# 简介

查询构建器建立在 [Database Access Objects](https://www.yiichina.com/doc/guide/2.0/db-dao) 基础之上

创建 程序化的、DBMS无关的SQL语句

可读性更强

安全性更强

注释

DBMS：数据库管理系统

# 使用步骤

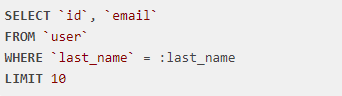
使用查询构建器通常包含以下两个步骤：

1. 创建一个 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 对象来代表一条 SELECT SQL 语句的不同子句（例如 SELECT, FROM）。
2. 执行 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 的一个查询方法（例如：all()）从数据库当中检索数据。

如下所示代码是查询构造器的一个典型用法：



上面的代码将会生成并执行如下的SQL语句，其中 :last\_name 参数绑定了 字符串 'Smith'。



****提示：****你平时更多的时候会使用 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 而不是 [yii\db\QueryBuilder](https://www.yiichina.com/doc/api/2.0/yii-db-querybuilder)。 当你调用其中一个查询方法时，后者将会被前者隐式的调用。[yii\db\QueryBuilder](https://www.yiichina.com/doc/api/2.0/yii-db-querybuilder)主要负责将 DBMS 不相关的 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 对象转换成 DBMS 相关的 SQL 语句（例如， 以不同的方式引用表或字段名称）。

# 创建查询

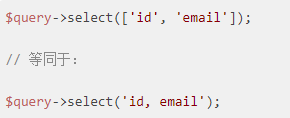
创建一个 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 对象

* 调用不同的查询构建方法来代表SQL语句的不同子句。
  + 例如，为了指定 SQL 语句当中的 FROM 子句，你应该调用 from() 方法。
* 链式调用。所有的查询构建器方法返回的是查询对象本身，也就是说，你可以把多个方法的调用串联起来。

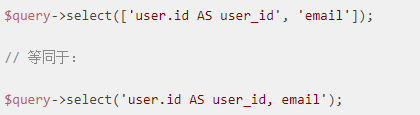
## [select()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "select()-detail)

[select()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "select()-detail) 方法用来指定 SQL 语句当中的 SELECT 子句。

使用一个数组或者字符串来定义需要查询的字段



可以加表前缀、字段别名

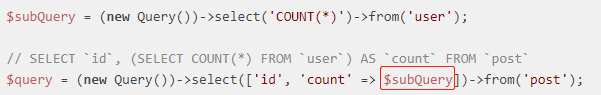


使用数据库的表达式



子查询

在定义每一个子查询的时候， 你应该使用 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 对象。



选取附加字段

可以根据判断条件自由添加附加字段



## [from()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "from()-detail)

[from()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "from()-detail) 方法指定了 SQL 语句当中的 FROM 子句。

例如：

// SELECT \* FROM `user`

$query->from('user');

子查询



## [where()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "where()-detail)

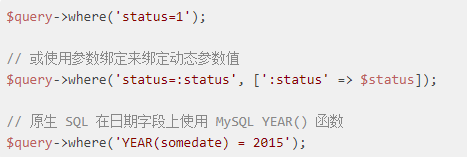
[where()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "where()-detail) 方法定义了 SQL 语句当中的 WHERE 子句。

你可以使用如下四种格式来定义 WHERE 条件：

* 字符串格式，例如：'status=1'
* 哈希格式，例如： ['status' => 1, 'type' => 2]
* 操作符格式，例如：['like', 'name', 'test']
* 对象格式，例如：new LikeCondition('name', 'LIKE', 'test')

