jinyPHP 2권 - 21. 뷰 Views

**21. 뷰 View**

====================

이번 장에서는 가상의 테이블 VIEW에 대해서 살펴 보겠습니다. View 는 물리적으로 실제 파일 시스템과 매칭 되어 있는 형태가 아니라 특정 쿼리 명령에 매칭 되어진 가상의 테이블 입니다.

MYSQL의 view 기능은 5.0 버전 이상에서만 동작을 합니다.

SELECT 쿼리, WHERE, 서브쿼리 복합문, UNION, JOIN등 다양한 결합과 조건으로 데이터를 출력을 할 수 있습니다. VIEW는 이러한 출력의 결과물을 가상의 테이블화 하여 재사용을 할 수 있는 기능입니다. VIEW는 쿼리를 기반으로 만들어 지는 테이블로, 가상테이블에 대한 실제적인 파일은 가지고 있지 않습니다.

쿼리 결과를 기반으로 생성된 가상의 테이블은 실제 파일을 접근하는 것과 같은 동일한 작업을 할 수 있습니다. view 테이블 안에서도 SELECT, UPDATE 등의 쿼리 작업이 가능합니다. view 에서 데이터를 변경을 하면, 실제적인 테이블에서도 데이터가 변경이 됩니다.

다양한 where조건과 join 값의 결과물을 가상테이블 view를 통하여 편리하고 빠르게 데이터를 액세스 할 수 있습니다.

**21.1 view 생성**

====================

쿼리 기반의 가상테이블은 create view 명령을 통하여 생성을 할 수 있습니다. 가상테이블은 SELECT 결과조회를 가상테이블의 이름으로 별칭을 선언합니다. 다음은 View 테이블 생성의 쿼리 문법입니다.

**쿼리문법)**

|  |
| --- |
| CREATE VIEW view\_name **AS**  SELECT column1, column2, … FROM table\_name WHERE condition; |

가상테이블의 기반인 실제적인 테이블 데이터가 변경이 되면 VIEW 의 데이터도 변경이 됩니다. 또한 VIEW 에서도 데이터를 변경을 하면 실제적인 테이블의 데이터도 변경이 됩니다.

조건이 설정되어 있는 view 도 실제적인 테이블의 데이터가 변경이 되었을 때, view의 조건도 자동으로 반영이 되어 출력됩니다.

**21.1.1 쿼리 실습**

====================

products 테이블에서 판매가 활성화된 상품만 조회하여 새로운 가상테이블을 생성을 합니다.

**CREATE VIEW [Current Product List] AS**

**SELECT ProductID, ProductName FROM Products WHERE enable = ‘on’;**

이렇게 생성된 가상테이블도 기존 방식과 동일한 형태로 데이터를 조회를 할 수 있습니다.

**SELECT \* FROM [Current Product List];**

다음은 이중 select를 이용하여 결과값을 조회한 후, 이 결과를 가상의 테이블로 생성을 합니다.

CREATE VIEW [Products Average Price] AS

SELECT ProductName, UnitPrice FROM Products

WHERE UnitPrice > **(SELECT AVG(UnitPrice) FROM Products)**;

이렇게 생성된 가상테이블을 select 문을 통하여 조회가 가능합니다.

**SELECT \* FROM [Products Average Price];**

다음은 또다른 예제 입니다.

**CREATE VIEW [Category Sales For 2017] AS**

**SELECT DISTINCT CategoryName, Sum(ProductSales) AS CategorySales FROM [Product Sales for 2017] GROUP BY CategoryName;**

이렇게 생성된 가상테이블을 select 문을 통하여 조회가 가능합니다.

**SELECT \* FROM [Category Sales For 2017];**

또한 where 절을 추가하여 검색을 할 수도 있습니다.

