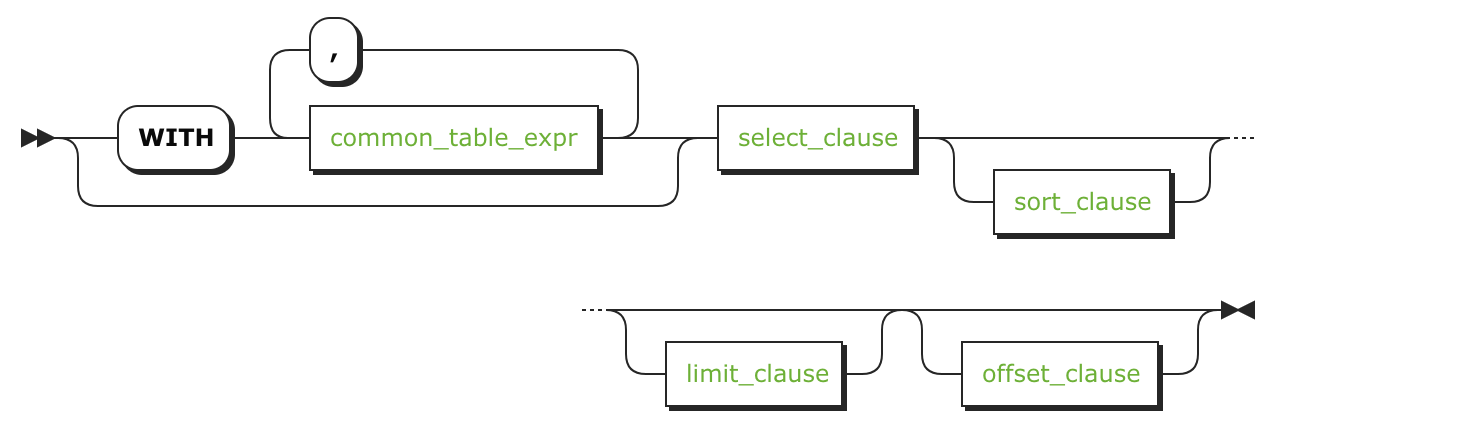
选择查询在CockroachDB中读取和处理数据. 它们比简单的SELECT子句更通用: 他们可以使用set操作对一个或多个selection子句进行分组，并可以请求特定的排序或行限制。

以下情况可以使用选择查询:

* 像其他SQL语句一样在查询的顶层。
* 在括号之间作为子查询。
* 作为将表格数据作为输入的其他语句的操作数，例如INSERT，UPSERT，CREATE TABLE AS或ALTER ... SPLIT AT。

概要



参数

| **Parameter** | **Description** |
| --- | --- |
| common\_table\_expr | 查看 [Common Table Expressions](https://www.cockroachlabs.com/docs/stable/common-table-expressions.html). |
| select\_clause | 有效的选择子句，单独使用或使用set操作。 |
| sort\_clause | 可选的ORDER BY子句，更多信息查看[Ordering Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/ordering-query-results/) |
| limit\_clause | 可选的LIMIT 子句，更多信息查看[Limiting Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/limit-query-results/) |
| offset\_clause | 可选的OFFSET子句，更多信息查看[Limiting Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/limit-query-results/) |