### 字符串格式

在定义非常简单的查询条件的时候，字符串格式是最合适的。 它看起来和原生 SQL 语句差不多。例如：



千万不要像如下的例子一样直接在条件语句当中嵌入变量，特别是当这些变量来源于终端用户输入的时候， 因为这样我们的软件将很容易受到 SQL 注入的攻击。

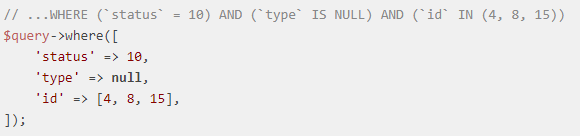


### 哈希格式

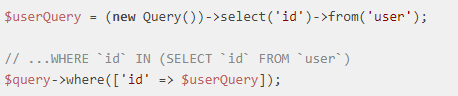
哈希格式最适合用来指定多个 AND 串联起来的简单的"等于断言"子条件。

它是以数组的形式来书写的，数组的键表示字段的名称，而数组的值则表示 这个字段需要匹配的值。

例如：



子查询



优点：

* 哈希格式非常的智能，能恰当地处理数值当中的空值和数组。
* 不需要手动添加参数绑定。

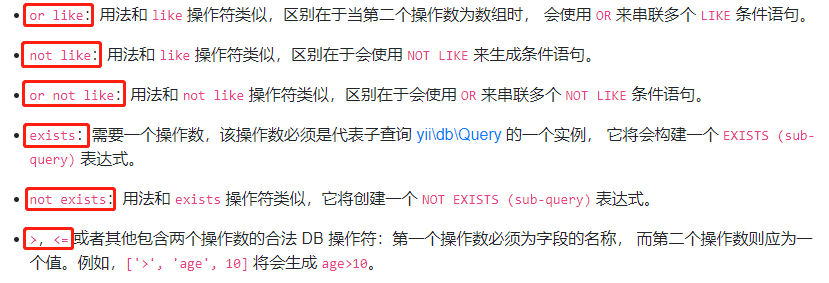
### 操作符格式

操作符格式允许你指定类程序风格的任意条件语句，如下所示：

[操作符, 操作数1, 操作数2, ...]

其中每个操作数可以是字符串格式、哈希格式或者嵌套的操作符格式， 而操作符可以是如下列表中的一个：





优点：

* 自动参数绑定
* 非常灵活

### 对象格式（Object Form）

对象格式自 2.0.14 开始提供。

在查询构建器上自定义查询条件类，不建议，yii提供的查询方式完全够用，复杂的写原生SQL即可。

在内部，上面描述的格式在构建 SQL 之前被隐式转换为对象格式，也就是说字符串、哈希、操作符格式是基于对象格式的，我们也可以用对象格式自定义自己的格式。

## [orderBy()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "orderBy()-detail)

[orderBy()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "orderBy()-detail) 方法是用来指定 SQL 语句当中的 ORDER BY 子句的。例如，

数组格式

// ... ORDER BY `id` ASC, `name` DESC

$query->orderBy([

'id' => SORT\_ASC,

'name' => SORT\_DESC,

]);

字符串格式

$query->orderBy('id ASC, name DESC');

调用 [addOrderBy()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "addOrderBy()-detail) 来为 ORDER BY 片断添加额外的子句。 例如，

$query->orderBy('id ASC')

->addOrderBy('name DESC');

## [groupBy()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "groupBy()-detail)

[groupBy()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "groupBy()-detail) 方法是用来指定 SQL 语句当中的 GROUP BY 片断的。例如，

数组格式

// ... GROUP BY `id`, `status`

$query->groupBy(['id', 'status']);

字符串格式

$query->groupBy('id, status');

调用 [addOrderBy()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "addOrderBy()-detail) 来为 GROUP BY 子句添加额外的字段。例如，

$query->groupBy(['id', 'status'])

->addGroupBy('age');

## [having()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "having()-detail)

[having()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "having()-detail) 方法是用来指定 SQL 语句当中的 HAVING 子句。它带有一个条件， 和 [where()](https://www.yiichina.com/doc/guide/2.0/db-query-builder" \l "where) 中指定条件的方法一样。例如，

// ... HAVING `status` = 1

$query->having(['status' => 1]);

请查阅 [where()](https://www.yiichina.com/doc/guide/2.0/db-query-builder" \l "where) 的文档来获取更多有关于如何指定一个条件的细节。

你可以调用 [andHaving()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "andHaving()-detail) 或者 [orHaving()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "orHaving()-detail) 方法来为 HAVING 子句追加额外的条件，例如，

// ... HAVING (`status` = 1) AND (`age` > 30)

$query->having(['status' => 1])

->andHaving(['>', 'age', 30]);

## [limit()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "limit()-detail) 和 [offset()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "offset()-detail)

[limit()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "limit()-detail) 和 [offset()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "offset()-detail) 是用来指定 SQL 语句当中 的 LIMIT 和 OFFSET 子句的。例如，

// ... LIMIT 10 OFFSET 20

$query->limit(10)->offset(20);

如果你指定了一个无效的 limit 或者 offset（例如，一个负数），那么它将会被忽略掉。

## [join()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "join()-detail)

[join()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "join()-detail) 是用来指定 SQL 语句当中的 JOIN 子句的。例如，