**SELECT \* FROM [Category Sales For 2017] WHERE CategoryName = 'cpu';**

**21.1.2 PHP 실습**

====================

PHP 코드를 통하여 VIEW를 실습해 봅니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| // 입력된 select 쿼리를 기반으로 view를 셍성합니다.  public function createView($view, $selectQuery)  {  if ($view) {  if ($selectQuery) {  $queryString = "CREATE VIEW $view AS $selectQuery";  // 쿼리를 전송합니다.  if (mysqli\_query($this->dbcon, $queryString)=== TRUE) {  $this->msgEcho("쿼리성공] ".$queryString);  $this->msgEcho(" 뷰 생성!");  // 객체 반환, 매서드체인  return $this;  } else {  $this->msgEcho("Error] ".$queryString);  }    } else {  $this->msgEcho("Error] 조건 select 쿼리를 입력해 주세요..");  }  } else {  $this->msgEcho("Error] 뷰 이름을 적어주세요.");  }  } |

**예제파일) view-01.php**

|  |
| --- |
| <?php  include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();  // 뷰를 생성합니다.  $db->createView("viewtest", "select \* from members");  ?> |

출력화면)

mysql connected!

쿼리성공] CREATE VIEW viewtest AS select \* from members

뷰 생성!

**21.2 view 확인**

====================

VIEW 로 생성된 가상테이블도 기존 사용하던 테이블 목록 출력 쿼리에서도 확인할 수 있습니다. SHOW TABLES는 가상테이블과 물리적 테이블을 구분하지는 않습니다.

기존 테이블 목록을 확인하는 것과 같은 명령을 실행하시면 됩니다.

**쿼리문법)**

|  |
| --- |
| SHOW TABLES; |

show tables 명령은 테이블의 이름만 출력되기 때문에 기존 물리적 테이블인지 가상 테이블인지 확인을 할 수 없습니다.

**쿼리문법)**

|  |
| --- |
| show create table 테이블명; |

이런 경우에는 테이블의 상세 정보를 확인해 보면 알 수 있습니다.

|  |
| --- |
| mysql> **show create table mem\_view;**  +----------+--------------------------------------+----------------------+----------------------+  | View | Create View | character\_set\_client | collation\_connection |  +----------+--------------------------------------+----------------------+----------------------+  | mem\_view | CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW `mem\_view` AS select `members`.`Id` AS `Id`,`members`.`LastName` AS `LastName`,`members`.`FirstName` AS `FirstName`,`members`.`Address` AS `Address`,`members`.`City` AS `City`,`members`.`Country` AS `Country`,`members`.`manager` AS `manager`,`members`.`email` AS `email` from `members` | utf8 | utf8\_general\_ci |  +----------+--------------------------------------+----------------------+----------------------+  1 row in set (0.00 sec) |

가상테이블도 기존 물리적 테이블과 같이 상세 구조 정보를 확인할 수 있습니다. 심지어 명령어도 동일하게 같이 사용할 수 있습니다.

**쿼리문법)**

|  |
| --- |
| DESC TABLES; |

다음은 가상테이블의 상세 정보를 DESC로 확인을 합니다.

|  |
| --- |
| mysql> **desc mem\_view;**  +-----------+--------------+------+-----+---------+-------+  | Field | Type | Null | Key | Default | Extra |  +-----------+--------------+------+-----+---------+-------+  | Id | int(11) | NO | | 0 | |  | LastName | varchar(255) | YES | | NULL | |  | FirstName | varchar(255) | YES | | NULL | |  | Address | varchar(255) | YES | | NULL | |  | City | varchar(255) | YES | | NULL | |  | Country | varchar(255) | YES | | NULL | |  | manager | varchar(100) | YES | | NULL | |  | email | varchar(255) | YES | | NULL | |  +-----------+--------------+------+-----+---------+-------+  8 rows in set (0.01 sec) |

**21.3 view 데이터 삽입**

====================

VIEW로 생성된 가상테이블도 새로운 데이터를 삽입을 할 수 있습니다. 하지만, 모든 가상테이블에 데이터를 삽입을 할 수는 없습니다. 가상테이블에 데이터를 삽입을 하는 것은 상황마다 다르게 동작합니다.

모든 가상테이블에 데이터를 삽입할 수 없는 이유는 VIEW 를 생성하는 기본 쿼리의 조건이 매우 다양하기 때문입니다. 가상테이블은 여러 테이블의 조건 등을 결합하여 만들 수가 있는데, JOIN, UNION, 서브 쿼리를 통하여 생성한 VIEW 에서는 INSERT, UPDATE 쿼리를 실행할 수 없습니다.

