UPSERT 语句是 [INSERT ON CONFLICT](http://doc.cockroachchina.baidu.com/#develop/sql-statements/INSERT/#on-conflict-clause)语句的简称。当UPSERT的指定值不违反唯一性约束时，插入数据，如果违反了唯一性约束，则更新这行数据。

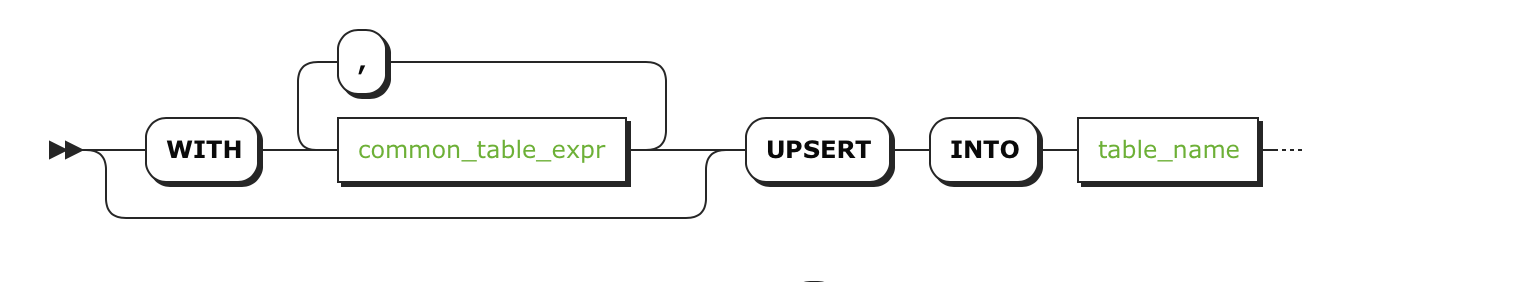
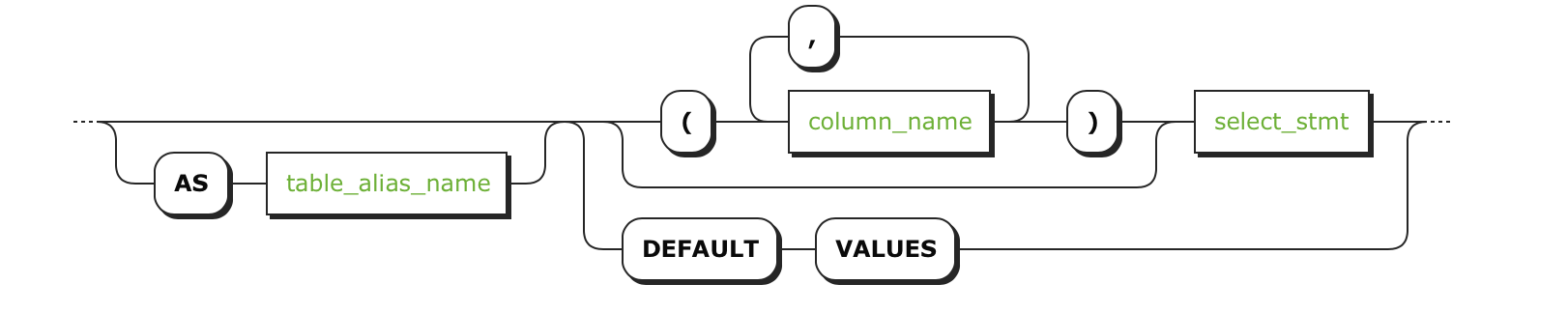
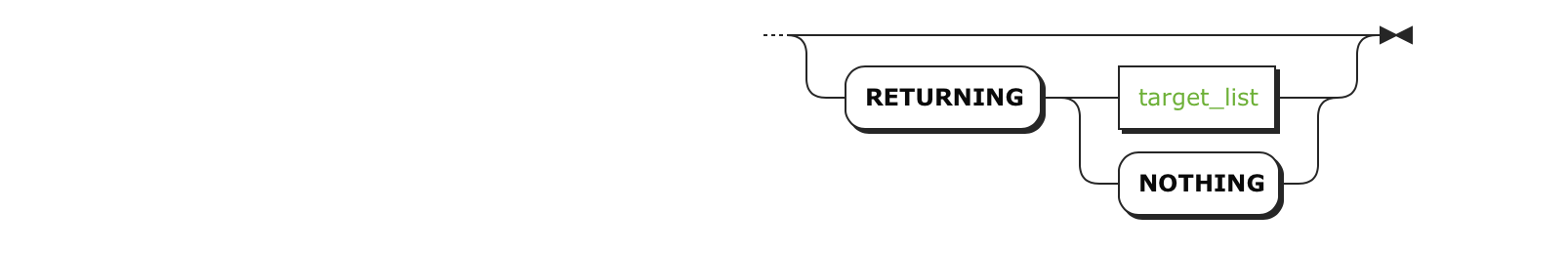
注意事项