// ... LEFT JOIN `post` ON `post`.`user\_id` = `user`.`id`

$query->join('LEFT JOIN', 'post', 'post.user\_id = user.id');



### 连接子查询

将需要被连接的子查询指定 为一个 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 对象，例如，

$subQuery = (**new** \yii\db\Query())->from('post');

$query->leftJoin(['u' => $subQuery], 'u.id = author\_id');

在这个例子当中，你应该将子查询放到一个数组当中，而数组当中的键，则为这个子查询的别名。

# 查询方法

[yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 提供了一整套的用于不同查询目的的方法。

* [all()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "all()-detail)：将返回一个由行组成的数组，每一行是一个由名称和值构成的关联数组（译者注：省略键的数组称为索引数组）。
* [one()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "one()-detail)：返回结果集的第一行。
* [column()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "column()-detail)：返回结果集的第一列。
* [scalar()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "scalar()-detail)：返回结果集的第一行第一列的标量值。
* [exists()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "exists()-detail)：返回一个表示该查询是否包含结果集的值。
* [count()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "count()-detail)：返回 COUNT 查询的结果。
* 其它集合查询方法：包括 [sum($q)](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "sum()-detail), [average($q)](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "average()-detail), [max($q)](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "max()-detail), [min($q)](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "min()-detail) 等。$q 是一个必选参数， 既可以是一个字段名称，又可以是一个 DB 表达式。

****注意：****[one()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "one()-detail) 方法只返回查询结果当中的第一条数据， 条件语句中不会加上 LIMIT 1 条件。如果需要加limit 1，应该显示调用 limit(1) 方法。

所有的这些查询方法都有一个可选的参数 $db, 该参数指代的是 [DB connection](https://www.yiichina.com/doc/api/2.0/yii-db-connection)， 执行一个 DB 查询时会用到。如果你省略了这个参数，那么 db [application component](https://www.yiichina.com/doc/guide/2.0/structure-application-components) 将会被用作 默认的 DB 连接。

## 内在运作机制

当你调用 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 当中的一个查询方法的时候，实际上内在的运作机制如下：

* 在当前 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 的构造基础之上，调用 [yii\db\QueryBuilder](https://www.yiichina.com/doc/api/2.0/yii-db-querybuilder) 来生成一条 SQL 语句；
* 利用生成的 SQL 语句创建一个 [yii\db\Command](https://www.yiichina.com/doc/api/2.0/yii-db-command) 对象；
* 调用 [yii\db\Command](https://www.yiichina.com/doc/api/2.0/yii-db-command) 的查询方法（例如，queryAll()）来执行这条 SQL 语句，并检索数据。

## 测试SQL语句

测试或者使用一个由 [yii\db\Query](https://www.yiichina.com/doc/api/2.0/yii-db-query) 对象创建的 SQL 语句。 你可以使用以下的代码来达到目的：



## 索引查询结果

当你在调用 [all()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "all()-detail) 方法时，它将返回一个以连续的整型数值为索引的数组。

而有时候你可能希望使用一个特定的字段或者表达式的值来作为索引结果集数组。那么你可以在调用 [all()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "all()-detail) 之前使用 [indexBy()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "indexBy()-detail) 方法来达到这个目的。 例如，

// 返回 [100 => ['id' => 100, 'username' => '...', ...], 101 => [...], 103 => [...], ...]

$query = (**new** \yii\db\Query())

->from('user')

->limit(10)

->indexBy('id')

->all();

如需使用表达式的值做为索引，那么只需要传递一个匿名函数给 [indexBy()](https://www.yiichina.com/doc/api/2.0/yii-db-querytrait" \l "indexBy()-detail) 方法即可：

$query = (**new** \yii\db\Query())

->from('user')

->indexBy(**function** ($row) {

**return** $row['id'] . $row['username'];

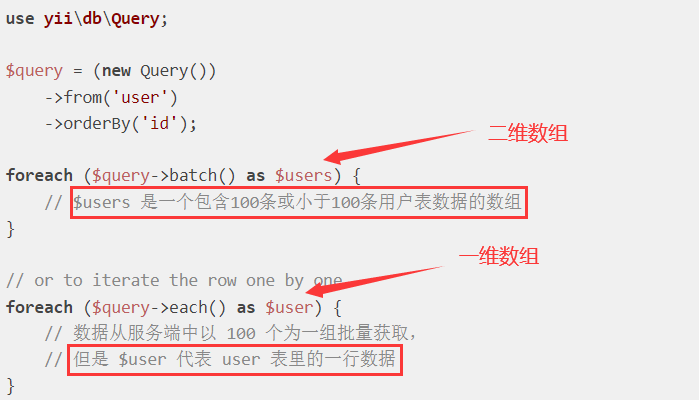
})->all();

