**OceanBase SQL管理员手册[[1]](#footnote-1)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **编号** | **文档版本** | **修订章节** | **修订原因** | **修订日期** | **修订人** |
| **1** | 0.1 |  | 新建文档 | 2013/1/15 | 竹翁 |
| **2** | 0.2 |  | 细节修订 | 2013/1/21 | 竹翁 |
| **3** | 0.3 |  | 修订“系统自举和初始化”一章 | 2013/1/24 | 竹翁 |
| **4** | 0.4 |  | 简化“内部表”说明，引用外部文档；新增alter system set, show parameters, show create table语句说明。 | 2013/3/14 | 竹翁 |

[1. 用户及权限管理 3](#_Toc351037425)

[1.1. 用户及权限管理语句 3](#_Toc351037426)

[1.1.1. 新建用户 3](#_Toc351037427)

[1.1.2. 删除用户 3](#_Toc351037428)

[1.1.3. 修改密码 3](#_Toc351037429)

[1.1.4. 重命名用户 3](#_Toc351037430)

[1.1.5. 锁定用户 4](#_Toc351037431)

[1.1.6. 授予权限 4](#_Toc351037432)

[1.1.7. 撤回权限 5](#_Toc351037433)

[1.1.8. 查看权限 5](#_Toc351037434)

[1.2. 用例 6](#_Toc351037435)

[2. 系统变量和配置 6](#_Toc351037436)

[2.1. 修改系统变量 6](#_Toc351037437)

[2.2. 修改系统配置项 7](#_Toc351037438)

[用例 7](#_Toc351037439)

[2.3. 查看配置项的值 8](#_Toc351037440)

[用例 8](#_Toc351037441)

[3. 其他SQL语句 8](#_Toc351037442)

[3.1. show语句 8](#_Toc351037443)

[3.1.1. show columns 8](#_Toc351037444)

[3.1.2. show create table 9](#_Toc351037445)

[3.1.3. show tables 9](#_Toc351037446)

[3.1.4. show variables 9](#_Toc351037447)

[3.1.5. show parameters 9](#_Toc351037448)

[3.1.6. show warnings 9](#_Toc351037449)

[3.1.7. show grants 9](#_Toc351037450)

[3.2. describe语句 9](#_Toc351037451)

[3.3. explain语句 10](#_Toc351037452)

[3.4. alter system set语句 10](#_Toc351037453)

[4. 内部表 10](#_Toc351037454)

[5. 系统自举和初始化 10](#_Toc351037455)

[5.1. 步骤一：Bootstrap 10](#_Toc351037456)

[5.2. 步骤二：修改初始配置 11](#_Toc351037457)

[5.3. 步骤三：账户管理 11](#_Toc351037458)

[6. 实现上的限制 11](#_Toc351037459)

[7. SQL优化 12](#_Toc351037460)

[7.1. 执行计划 12](#_Toc351037461)

[7.2. 内部优化规则 12](#_Toc351037462)

[7.2.1. 主键索引 12](#_Toc351037463)

[7.2.2. 并发执行 13](#_Toc351037464)

# 用户及权限管理

## 用户及权限管理语句

### 新建用户

CREATE USER ***user\_specification***

[, ***user\_specification***] ...

***user\_specification***:

***user*** IDENTIFIED BY '***password***'

必须拥有全局的CREATE USER权限或对\_\_users表的INSERT权限，才可以使用CREATE USER命令。新建成功后，\_\_users表会新增一行该用户的表项，新建的用户没有任何权限。如果同名用户已经存在，则报错。

用户名格式与一个普通SQL字符串相同，如’Tom123’, ‘Han MeiMei’, 等。

此处密码为明文，存入\_\_users表后，服务器端会变为密文存储下来。

### 删除用户

DROP USER ***user*** [, ***user***] ...

必须拥有全局的CREATE USER权限或对\_\_users表的DELETE权限，才可以使用DROP USER命令。成功删除用户后，这个用户的**所有权限**也会被一同删除。

### 修改密码

SET PASSWORD [FOR ***user***] = '***password***'

或

ALTER USER ***user*** IDENTIFIED BY '***password***'

如果没有For user子句，则修改当前用户的密码。任何成功登陆的用户都可以修改当前用户的密码。

如果有For user子句，或使用第二种语法，则修改指定用户的密码。必须拥有对\_\_users表的UPDATE权限，才可以修改制定用户的密码。

### 重命名用户

RENAME USER ***old\_user*** TO ***new\_user***

[, ***old\_user*** TO ***new\_user***] ...