可选的LIMIT和OFFSET子句可以按任何顺序出现，但如果ORDER BY也存在，则必须在它后面使用。

因为WITH，ORDER BY，LIMIT和OFFSET子句都是可选的，所以任何简单的选择子句也是有效的选择查询。

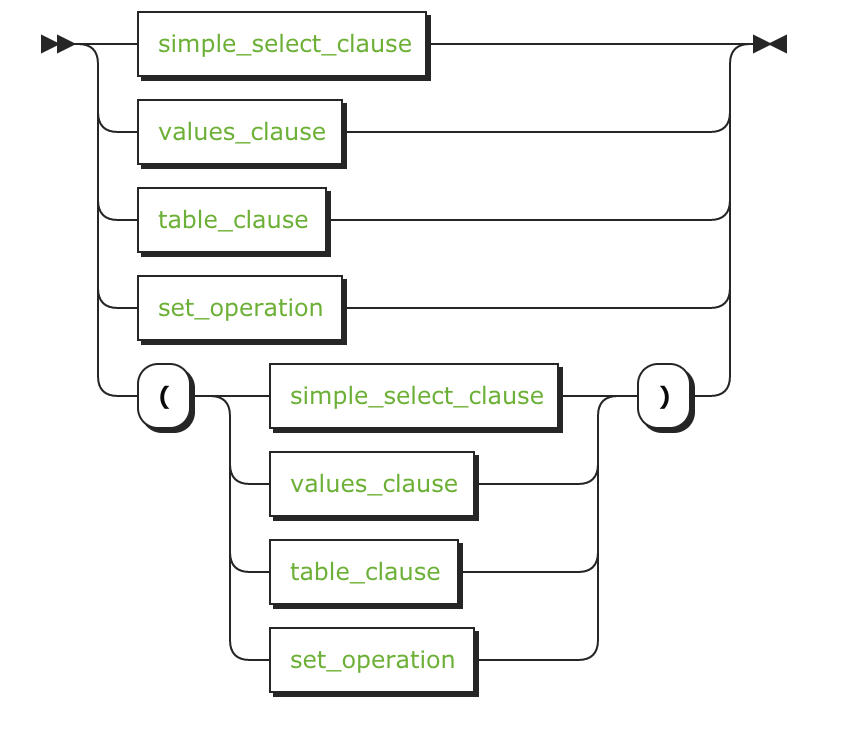
选择子句

Selection子句是选择查询的主要组成部分。 他们定义表格数据。 有四种特定的语法形式统称为选择子句：

| **Form** | **Usage** |
| --- | --- |
| [SELECT](https://www.cockroachlabs.com/docs/stable/#select-clause) | 从各种来源加载或计算表格数据。 这是最常见的选择子句。 |
| [VALUES](https://www.cockroachlabs.com/docs/stable/#values-clause) | 列出客户端的表格数据。 |
| [TABLE](https://www.cockroachlabs.com/docs/stable/#table-clause) | 从数据库加载表格数据。 |
| [Set Operations](https://www.cockroachlabs.com/docs/stable/#set-operations) | 合并来自两个或更多选择子句的表格数据。 |

要对selection子句执行连接或其他关系操作，请使用table表达式并使用TABLE或SELECT将其转换回选择子句。

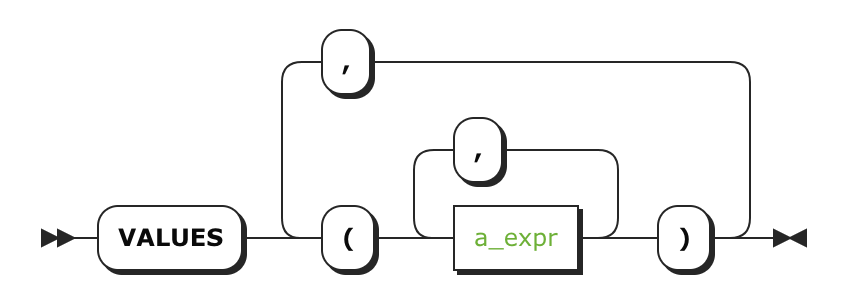
概要



概览

VALUES 子句

语法



VALUES子句定义由括号内列出的表达式定义的表格数据。每个括号组在结果表中定义一行。

结果表数据的列名称自动生成。当VALUES子句用作子查询时，[这些名称可以用AS来修改](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/table-expressions/#aliased-table-expressions)

举例

> VALUES (1, 2, 3), (4, 5, 6);

+---------+---------+---------+

| column1 | column2 | column3 |

+---------+---------+---------+

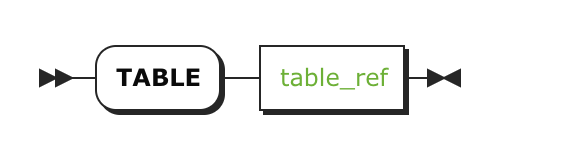
| 1 | 2 | 3 |

| 4 | 5 | 6 |

+---------+---------+---------+

TABLE 子句

语法



TABLE子句从指定的表中读取表格数据。 结果表数据的列以表的shema命名。

通常情况下，TABLE x等同于SELECT \* FROM x，使用起来更便捷。

提示：括号之间的任何table表达式都是TABLE的有效操作数，而不仅仅是简单的表或视图名称。

举例

> **CREATE** **TABLE** employee\_copy **AS** **TABLE** employee;

