

1 REM Author : Henry
2 REM Date : 2024.06.12
3 REM Objective :
4 REM Environment : Ubuntu Server 22.04 LTS, MySQL Workbench 8.0 CE, MySQL Community
Server 8.0.37-0ubuntu0.22.04.3 (ubuntu)

5
6 REM SQL

7 1. 장점

- 8 1) SQL 질의문 하나로 원하는 데이터를 검색 및 조작할 수 있다.
- 9 2) 사용자가 이해하기 쉬운 단어로 구성
- 10 3) 복잡한 로직을 간단하게 작성할 수 있다.
- 11 4) ANSI 에 의해 문법이 표준화되어 있다.

12
13 2. 단점

- 14 1) 반복처리를 할 수 없다(LOOP)
- 15 2) 비교처리를 할 수 없다(IF).
- 16 3) Error 처리를 할 수 없다(EXCEPTION).
- 17 4) SQL 문을 캡슐화할 수 없다.
- 18 5) 변수선언을 할 수 없다.
- 19 6) 실행할 때마다 분석작업 후 실행한다.
- 20 7) Network Traffic 을 유발한다.
- 21 8) SQL 문 자체는 비 절차적 언어이므로, 여러 개의 질의문 사이에 연결이나 절차가 있어야 할
때에는 사용할 수 없다.
- 22 9) 실제 프로그래밍에서는 다른 언어를 사용해서 각각의 SQL 문들을 서로 연관되도록 하고 절
차적 또는 순차적인 단계를 가지고 SQL문이 실행되도록 해야 한다.
- 23 10) 다른 언어를 이용해서 처리해도 되고, MySQL 자체적으로는 SQL Programming을
사용한다.
- 24 11) 다른 RDBMS에서는 사용할 수 없다.

25
26
27 REM SQL Programming

28 1. 개요

- 29 1) SQL 문의 제한을 극복하기 위해 MySQL에서 추가적으로 만든, SQL 언어에 절차적인
프로그래밍 언어를 가미해 생성
- 30 2) 일반 프로그래밍의 언어적인 요소를 거의 다 가지고 있다.

31
32 2. 특징

- 33 1) 프로그램 개발시 모듈화
- 논리적 문장들을 그룹화
- 복잡한 프로그램 모듈을 그룹화가능
- 36 2) 변수 선언
- 37 3) 절차적 구조로 된 프로그래밍
- 조건문, 반복문
- 38 4) ERROR 처리 가능

39
40
41 3. 구조

- 42 delimiter --> block의 시작 (필수)
- 43 DECLARE --> 변수의 선언 (선택)
- 44 SET --> 변수의 선언 및 값 할당(선택)
- 45 BEGIN --> 실행부 시작 (필수)
- 46 END; --> block의 끝 (필수)

DECLARE SET 가 .

SQL Programing SQL

47

48 4. MySQL User-defined Variables

49 1)MySQL은 또한 한 명령문에서 다른 명령문으로 값을 전달할 수 있는 사용자 정의 변수의 개념을 지원한다.

50 2)MySQL의 사용자 정의 변수는 @var_name으로 작성

51 3)여기서 var_name은 변수의 이름이며 영숫자 문자, ., _ 및 \$로 구성될 수 있다.

52 4)사용자 정의 변수는 세션에 따라 다르다.

53 -한 클라이언트에서 정의한 변수는 다른 클라이언트와 공유되지 않으며 세션이 종료되면 이러한 변수는 자동으로 만료된다.

54 5)변수 이름은 대소문자를 구별하지 않는다.

55 -@mark or @Mark 같다.

56 6)최대 이름의 길이는 64 글자이다.

57 7)변수이름에는 특수문자 즉 !, #, ^, -, 등...이 포함될 수 있다.

58 -단, 인용부호로 묶어야 한다.

59 -@'var@1' or @"var^2" or @`var3` (백틱)

60 8)이러한 변수는 선언할 수 없으며, 선언 시에만 초기화된다.

61 -즉, 값을 할당해야 한다.

62 9)선언되지 않은 변수는 SQL문장이 수행될 때 NULL로서 설정된다.

63 10)변수 선언시 데이터타입(integer, floating-point, decimal, binary, nonbinary string or NULL value)을 지정

64 11)Syntax

65 SET @var_name = expression

66

67 12)연산자 사용

68 -SET @변수명 을 사용시 = 대입연산자를 사용한다

69 -SELECT @변수명 을 사용시 := 과 같은 대입연산자를 사용한다.