重命名指定用户的用户名，成功后，新用户拥有原用户的所有权限。必须拥有全局CREATE USER权限或者对\_\_users表的UPDATE权限，才可以使用本命令。

### 锁定用户

ALTER USER ***user*** ***lock\_specification***

***lock\_specification*** :

LOCKED

| UNLOCKED

锁定或者解锁用户。被锁定的用户不允许登陆。必须拥有对\_\_users表的UPDATE权限，才可以执行本命令。

### 授予权限

GRANT

***priv\_type*** [, ***priv\_type***] ...

ON ***priv\_level***

TO ***user\_specification*** [, ***user\_specification***] ...

***priv\_level***:

\*

| ***tbl\_name***

***user\_specification***:

***user***

给特定用户授予权限。如果用户不存在则报错。当前用户必须拥有被授予的权限，并且拥有GRANT OPTION权限，才能授予成功。

**可以GRANT 和REVOKE的权限**

| **权限** | **说明** |
| --- | --- |
| [**ALL [PRIVILEGES]**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_all) | 授予除了 [**GRANT OPTION**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_grant-option)以外所有权限 |
| [**ALTER**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_alter) | 授权使用 [**ALTER TABLE**](http://dev.mysql.com/doc/refman/5.0/en/alter-table.html) |
| [**CREATE**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_create) | 授权使用**CREATE TABLE** |
| [**CREATE USER**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_create-user) | 授权使用**[CREATE USER](http://dev.mysql.com/doc/refman/5.0/en/create-user.html" \o "13.7.1.1. CREATE USER Syntax)**, [**DROP USER**](http://dev.mysql.com/doc/refman/5.0/en/drop-user.html), [**RENAME USER**](http://dev.mysql.com/doc/refman/5.0/en/rename-user.html), and [**REVOKE ALL PRIVILEGES**](http://dev.mysql.com/doc/refman/5.0/en/revoke.html) |
| [**DELETE**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_delete) | 授权使用 [**DELETE**](http://dev.mysql.com/doc/refman/5.0/en/delete.html) |
| [**DROP**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_drop) | 授权使用**DROP TABLE** |
| [**GRANT OPTION**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_grant-option) | 授权使用**GRANT**和**REVOKE** |
| [**INSERT**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_insert) | 授权使用**[INSERT](http://dev.mysql.com/doc/refman/5.0/en/insert.html" \o "13.2.5. INSERT Syntax)** |
| [**SELECT**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_select) | 授权使用**[SELECT](http://dev.mysql.com/doc/refman/5.0/en/select.html" \o "13.2.8. SELECT Syntax)** |
| [**UPDATE**](http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_update) | 授权使用**[UPDATE](http://dev.mysql.com/doc/refman/5.0/en/update.html" \o "13.2.10. UPDATE Syntax)** |
| **REPLACE** | 授权使用**REPLACE** |
| **SUPER** | 授权使用**SET GLOBAL**修改全局系统参数 |

这些是基本的权限，以后会根据需求逐步增加新的权限。

本用户自动拥有自己创建的对象（目前基本上只有表）。例如，用户user1创建了表t1,t2,那用户user1应该自动就有对t1,t2的ALL PRIVILEGES及GRANT OPTION权限，不再需要额外去授权。

### 撤回权限

REVOKE

***priv\_type*** [, ***priv\_type***] ...

ON ***priv\_level***

FROM ***user*** [, ***user***] ...

REVOKE ALL PRIVILEGES, GRANT OPTION

FROM ***user*** [, ***user***] ...

第一种语句撤销指定用户的特定权限。对于第一种语法，当前用户必须拥有被撤销的权限，并且拥有GRANT OPTION权限。

第二种语句撤销指定用户的所有权限。对于这种语句，当前用户必须拥有全局GRANT OPTION权限，或者对权限表的UPDATE及DELETE权限。

注意，撤回操作**不会**级联。例如，用户user1给user2授予了某些权限，撤回user1的权限**不会**同时也撤回user2的相应权限。

### 查看权限

SHOW GRANTS [FOR ***user***]

如果不指定用户名，则缺省显示当前用户的权限。对于当前用户，总可以查看自己的权限。如果要查看其他指定用户的权限，必须拥有对权限表的SELECT权限。

## 用例

系统初始化后，只有一个管理员账户admin，初始密码admin。管理员用这组用户名密码登录系统。

1. 修改admin管理员的初始密码

SET PASSWORD ‘mypass’