* UPSERT仅仅考虑 [Primary Key](http://doc.cockroachchina.baidu.com/#develop/constraints/primary-key/) 列的唯一性约束。而INSERT ON CONFLICT是更加灵活的，可以考虑其他列的唯一性约束。更多相关信息，请看 [How UPSERT Transforms into INSERT ON CONFLICT](http://doc.cockroachchina.baidu.com/#develop/sql-statements/UPSERT/#how-upsert-transforms-into-insert-on-conflict) 。
* 插入/更新表的所有行数据，且表没有二级索引时，UPSERT语句比INSERT ON CONFLICT语句更快，因为UPSERT语句会直接写入。可以使用两列简单的SQL表来模拟[simulate direct KV access](https://www.cockroachlabs.com/docs/stable/frequently-asked-questions.html#can-i-use-cockroachdb-as-a-key-value-store)。
* 一个 多行 UPSERT语句的性能比多个单更新的UPSERT 语句快。所以，尽可能使用 多行更新的 UPSERT，而不是多个一行更新的 UPSERT语句。

所需权限

用户必须拥有操作表的 INSERT 和 UPDATE [privileges](http://doc.cockroachchina.baidu.com/#deploy/access-management/privileges/)。

概要

参数

| **Parameter** | **Description** |
| --- | --- |
| common\_table\_expr | 请看[Common Table Expressions](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/common-table-expressions/). |
| table\_name | 表的名字 |
| AS table\_alias\_name | 表的别名，使用别名之后，会隐藏真正的表名。 |
| column\_name | 插入期间要填充的列的名字 |
| select\_stmt | 一个[selection query](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/selection-queries/)。该查询返回的值的 [data type](http://doc.cockroachchina.baidu.com/#develop/data-types/overview/)必须与 =左边的列匹配。而且，如果在INTO之后列出了列名，则值必须按照对应的列排列；否则，值需要按照表中列的声明顺序进行排列。 |
| DEFAULT VALUES | 为了给所有列填充 [default values](http://doc.cockroachchina.baidu.com/#develop/constraints/default-value/), 在select\_stmt中使用DEFAULT VALUES。为了使用默认值填充特定列，在该值保留在a\_expr之外，或者在合适的位置使用DEFAULT。 |
| RETURNING target\_list | 基于插入值的返回值,在target\_list中可以是表中具体的列名，其中，\* 表示所有列，也可以是一个使用[scalar expressions](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/scalar-expressions/)的计算。  在一个[事务](http://doc.cockroachchina.baidu.com/#develop/transactions/)中, 使用 RETURNING NOTHING 不返回任何值。 |

如何将UPSERT 转化成 INSERT ON CONFLICT

UPSERT 仅仅考虑 [primary key](/develop/constraints/primary-key/列的唯一性。例如，假定a 和 b是主键列，一下的的 UPSERT与INSERT ON CONFLICT 是一样的：

> UPSERT INTO t (a, b, c) VALUES (1, 2, 3);

> **INSERT** **INTO** t (a, b, c)

**VALUES** (1, 2, 3)

**ON** CONFLICT (a, b)

**DO** **UPDATE** **SET** c = excluded.c;

INSERT ON CONFLICT 是更加灵活的，可以考虑非主键列的唯一性。更多相关信息，请看 [Upsert that Fails (Conflict on Non-Primary Key)](http://doc.cockroachchina.baidu.com/#develop/sql-statements/UPSERT/#upsert-that-fails-conflict-on-non-primary-key) 。