70

71 mysql>SET @var1 = 2+6;

72 mysql>SET @var2 := @var1-2;

73

74 mysql>SELECT @var1, @var2;

75

76 @var1 @var2

77 -----

78 8 6

79

80

81 mysql>SELECT @var3;

82

83 @var3

84 -----

85 NULL

86

87

88 mysql>SELECT @var3 := 4;

89

90 @var3 := 4

91 -----

92 4

93

94 mysql>SELECT @var4 = 5;

@var4=5

NULL

@var4 SET NULL

```
mysql> SET @v1 = X'41';
mysql> SET @v2 = X'41'+0;
mysql> SET @v3 = CAST(X'41' AS UNSIGNED);
mysql> SELECT @v1, @v2, @v3;
```

```
+-----+-----+-----+
| @v1 | @v2 | @v3 |
+-----+-----+-----+
| A   | 65  | 65  |
+-----+-----+-----+
```

```
mysql> SET @v1 = b'1000001';
mysql> SET @v2 = b'1000001'+0;
mysql> SET @v3 = CAST(b'1000001' AS UNSIGNED);
mysql> SELECT @v1, @v2, @v3;
```

```
+-----+-----+-----+
| @v1 | @v2 | @v3 |
+-----+-----+-----+
| A   | 65  | 65  |
+-----+-----+-----+
```

```
SET @total_salary = (SELECT SUM(sal) FROM emp);
SELECT @total_salary;
```

13) 변수의 종류

-지역변수

--지역(로컬)변수는 프로시저(Procedure) 또는 트리거(Triiger) 내에서 로컬 변수 및 입력 매개 변수로 사용할 수 있다.

--즉, Declares 내 지역(로컬)변수를 사용함을 의미 한다.

---Syntax

DECLARE variable_name datatype(size) [**DEFAULT** default_value];

DECLARE RTN_VAL VARCHAR(8);

DECLARE total_price Oct(8,2) **DEFAULT** 0.0;

DECLARE a,b,c INT **DEFAULT** 0;

DELIMITER //

:() .

Create Procedure Test()

BEGIN BEGIN END Test

DECLARE A INT **DEFAULT** 100;

DECLARE B INT;

DECLARE C INT;

DECLARE D INT;

```

145      SET B = 90;
146      SET C = 45;
147      SET D = A + B - C;
148      SELECT A, B, C, D;
149  END
150  //
151  DELIMITER ;
152
153  CALL Test();
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179

```

-시스템 변수

--MySQL은 기본적으로 선언된 변수들이 존재한다. 이를 시스템 변수라 한다.
 --시스템 변수는 GLOBAL 또는 세션단위로 사용가능하다.
 --즉, 서버의 전체 작업과 클라이언트 연결 후 작업등 모든 부분에 영향을 준다.

--시스템 변수 선언

-- Syntax to Set value to a Global variable:

```

SET GLOBAL sort_buffer_size=1000000;
SET @@global.sort_buffer_size=1000000;

```

-- Syntax to Set value to a Session variable:

```

SET sort_buffer_size=1000000;
SET SESSION sort_buffer_size=1000000;
SET @@sort_buffer_size=1000000;
SET @@local.sort_buffer_size=10000;

```

--시스템 변수 확인

--- 모든 변수 확인

```
SHOW VARIABLES;
```

--- 특정 변수 확인

```
SELECT @@sort_buffer_size;
```

5. 조건문

1)IF 문

-Syntax

```
IF 조건 THEN
```

```
처리문;
```

```
END IF;
```

```
delimiter //
```

```
CREATE PROCEDURE if_test()
```

```
BEGIN
```

```
DECLARE var INT;
```

```
SET var = 52;
```

```
IF var % 2 = 0 THEN
```

```
SELECT 'Even Number';
```

```
END IF;
```

```
END
```

```

IF      THEN      // IF
...
ELSE IF      THEN //
...
ELSE              //
...
END IF;          // IF

```

```

196 //
197 delimiter ;
198
199 CALL if_test();
200
201
202 IF 조건 THEN
203     처리문1;
204 ELSE
205     처리문2;
206 END IF;
207
208 delimiter //
209 CREATE PROCEDURE if_test()
210 BEGIN
211     DECLARE var INT;
212     SET var = 51;
213     IF var % 2 = 0 THEN
214         SELECT 'Even Number';
215     ELSE
216         SELECT 'Odd Number';
217     END IF;
218 END
219 //
220 delimiter ;
221
222 CALL if_test();
223
224
225 IF 조건1 THEN
226     처리문1;
227 ELSEIF 조건2 THEN
228     처리문2;
229 ELSEIF 조건3 THEN
230     처리문3;
231 ...
232 ELSE
233     처리문N;
234 END IF;
235
236 delimiter //
237 CREATE PROCEDURE if_test()
238 BEGIN
239     DECLARE season VARCHAR(20);
240     SET season = '여름';
241     IF season = '봄' THEN
242         SELECT '진달래, 개나리';
243     ELSEIF season = '여름' THEN
244         SELECT '장미, 아카시아';
245     ELSEIF season = '가을' THEN
246         SELECT '코스모스, 백합';