따라서 가상의 view 테이블에 INSERT 명령을 통하여 데이터를 삽입을 하는 데는 약간의 제한이 있습니다. 하지만 그 이외의 view 가상테이블에는 INSERT, UPDATE 쿼리를 실행할 수 있습니다.

VIEW 에 INSERT 쿼리를 실행하면, 어떤 테이블에는 데이터가 들어갈 수도 있고, 다른 테이블에는 데이터가 없을 수도 있습니다.

**21.4 view 조건일치**

====================

VIEW 가상 테이블의 일부는 간단한 where 절을 포함한 조건일수 있습니다. 이러한 가상 테이블은 데이터를 삽입을 할 수 있는데, where 절과는 상관 없이 데이터를 입력할 수 있습니다.

만일, WHERE 조건문이 맞지 않는 데이터를 삽입을 하고자 할 경우에는 다음과 같이 WITH CHECK OPTION 명령을 통하여 데이터 입력을 필터링 할 수 있습니다.

**쿼리문법)**

|  |
| --- |
| CREATE VIEW view\_name **AS**  SELECT column1, column2, … FROM table\_name WHERE condition  **WITH CHECK OPTION**; |

view 를 생성시에 WITH CHECK OPTION 키워드를 추가하면, view 에 데이터를 입력할 때 조건이 일하지 않으면 오류를 발생합니다.

**21.5 View 업데이트**

====================

가상테이블 VIEW 의 일부분만 업데이트 하는 방법은 없습니다. 하지만, 기존의 가상테이블 조건 전체로 덮어쓰기 형태로 업데이트를 할 수 있습니다.

**SQL CREATE OR REPLACE VIEW 문법)**

|  |
| --- |
| CREATE **OR REPLACE** VIEW view\_name AS  SELECT column1, column2, … FROM table\_name WHERE condition; |

CREATE 를 통하여 view 를 생성할때 **OR REPLACE** 키워드를 같이 사용하면 기존의 생성된 view 를 덮어쓰기 하여 업데이트를 할 수 있습니다.

**21.5.1 쿼리 실습**

====================

다음과 같은 예로 VIEW 테이블을 업데이트 하는 쿼리를 작성해 볼 수 있습니다.

CREATE **OR REPLACE** VIEW [Current Product List] AS

SELECT ProductID, ProductName, Category FROM Products WHERE enable = ‘on’;

**21.5.2 PHP 실습**

====================

PHP 코드를 통하여 실습해 봅니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| public function updateView($view, $selectQuery)  {  if ($view) {  if ($selectQuery) {  $queryString = "CREATE OR REPLACE VIEW $view AS $selectQuery";  // 쿼리를 전송합니다.  if (mysqli\_query($this->dbcon, $queryString)=== TRUE) {  $this->msgEcho("쿼리성공] ".$queryString);  $this->msgEcho(" 뷰 생성!");  return true;  } else {  $this->msgEcho("Error] ".$queryString);  return false;  }    } else {  $this->msgEcho("Error] 조건 select 쿼리를 입력해 주세요..");  return false;  }  } else {  $this->msgEcho("Error] 뷰 이름을 적어주세요.");  return false;  }  } |

**예제파일) view-03.php**

|  |
| --- |
| <?php    include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();    // 뷰를 업데이트 합니다.  $db->updateView("viewtest", "select \* from members where Id=1");    ?> |

출력화면)

mysql connected!

쿼리성공] CREATE OR REPLACE VIEW viewtest AS select \* from members where Id=1

뷰 생성!

**21.6 View 컬럼 변경**

====================

ALTER VIEW 키워드를 통하여 view 의 컬럼구조를 변경할 수 있습니다. ALTER VIEW는 CREATE VIEW 를 통하여 새로 생성하는 것과 거의 유사합니다.

**쿼리문법)**

|  |
| --- |
| ALTER VIEW 뷰이름 AS SELECT 컬럼명 FROM 테이블명; |

**21.6.1 쿼리 실습**

====================

기존 가상 테이블의 데이터를 출력해 봅니다.

|  |
| --- |
| mysql> select \* from viewtest;  +----+----------+-----------+----------+-------+---------+-----------+-------------------+  | Id | LastName | FirstName | Address | City | Country | manager | email |  +----+----------+-----------+----------+-------+---------+-----------+-------------------+  | 1 | hojin | lee | shinchon | seoul | Korea | infohojin | hojin@jinyphp.com |  +----+----------+-----------+----------+-------+---------+-----------+-------------------+  1 row in set (0.00 sec) |

가상테이블의 컬럼을 새롭게 지정합니다. 뷰의 출력 컬럼필드를 수를 변경해 보겠습니다.