例子

Upsert 一行(无冲突)

在这个例子中，id列是主键，由于UPSERT语句中的id值与当前表中的值不冲突，所以会直接插入表中。

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

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

| id | balance |

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

| 1 | 10000.5 |

| 2 | 20000.75 |

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

> UPSERT INTO accounts (id, balance) VALUES (3, 6325.20);

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

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

| id | balance |

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

| 1 | 10000.5 |

| 2 | 20000.75 |

| 3 | 6325.2 |

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

Upsert 多行数据

在这个例子中，UPSERT语句插入多行数据。

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

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

| id | balance |

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

| 1 | 10000.5 |

| 2 | 20000.75 |

| 3 | 6325.2 |

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

> UPSERT INTO accounts (id, balance) VALUES (4, 1970.4), (5, 2532.9), (6, 4473.0);

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

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

| id | balance |

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

| 1 | 10000.5 |

| 2 | 20000.75 |

| 3 | 6325.2 |

| 4 | 1970.4 |

| 5 | 2532.9 |

| 6 | 4473.0 |

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

Upsert 更行一行数据 (在主键有冲突)

在这个例子中，id是主键，而且UPSERT语句中的id值在表中存在，所以UPSERT会更新列balance的值。

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

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

| id | balance |

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

| 1 | 10000.5 |

| 2 | 20000.75 |

| 3 | 6325.2 |

| 4 | 1970.4 |

| 5 | 2532.9 |

| 6 | 4473.0 |

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

> UPSERT INTO accounts (id, balance) VALUES (3, 7500.83);

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

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

| id | balance |

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

| 1 | 10000.5 |

| 2 | 20000.75 |

| 3 | 7500.83 |

| 4 | 1970.4 |

| 5 | 2532.9 |

| 6 | 4473.0 |

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

Upsert 失败(在非主键列中存在冲突 )

当UPSERT中的数据违反了非主键列的唯一性约束，UPSERT不会更新行中的数据。在这个例子中，a是主键列，而且b列有 [Unique constraint](http://doc.cockroachchina.baidu.com/#develop/constraints/unique/)。由于插入b列的值不是唯一的，所以UPSERT会失败。

> **SELECT** \* **FROM** unique\_test;

+---+---+

| a | b |

+---+---+

| 1 | 1 |

| 2 | 2 |

| 3 | 3 |

+---+---+

> UPSERT INTO unique\_test VALUES (4, 1);

pq: duplicate key value (b)=(1) violates unique constraint "unique\_test\_b\_key"

在这样一个例子中，你需要使用 [INSERT ON CONFLICT](http://doc.cockroachchina.baidu.com/#develop/sql-statements/INSERT/)语句指定b列是需要考虑的唯一性约束列。

> **INSERT** **INTO** unique\_test **VALUES** (4, 1) **ON** CONFLICT (b) **DO** **UPDATE** **SET** a = excluded.a;

> **SELECT** \* **FROM** unique\_test;

+---+---+

| a | b |

+---+---+

| 2 | 2 |

| 3 | 3 |

| 4 | 1 |

+---+---+

See Also

* [Selection Queries](http://doc.cockroachchina.baidu.com/#develop/sql-syntax/selection-queries/)
* [DELETE](http://doc.cockroachchina.baidu.com/#develop/sql-statements/DELETE/)
* [INSERT](http://doc.cockroachchina.baidu.com/#develop/sql-statements/INSERT/)
* [UPDATE](http://doc.cockroachchina.baidu.com/#develop/sql-statements/UPDATE/)
* [TRUNCATE](http://doc.cockroachchina.baidu.com/#develop/sql-statements/TRUNCATE/)
* [ALTER TABLE](http://doc.cockroachchina.baidu.com/#develop/sql-statements/ALTER-TABLE/)
* [DROP TABLE](http://doc.cockroachchina.baidu.com/#develop/sql-statements/DROP-TABLE/)
* [DROP DATABASE](http://doc.cockroachchina.baidu.com/#develop/sql-statements/DROP-DATABASE/)
* [Other SQL Statements](http://doc.cockroachchina.baidu.com/#develop/sql-statements/overview/)