jinyPHP 2권 - 014. 데이터의 정렬

**14. 데이터의 정렬**

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

테이블에 입력된 데이터의 출력은 입력한 데이터 순서대로 출력이 됩니다. 데이터를 검색하여 출력을 할 때 무작위로 데이터가 출력이 되면 결과를 쉽게 분석하기 어렵습니다.

데이터를 검색하고 출력할 때는 처리한 결과를 쉽게 알기 위해서 정렬을 하여 출력을 합니다. 정렬기능은 결과물을 보기 좋게, 가독성 있게 만들어 주는 중요한 기능입니다.

**14.1 정렬**

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

쿼리에서 정렬 기능을 추가하기 위해서는 ORDER BY 키워드를 사용합니다. 정렬은 컬럼값을 기준으로 오름차순(ASC), 내림차순(DESC) 정렬을 할 수 있습니다.

**쿼리문법)**

|  |
| --- |
| ~ ORDER BY 컬럼명 정렬방식; |

SELECT 쿼리 문장 마지막에 **ORDER BY 컬럼명** 을 추가하여 정렬을 하실 수 있습니다.

* ASC: 오름차순
* DESC : 내림차순

**14.1.1 쿼리 실습**

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

회원목록 테이블에서 Id 값을 기준으로 내림 차순 정렬을 합니다.

**select \* from members order by Id desc;**

|  |
| --- |
| mysql> **select \* from members order by Id desc;**  +----+----------+-----------+-------------+-------+---------+  | Id | LastName | FirstName | Address | City | Country |  +----+----------+-----------+-------------+-------+---------+  | 4 | eric | jung | seocho-gu | seoul | korea |  | 3 | james | kim | youngdun-po | seoul | korea |  | 2 | jiny | NULL | NULL | NULL | korea |  | 1 | hojin | lee | shinchon | seoul | korea |  +----+----------+-----------+-------------+-------+---------+  4 rows in set (0.00 sec) |

위의 예제는 데이터 출력시 Id 값 내림차순으로 변경하여 출력을 합니다.

만일 정렬을 해야 되는 컬럼이 여러 개일 경우 콤마(,)를 통하여 정렬할 컬럼을 추가할 수 있습니다.

**SELECT \* FROM members ORDER BY Country, City;**

또는 여러 컬럼을 정렬을 할 때 컬럼별로 정렬 방식을 지정을 할 수도 있습니다.

**SELECT \* FROM members ORDER BY Country ASC, City DESC;**

**14.1.2 PHP 실습**

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

PHP 코드를 통하여 생성한 쿼리에 정렬을 추가하여 실행해 보도록 합니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| public function orderBy($query,$by)  {  return $query . " ORDER BY " . $by;  } |

**예제파일) orderby-01.php**

|  |
| --- |
| <?php  include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();  $query = "select FirstName, LastName, City, Country from members";  $queryString = $db->orderBy($query,"Country ASC, City DESC");  if ($rowss = $db->selectRowss($queryString)) {  echo "tables fields = ". count($rowss) . "<br>";    for ($i=0;$i<count($rowss);$i++) {  echo $i."=";  print\_r($rowss[$i]);  echo "<br>";  }  }  ?> |

출력화면)

mysql connected!

쿼리성공] select FirstName, LastName, City, Country from members ORDER BY Country ASC, City DESC

tables fields = 6

0=stdClass Object ( [FirstName] => lee [LastName] => hojin [City] => seoul [Country] => Korea )

1=stdClass Object ( [FirstName] => kim [LastName] => james [City] => seoul [Country] => Korea )

2=stdClass Object ( [FirstName] => hojin [LastName] => lee [City] => [Country] => Korea )

3=stdClass Object ( [FirstName] => jiny [LastName] => 1234 [City] => [Country] => Korea )

4=stdClass Object ( [FirstName] => jiny [LastName] => 1234 [City] => [Country] => Korea )

5=stdClass Object ( [FirstName] => jiny [LastName] => 123400 [City] => [Country] => Korea )

**14.2 그룹**

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

데이터를 검색할 때 정렬 기능 이외에 같이 사용하는 명령중에서 그룹이 있습니다. 그룹은 컬럼의 값을 기준으로 동일한 내용을 가지는 값들끼리 그룹으로 지정하여 출력을 하는 것입니다.

컬럼의 값을 기준으로 그룹을 설정하는데, SELECT 쿼리 마지막에 **GROUP BY** 키워드를 추가하여 사용하면 됩니다.

**쿼리문법)**

|  |
| --- |
| ~ GROUP BY 그룹컬럼명; |

**14.2.1 쿼리 실습**

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

회원 목록을 도시별로 그룹화 화여 출력을 해봅니다. 먼저 그룹화 전의 데이터를 확인해 봅니다.

|  |
| --- |
| mysql> **select \* from members;**  +----+----------+-----------+-------------+-------+---------+-----------+-------------------+  | Id | LastName | FirstName | Address | City | Country | manager | email |  +----+----------+-----------+-------------+-------+---------+-----------+-------------------+  | 1 | hojin | lee | shinchon | seoul | korea | infohojin | hojin@jinyphp.com |  | 2 | jiny | NULL | NULL | NULL | korea | infohojin | jiny@jinyphp.com |  | 3 | james | kim | youngdun-po | seoul | korea | info | james@jinyphp.com |  | 4 | eric | jung | seocho-gu | busan | korea | info | eric@jinyphp.com |  +----+----------+-----------+-------------+-------+---------+-----------+-------------------+  4 rows in set (0.00 sec) |

이번에는 도시별로 그룹화 처리한 쿼리 명령을 생성하여 실행해 봅니다.