**alter view viewtest as select FirstName, Lastname from members where Id = 1;**

|  |
| --- |
| mysql> alter view viewtest as select FirstName, Lastname from members where Id = 1;  Query OK, 0 rows affected (0.01 sec) |

정상적으로 뷰가 변경이 되었습니다. 다시한번 가상 테이블의 목록을 출력해 봅니다.

|  |
| --- |
| mysql> select \* from viewtest;  +-----------+----------+  | FirstName | Lastname |  +-----------+----------+  | lee | hojin |  +-----------+----------+  1 row in set (0.00 sec) |

2개의 컬럼만 출력되는 것을 확인 할 수 있습니다.

**21.6.2 PHP 실습**

====================

PHP 코드를 통하여 실습해 봅니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| public function alterView($view, $selectQuery)  {  if ($view) {  if ($selectQuery) {  $queryString = "ALTER VIEW $view AS $selectQuery";  // 쿼리를 전송합니다.  if (mysqli\_query($this->dbcon, $queryString)=== TRUE) {  $this->msgEcho("쿼리성공] ".$queryString);  $this->msgEcho(" 뷰 생성!");  return true;  } else {  $this->msgEcho("Error] ".$queryString);  return false;  }    } else {  $this->msgEcho("Error] 조건 select 쿼리를 입력해 주세요..");  return false;  }  } else {  $this->msgEcho("Error] 뷰 이름을 적어주세요.");  return false;  }  } |

**예제파일) view-05.php**

|  |
| --- |
| <?php    include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();    // 뷰를 업데이트 합니다.  $db->alterView("viewtest", "select FirstName, LastName from members where Id=1");    ?> |

출력화면)

mysql connected!

쿼리성공] ALTER VIEW viewtest AS select FirstName, LastName from members where Id=1

뷰 생성!

**21.7 View 삭제**

====================

생성한 가상테이블은 drop명령을 통하여 삭제를 할 수 있습니다. 테이블을 삭제하는 명령과 비슷하지만 TABLES 키워드 대신에 VIEW 키워드를 사용합니다.

**쿼리문법)**

|  |
| --- |
| DROP VIEW 뷰이름; |

만일 DROP VIEW 를 통하여 생성한 가상테이브를 삭제 할려고 할때, 가상테이블이 없으면 오류를 발생합니다. 즉, 존재하지 않는 view 테이블을 삭제를 하려고 할때 오류를 발생합니다.

이런경우에는 IF EXISTS 를 같이 사용해 주시면 좋습니다.

**쿼리문법)**

|  |
| --- |
| DROP VIEW **IF EXISTS** 뷰이름; |

**20.7.1 쿼리 실습**

====================

생성된 가상테이블을 삭제해 봅니다.

**drop view viewtest;**

|  |
| --- |
| mysql> drop view viewtest;  Query OK, 0 rows affected (0.00 sec) |

삭제 쿼리가 정상적으로 실행이 되었습니다.

**21.7.2 PHP 실습**

====================

PHP 코드를 통하여 실습해 봅니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| public function dropView($view)  {  if ($view){  $queryString = "DROP VIEW IF EXISTS $view";  // 쿼리를 전송합니다.  if (mysqli\_query($this->dbcon, $queryString)=== TRUE){  $this->msgEcho("쿼리성공] ".$queryString);  $this->msgEcho(" 뷰 삭제!");  // 객체 반환, 매서드체인  return $this;  } else {  $this->msgEcho("Error] ".$queryString);  }  } else {  $this->msgEcho("Error] 뷰 이름을 적어주세요.");  }  } |

**예제파일) view-04.php**

|  |
| --- |
| <?php  include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();  // 뷰를 삭제합니다.  $db->dropView("viewtest");  ?> |

출력화면)

mysql connected!

쿼리성공] DROP VIEW IF EXISTS viewtest

뷰 삭제!