此语句将表employee中的内容复制到新表中。不过，TABLE子句不保留索引、外键和从其读取的表的模式中的约束和默认信息，所以在这个例子中，新表employee\_copy可能比employee具有更简单的schema。

另一个例子：

> TABLE employee;

> **INSERT** **INTO** employee\_copy **TABLE** employee;

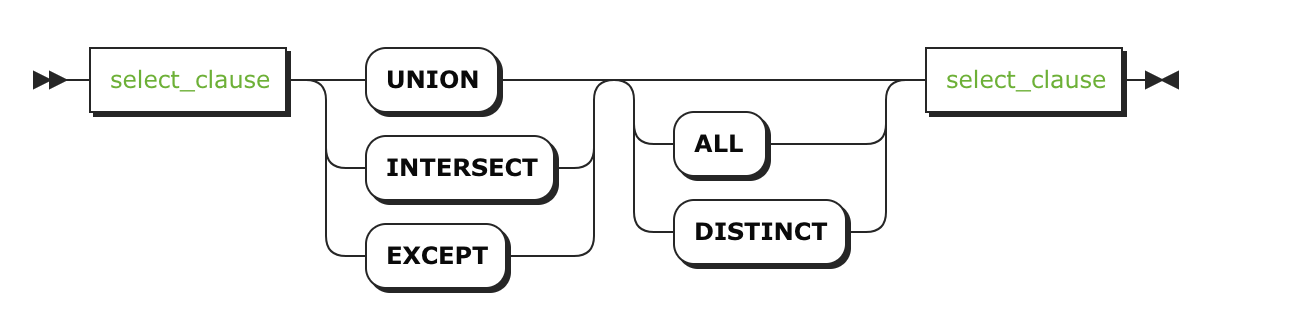
SELECT 子句

更多信息请查看 [Simple SELECT Clause](http://doc.cockroachchina.baidu.com/#develop/sql-statements/SELECT/)

集合操作

集合操作组合了两个选择子句的数据。它们作为其他集合操作的操作数或选择查询中的主要组件。

概要



集合运算符

SQL允许您比较多个选择子句的结果。您可以将每个集合运算符视为表示布尔运算符：

* UNION = OR
* INTERSECT = AND
* EXCEPT = NOT

默认情况下，每个比较只显示每个值的一个副本（类似于SELECT DISTINCT）。但是，每个函数还允许你向子句添加ALL以显示重复值。

Union: 结合两个查询

UNION将两个查询的结果合并为一个结果。

> **SELECT** name

**FROM** accounts

**WHERE** state\_opened **IN** ('AZ', 'NY')

**UNION**

**SELECT** name

**FROM** mortgages

**WHERE** state\_opened **IN** ('AZ', 'NY');

+-----------------+

| name |

+-----------------+

| Naseem Joossens |

| Ricarda Caron |

| Carola Dahl |

| Aygün Sanna |

+-----------------+

要显示重复的行，可以使用ALL。

> **SELECT** name

**FROM** accounts

**WHERE** state\_opened **IN** ('AZ', 'NY')

**UNION** **ALL**

**SELECT** name

**FROM** mortgages

**WHERE** state\_opened **IN** ('AZ', 'NY');

+-----------------+

| name |

+-----------------+

| Naseem Joossens |

| Ricarda Caron |

| Carola Dahl |

| Naseem Joossens |

| Aygün Sanna |

| Carola Dahl |

+-----------------+

Intersect: 检索两个查询的交集

INTERSECT只查找两个查询操作数中都存在的值。

> **SELECT** name

**FROM** accounts

**WHERE** state\_opened **IN** ('NJ', 'VA')

**INTERSECT**

**SELECT** name

**FROM** mortgages;

+-----------------+

| name |

+-----------------+

| Danijel Whinery |

| Agar Archer |

+-----------------+

Except: 从一个结果中排除另一个查询的结果

EXCEPT查找第一个查询操作数中存在的值，但不在第二个查询操作数中的值。

> **SELECT** name

**FROM** mortgages

**EXCEPT**

**SELECT** name

**FROM** accounts;

+------------------+

| name |

+------------------+

| Günay García |

| Karla Goddard |

| Cybele Seaver |

+------------------+

排序结果

下面举了一些例子，更多详细信息查看 [Ordering Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/ordering-query-results/).