**select \* from members group by City;**

|  |
| --- |
| mysql> **select \* from members group by City;**  +----+----------+-----------+-----------+-------+---------+-----------+-------------------+  | Id | LastName | FirstName | Address | City | Country | manager | email |  +----+----------+-----------+-----------+-------+---------+-----------+-------------------+  | 2 | jiny | NULL | NULL | NULL | korea | infohojin | jiny@jinyphp.com |  | 4 | eric | jung | seocho-gu | busan | korea | info | eric@jinyphp.com |  | 1 | hojin | lee | shinchon | seoul | korea | infohojin | hojin@jinyphp.com |  +----+----------+-----------+-----------+-------+---------+-----------+-------------------+  3 rows in set (0.00 sec) |

GROUP BY 컬럼 을 기준으로 각각 1개의 데이터만 출력됩니다. 그룹설정은 테이블의 데이터의 종류를 판별할 때 매우 유용합니다.

**14.2.2 각 그룹의 갯수**

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

GROUP BY 는 중복되지 않은 컬럼의 값 1개씩만 출력됩니다. 내부함수 count()를 이용하면 각각의 그룹의 갯수를 같이 확인을 할 수 있습니다.

**SELECT City, count(\*) from members group by City;**

|  |
| --- |
| mysql> **SELECT City, count(\*) from members group by City;**  +-------+----------+  | City | count(\*) |  +-------+----------+  | NULL | 1 |  | busan | 1 |  | seoul | 2 |  +-------+----------+  3 rows in set (0.00 sec) |

GROUP BY 명령어는 하나이상의 컬럼을 그룹화 처리가 가능합니다. 한 예로 어떤 컬럼의 수를 카운트하거나 최대값, 최소값, 합계, 평균등의 작업을 할 때 유용하게 사용을 할 수 있습니다.

**14.2.3 PHP 실습**

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

PHP 코드를 통하여 그룹 기능을 실습해 봅니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| public function groupBy($query,$by)  {  return $query . " GROUP BY " . $by;  } |

**예제파일) groupby-01.php**

|  |
| --- |
| <?php  include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();  $query = "select FirstName, LastName, City, Country from members";  $queryString = $db->groupBy($query,"City");  if($rowss = $db->selectRowss($queryString)){  echo "tables fields = ". count($rowss) . "<br>";    for($i=0;$i<count($rowss);$i++){  echo $i."=";  print\_r($rowss[$i]);  echo "<br>";  }  }  ?> |

출력화면)

mysql connected!

쿼리성공] select FirstName, LastName, City, Country from members GROUP BY City

tables fields = 2

0=stdClass Object ( [FirstName] => hojin [LastName] => lee [City] => [Country] => Korea )

1=stdClass Object ( [FirstName] => lee [LastName] => hojin [City] => seoul [Country] => Korea )

**14.3 HAVING**

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

HAVING은 그룹화 결과를 가지고 추가적인 조건을 설정할 수 있습니다.

**쿼리문법)**

|  |
| --- |
| SELECT 컬럼, 컬럼 FROM 테이블명  GROUP BY 그룹컬럼  HAVING 조건; |

**14.3.1 쿼리 실습**

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

아래 쿼리 예제는 주문 목록에서 이메일 회원별로 그룹화를 합니다. 그룹화된 목록에서 그룹으로 묽인 횟수가 1보다 큰 조건을 찾습니다.

**SELECT email, count(email) as order\_count FROM orders1 group by email HAVING order\_count >=1;**

|  |
| --- |
| mysql> **SELECT email, count(email) as order\_count FROM orders1 group by email HAVING order\_count >=1;**  +-------------------+-------------+  | email | order\_count |  +-------------------+-------------+  | eric@jinyphp.com | 1 |  | hojin@jinyphp.com | 2 |  +-------------------+-------------+  2 rows in set (0.00 sec) |

각각의 그룹을 카운트 하여 주문횟수가 1인 이상의 회원만 추출하는 HAVING 검색 조건을 넣습니다.

WHERE 조건 우선 사용 => GROUP BY 사용 => ORDER BY 정렬 사용 => HAVING 사용 조건 순으로 결합할 수 있습니다.

**14.3.2 PHP 실습**

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

PHP 코드를 통하여 Having 처리 실습을 해봅니다.

**PHP 예제)**

mysql.class.php 파일에 메서드 예제를 추가합니다.

|  |
| --- |
| public function groupByHaving($query,$by,$where)  {  return $query . " GROUP BY " . $by . " HAVING $where";  } |

**예제일) groupby-02.php**

|  |
| --- |
| <?php  include "dbinfo.php";  include "mysql.class.php";    // ++ Mysqli DB 연결.  $db = new JinyMysql();  $query = "SELECT email, count(email) as order\_count FROM orders";  $queryString = $db->groupByHaving($query,"email","order\_count >=1");  if($rowss = $db->selectRowss($queryString)){  echo "tables fields = ". count($rowss) . "<br>";    for($i=0;$i<count($rowss);$i++){  echo $i."=";  print\_r($rowss[$i]);  echo "<br>";  }  }  ?> |

출력화면)

mysql connected!

쿼리성공] SELECT email, count(email) as order\_count FROM orders GROUP BY email HAVING order\_count >=1

tables fields = 2

0=stdClass Object ( [email] => eric@jinyphp.com [order\_count] => 1 )

1=stdClass Object ( [email] => hojin@jinyphp.com [order\_count] => 2 )