```

```

247         ELSE
248             SELECT '동백, 매화';
249         END IF;
250     END
251     //
252     delimiter ;
253
254     CALL if_test();
255
256
257 --성적관리프로그램
258     delimiter //
259     CREATE PROCEDURE sungjukmgmt()
260     BEGIN
261         DECLARE irum VARCHAR(20);
262         DECLARE hakbun CHAR(6);
263         DECLARE kor, eng, mat, tot INT DEFAULT 0;
264         DECLARE average DECIMAL(5, 2) DEFAULT 0.00;
265         DECLARE hakjum CHAR(1) DEFAULT 'F';
266
267         SET irum = '백두산';
268         SET hakbun = '21-001';
269         SET kor = 78, eng = 89, mat = 99;
270         SET tot = kor + eng + mat;
271         SET average = tot / 3;
272         IF average <= 100 AND average >= 90 THEN
273             SET hakjum = 'A';
274         ELSEIF average < 90 AND average >= 80 THEN
275             SET hakjum = 'B';
276         ELSEIF average < 80 AND average >= 70 THEN
277             SET hakjum = 'C';
278         ELSEIF average < 70 AND average >= 60 THEN
279             SET hakjum = 'D';
280         ELSE
281             SELECT hakjum = 'F';
282         END IF;
283
284         SELECT CONCAT('이름 ==> ', irum, CHAR(10), '학번 ==> ', hakbun, CHAR(10),
285             '국어 ==> ', kor, CHAR(10), '영어 ==> ', eng, CHAR(10),
286             '수학 ==> ', mat, CHAR(10), '총점==> ', tot, CHAR(10),
287             '평균 ==> ', average, CHAR(10), '평점 ==> ', hakjum);
288     END
289     //
290     delimiter ;
291
292     CALL sungjukmgmt()
293
294
295 2)CASE문
296
297     CASE case_value

```

```

298 WHEN when_value THEN statement_list
299 [WHEN when_value THEN statement_list] ...
300 [ELSE statement_list]
301 END CASE
302
303 OR
304
305 CASE
306 WHEN search_condition THEN statement_list
307 [WHEN search_condition THEN statement_list] ...
308 [ELSE statement_list]
309 END CASE
310
311
312 --성적관리프로그램
313 delimiter //
314 CREATE PROCEDURE sungjukmgmt()
315 BEGIN
316 DECLARE irum VARCHAR(20);
317 DECLARE hakbun CHAR(6);
318 DECLARE kor, eng, mat, tot INT DEFAULT 0;
319 DECLARE average DECIMAL(5, 2) DEFAULT 0.00;
320 DECLARE hakjum CHAR(1) DEFAULT 'F';
321
322 SET irum = '백두산';
323 SET hakbun = '21-001';
324 SET kor = 78, eng = 89, mat = 99;
325 SET tot = kor + eng + mat;
326 SET average = tot / 3;
327
328 CASE
329 WHEN average >= 90 THEN
330 SET hakjum = 'A';
331 WHEN average >= 80 THEN
332 SET hakjum = 'B';
333 WHEN average >= 70 THEN
334 SET hakjum = 'C';
335 WHEN average >= 60 THEN
336 SET hakjum = 'D';
337 ELSE
338 SELECT hakjum = 'F';
339 END CASE;
340
341 SELECT CONCAT('이름 ==> ', irum, CHAR(10), '학번 ==> ', hakbun, CHAR(10),
342 '국어 ==> ', kor, CHAR(10), '영어 ==> ', eng, CHAR(10),
343 '수학 ==> ', mat, CHAR(10), '총점==> ', tot, CHAR(10),
344 '평균 ==> ', average, CHAR(10), '평점 ==> ', hakjum);
345 END
346 //
347 delimiter ;
348

```

349 **CALL** sungjukmgmt()

350

351

352 10. 반복문

353 1) Syntax

354 **WHILE** search_condition **DO**

355 statement_list

356 **END WHILE**

357

358 --5,4,3,2,1

359 delimiter //

360 **CREATE PROCEDURE** dowhile()

361 **BEGIN**

362 **DECLARE** i **INT** **DEFAULT** 5;

363 **DECLARE** str **VARCHAR**(50);

364 **SET** str = ' ';

365

366 **WHILE** i > 0 **DO**

367 **SET** str = **CONCAT**(str, i, ' ');

368 **SET** i = i - 1;

369 **END WHILE**;

370

371 **SELECT** **SUBSTRING**(**RTRIM**(str), 1, **LENGTH**(str) - 2);

372 **END**

373 //

374 delimiter ;

375

376 **CALL** dowhile();

377

378 --구구단

379 **CREATE TABLE** tbl_gugudan

380 (

381 **result** **VARCHAR**(100)

382);

383

384 delimiter //

385 **CREATE PROCEDURE** gugudan()

386 **BEGIN**

387 **DECLARE** i **INT**;

388 **DECLARE** j **INT**;

389 **DECLARE** str **VARCHAR**(100);

390

391 **SET** i = 1;

392 **WHILE** i < 10 **DO**

393 **SET** str = '';

394 **SET** j = 2;

395 **WHILE** j < 10 **DO**

396 **SET** str = **CONCAT**(str, j, ' x ', i, ' = ', j * i, ' ');

397 **SET** j = j + 1;

398 **END WHILE**;

399 **SET** i = i + 1;

MySQL	WHILE	.
WHILE (Boolean Expression) DO		
statement		
END WHILE		


```
400         INSERT INTO tbl_gugudan VALUES(str);
401     END WHILE;
402 END
403 //
404 delimiter ;
405
406 CALL gugudan();
407
408 SELECT * FROM tbl_gugudan;
```