## 批处理查询

当需要处理大数据的时候，像 [yii\db\Query::all()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "all()-detail) 这样的方法就不太合适了， 因为它们会把所有查询的数据都读取到客户端内存上。

为了解决这个问题， Yii 提供了批处理查询的支持。服务端先保存查询结果，然后客户端使用游标（cursor） 每次迭代出固定的一批结果集回来。

批处理查询的用法如下：



[yii\db\Query::batch()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "batch()-detail) 和 [yii\db\Query::each()](https://www.yiichina.com/doc/api/2.0/yii-db-query" \l "each()-detail) 方法将会返回一个实现了Iterator 接口 [yii\db\BatchQueryResult](https://www.yiichina.com/doc/api/2.0/yii-db-batchqueryresult) 的对象，可以用在 foreach 结构当中使用。在第一次迭代取数据的时候， 数据库会执行一次 SQL 查询，然后在剩下的迭代中，将直接从结果集中批量获取数据。默认情况下， 一批的大小为 100，也就意味着一批获取的数据是 100 行。你可以通过给 batch() 或者 each() 方法的第一个参数传值来改变每批行数的大小。

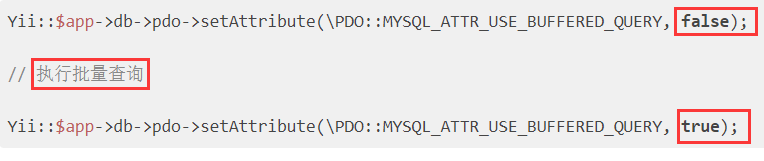
### MySQL中批量查询的局限性（Limitations of batch query in MySQL）

MySQL 是通过 PDO 驱动库实现批量查询的。默认情况下，MySQL 查询是 [带缓存的](http://php.net/manual/en/mysqlinfo.concepts.buffering.php)， 这违背了使用游标（cursor）获取数据的目的， 因为它不阻止驱动程序将整个结果集加载到客户端的内存中。

要禁用缓存并减少客户端内存的需求量，PDO 连接属性 PDO::MYSQL\_ATTR\_USE\_BUFFERED\_QUERY 必须设置为 false。

这样，直到整个数据集被处理完毕前，通过此连接是无法创建其他查询的。 这样的操作可能会阻碍 ActiveRecord 执行表结构查询。 如果这不构成问题（表结构已被缓存过了）， 我们可以通过切换原本的连接到非缓存模式，然后在批量查查询完成后再切换回来。

**表结构已被缓存过了**



如果**表结构没有被缓存，或在批量查询被处理过程中需要执行其他查询**， 你可以创建一个单独的非缓存链接到数据库：



****注意：****非缓存查询在 PHP 端使用更少的缓存，但会增加 MySQL 服务器端的负载。 建议您使用生产实践设计自己的代码以获取额外的海量数据，[例如，将数字键分段，使用非缓存的查询遍历](https://github.com/yiisoft/yii2/issues/8420" \l "issuecomment-296109257)。

### 批处理查询的替代方案

大数据尽量别用批量查询，对MySQL服务器的负担太大；

也不要一次性查大量数据，对MySQL服务器和PHP客户端的压力都大；

面对大数据量查询，可以根据id分段查询。

## 添加自定义查询条件和表达式（Adding custom Conditions and Expressions）

我们在 [查询条件-对象格式](https://www.yiichina.com/doc/guide/2.0/db-query-builder" \l "object-format) 章节中提到过，可以创建自定义的查询条件类。

不建议自定义查询条件类，很复杂，且局限性很大，其实就是把某些特定的查询条件自己封装一遍。

没这个必要，直接用yii提供的查询条件类即可，完全够用，再复杂就直接写原生SQL。

# 总结

其实查询构建器就是把一句完整的查询SQL根据子句拆分到多个构建方法中，通过底层构建达到可读性、安全性更强，与DBMS无关的效果。