按一列排序检索行

> **SELECT** \*

**FROM** accounts

**WHERE** balance **BETWEEN** 350 **AND** 500

**ORDER** **BY** balance **DESC**;

+----+--------------------+---------+----------+--------------+

| id | name | balance | type | state\_opened |

+----+--------------------+---------+----------+--------------+

| 12 | Raniya Žitnik | 500 | savings | CT |

| 59 | Annibale Karga | 500 | savings | ND |

| 27 | Adelbert Ventura | 500 | checking | IA |

| 86 | Theresa Slaski | 500 | checking | WY |

| 73 | Ruadh Draganov | 500 | checking | TN |

| 16 | Virginia Ruan | 400 | checking | HI |

| 43 | Tahirih Malinowski | 400 | checking | MS |

| 50 | Dusan Mallory | 350 | savings | NV |

+----+--------------------+---------+----------+--------------+

按多列排序检索行

列按照你在sortby\_list中列出的顺序排序，举个例子，ORDER BY a，b按行a对行进行排序，然后按行b值对具有相同值的行进行排序。

> **SELECT** \*

**FROM** accounts

**WHERE** balance **BETWEEN** 350 **AND** 500

**ORDER** **BY** balance **DESC**, name **ASC**;

+----+--------------------+---------+----------+--------------+

| id | name | balance | type | state\_opened |

+----+--------------------+---------+----------+--------------+

| 27 | Adelbert Ventura | 500 | checking | IA |

| 59 | Annibale Karga | 500 | savings | ND |

| 12 | Raniya Žitnik | 500 | savings | CT |

| 73 | Ruadh Draganov | 500 | checking | TN |

| 86 | Theresa Slaski | 500 | checking | WY |

| 43 | Tahirih Malinowski | 400 | checking | MS |

| 16 | Virginia Ruan | 400 | checking | HI |

| 50 | Dusan Mallory | 350 | savings | NV |

+----+--------------------+---------+----------+--------------+

限制行数和分页

下面举了一些例子，更多详细信息查看 [Limiting Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/limit-query-results/).

限制检索结果的数量

您可以使用LIMIT减少结果数量。

> **SELECT** id, name

**FROM** accounts

**LIMIT** 5;

+----+------------------+

| id | name |

+----+------------------+

| 1 | Bjorn Fairclough |

| 2 | Bjorn Fairclough |

| 3 | Arturo Nevin |

| 4 | Arturo Nevin |

| 5 | Naseem Joossens |

+----+------------------+

通过limit后的结果分页

如果要限制结果数量，但超出初始设置，请使用OFFSET继续下一组结果。这通常用于对并非所有值都需要立刻检索的需求进行分页。

> **SELECT** id, name

**FROM** accounts

**LIMIT** 5

OFFSET 5;

+----+------------------+

| id | name |

+----+------------------+

| 6 | Juno Studwick |

| 7 | Juno Studwick |

| 8 | Eutychia Roberts |

| 9 | Ricarda Moriarty |

| 10 | Henrik Brankovic |

+----+------------------+

组合性

选择子句在选择查询的上下文中定义。table表达式在SELECT的FROM子句的上下文中定义。然而，它们可以相互集成以形成更复杂的查询或语句。

使用任何选择子句作为选择查询

任何选择子句都可以用作选择查询而不做任何更改。

例如，构造SELECT \* FROM accounts是一个选择子句，它也是一个有效的选择查询，因此可以通过附加分号作为独立语句使用：

> **SELECT** \* **FROM** accounts;

+----+-----------------------+---------+----------+--------------+

| id | name | balance | type | state\_opened |

+----+-----------------------+---------+----------+--------------+

| 1 | Bjorn Fairclough | 1200 | checking | AL |

| 2 | Bjorn Fairclough | 2500 | savings | AL |

| 3 | Arturo Nevin | 250 | checking | AK |

[ truncated ]

+----+-----------------------+---------+----------+--------------+

同样，构造VALUES（1），（2），（3）也是一个选择子句，因此也可以单独用作选择查询：

> VALUES (1), (2), (3);

+---------+

| column1 |

+---------+

| 1 |

| 2 |

| 3 |

+---------+

(3 rows)

