까지의 숫자 중에서 1234의 배수 합계를 출력하는 SQL 프로그램을 작성하시오

<pre> 1 • USE cookDB; 2 • DROP PROCEDURE IF EXISTS whileProc; 3 4  DELIMITER \$\$ 5 • CREATE PROCEDURE whileProc( ) 6 • BEGIN 7     DECLARE i INT; 8     DECLARE hap INT; 9     SET i = 12345; 10    SET hap = 0; 11 12    WHILE (i &lt;= 67890) DO 13        IF [ ] THEN 14            SET hap = hap + i; 15        END IF; 16        SET i = [ ]; 17    END WHILE; 18 19    SELECT hap; 20 END \$\$ 21 DELIMITER ; 22 • CALL whileProc( );         </pre>	<table border="1"> <tr> <td></td> <td>hap</td> </tr> <tr> <td>▶</td> <td>1832490</td> </tr> </table>		hap	▶	1832490
	hap				
▶	1832490				