使用任何table表达式作为选择子句

任何table表达式都可以用作选择子句（也是选择查询），方法是在其前面加上TABLE或将其用作SELECT \* FROM的操作数。

例如，表名customer是一个table表达式，用于指定该表中的所有行。表达式TABLE accounts和SELECT \* FROM account都是有效的选择子句。

同样，SQL连接表达式customers c JOIN orders o ON c.id = o.customer\_id是一个table表达式。 您可以将其转换为有效的选择子句，从而将其转换为有效的选择查询，如下所示：

> TABLE (customers c JOIN orders o ON c.id = o.customer\_id);

> **SELECT** \* **FROM** customers c **JOIN** orders o **ON** c.id = o.customer\_id;

使用任何选择查询作为table表达式

任何选择查询（或选择子句）都可以用作table表达式，方法是将它括在括号之间，形成子查询。

例如，以下是选择查询，但不是有效的table表达式：

> **SELECT** \* **FROM** customers **ORDER** **BY** name **LIMIT** 5

要使它作为FROM或另一个table表达式的操作数，可以将它括在括号中，如下所示：

> **SELECT** id **FROM** (**SELECT** \* **FROM** customers **ORDER** **BY** name **LIMIT** 5);

> **SELECT** o.id

**FROM** orders o

**JOIN** (**SELECT** \* **FROM** customers **ORDER** **BY** name **LIMIT** 5) **AS** c

**ON** o.customer\_id = c.id;

使用其他语句的选择查询

选择查询也可以作为需要查询表数据的上下文中的有效操作数。

例如：

| **Statement** | **Example using SELECT** | **Example using VALUES** | **Example using TABLE** |
| --- | --- | --- | --- |
| [INSERT](http://doc.cockroachchina.baidu.com/#develop/constraints/insert/) | INSERT INTO foo SELECT \* FROM bar | INSERT INTO foo VALUES (1), (2), (3) | INSERT INTO foo TABLE bar |
| [UPSERT](http://doc.cockroachchina.baidu.com/#develop/sql-statements/UPSERT/) | UPSERT INTO foo SELECT \* FROM bar | UPSERT INTO foo VALUES (1), (2), (3) | UPSERT INTO foo TABLE bar |
| [CREATE TABLE AS](https://www.cockroachlabs.com/docs/stable/create-table-as.html) | CREATE TABLE foo AS SELECT \* FROM bar | CREATE TABLE foo AS VALUES (1),(2),(3) | CREATE TABLE foo AS TABLE bar |
| [ALTER ... SPLIT AT](http://doc.cockroachchina.baidu.com/#develop/sql-statements/SPLIT-AT/) | ALTER TABLE foo SPLIT AT SELECT \* FROM bar | ALTER TABLE foo SPLIT AT VALUES (1),(2),(3) | ALTER TABLE foo SPLIT AT TABLE bar |
| Subquery in a [table expression](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/table-expressions/) | SELECT \* FROM (SELECT \* FROM bar) | SELECT \* FROM (VALUES (1),(2),(3)) | SELECT \* FROM (TABLE bar) |
| Subquery in a [scalar expression](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/scalar-expressions/) | SELECT \* FROM foo WHERE x IN (SELECT \* FROM bar) | SELECT \* FROM foo WHERE x IN (VALUES (1),(2),(3)) | SELECT \* FROM foo WHERE x IN (TABLE bar) |

已知限制

在未来的CockroachDB版本中可能会解除以下限制。

在通用table表达式中使用values子句

在VALUES子句内的子查询中，还不可能在VALUES子句外部使用通用table表达式，例如：

**WITH** a **AS** (...) VALUES ((**SELECT** \* **FROM** a));

使用具有通用table表达式的Set操作

现在还不可能在set运算符的右操作数中使用在set表达式之外定义的通用table表达式，例如：

WITH a AS (**SELECT** 1)

**SELECT** \* **FROM** users **UNION** **SELECT** \* **FROM** a; *-- "a" used on the right, not yet supported.*

对于UNION，您可以通过交换操作数来解决此限制，对于其他集合运算符，您可以在右侧操作数内部定义CTE。

See Also

* [Simple SELECT Clause](http://doc.cockroachchina.baidu.com/#develop/sql-statements/SELECT/)
* [Table Expressions](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/table-expressions/)
* [Ordering Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/ordering-query-results/)
* [Limiting Query Results](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/limit-query-results/)