1. 新建系统管理员账户并授予管理权限

CREATE USER ‘Li Lei’ IDENTIFIED BY ‘I love Han Meimei’

GRANT ALL PRIVILEGES, GRANT OPTION ON \* TO ‘Li Lei’

注意，管理员’Li Lei’拥有和admin相同的权限。

1. 新建应用管理员账户app1admin

CREATE USER ‘app1admin’ IDENTIFIED BY ‘app1pass’

GRANT ALL PRIVILEGES, GRANT OPTION ON t1 TO ‘app1admin’

1. app1admin新建普通用户账户

CREATE USER ‘app1user’ IDENTIFIED BY ‘app1user’

GRANT SELECT, INSERT, UPDATE, ALTER ON t1 TO ‘app1user’

1. 系统管理员轮值，对管理员账户’Li Lei’改名

RENAME USER ‘Li Lei’ TO ‘Han Meimei’

1. 撤销app1user账户部分权限

REVOKE ALTER ON t1 FROM ‘app1user’

SET PASSWORD FOR ‘app1user’ = ‘newpass’

1. app1的权限收归系统管理员，删除账户app1admin

DROP USER app1admin

# 系统变量和配置

**OceanBase目前区分系统变量和配置。**存储在\_\_all\_sys\_param表中的为全局系统变量，存储在\_\_all\_sys\_config中的称为配置项。系统变量一般和SQL功能相关，比如autocommit，tx\_isolation等。配置项一般针对每一类server进行配置，一般影响某个server的行为，比如MS的线程池大小，RS的负载均衡策略等，当然也可以通过指定IP对某个server单独进行配置。

## 修改系统变量

系统变量主要是与SQL有关的系统参数。每一个系统变量在SQL会话内都有一份本地拷贝。用户通过SET语句可以修改本地系统变量。管理员（要求拥有PRIV\_SUPER权限）可以通过SET GLOBAL语句修改全局的系统变量，修改后的全局系统变量影响之后所有会话的行为。SET语句的语法见《OceanBase SQL用户手册》相关章节。需要注意的是，管理员要用正确的数据类型设置相应的系统变量的值。

## 修改系统配置项

ALTER SYSTEM SET ***variable\_assignment*** [, ***variable\_assignment***] ...

***variable\_assignment***:

***param\_name*** = ***expr*** [COMMENT ***‘text’***][SCOPE=***conf\_scope***] SERVER\_TYPE = ***server\_type* [*cluster\_scope | server\_address*]**

conf\_scope:

MEMORY

| SPFILE

| BOTH

***server\_type***:

ROOTSERVER

| UPDATESERVER

| CHUNKSERVER

| MERGESERVER

***cluster\_scope***:

CLUSTER = ***cluster\_id***

***server\_address***:

SERVER\_IP***= ‘server\_ip’*** SERVER\_PORT***=server\_port***

COMMENT子句是可选的。但是修改配置项的时候，强烈建议添加关于本次修改的注释。

SCOPE用来指定本次配置项修改的生效范围。MEMORY表明只修改内存中的配置项，修改立即生效，且本修改在server重启以后会失效（目前暂时没有配置项支持这种方式）。SPFILE表明只修改配置表中的配置项值，当server重启以后才生效。BOTH表明既修改配置表，又修改内存值，修改立即生效，且server重启以后配置值仍然生效。SCOPE默认值为BOTH。对于不能立即生效的配置项，如果SCOPE使用BOTH或MEMORY，会报错[[2]](#footnote-2)。

如果指定了CLUSTER，表明本配置项的修改正对指定集群的特定server类型，否则，针对所有集群的特定server类型。

如果指定了server\_address，则只修改指定server实例（进程）的某个配置项。

#### 用例

修改所有RS的线程数和balance\_max\_migrate\_out\_per\_cs：

ALTER SYSTEM SET thread\_count=30 COMMENT ‘by zhuweng’ SCOPE=SPFILE SERVER\_TYPE=ROOTSERVER, balance\_max\_migrate\_out\_per\_cs=50 COMMENT ‘by zhuweng’ SERVER\_TYPE=ROOTSERVER

修改（调高）集群4的所有RS的线程数（例如，因为集群4的RS机器性能更好）：

ALTER SYSTEM SET thread\_count=30 COMMENT ‘by zhuweng’ SCOPE=SPFILE SERVER\_TYPE=ROOTSERVER CLUSTER=4

修改特定RS的线程数：

ALTER SYSTEM SET thread\_count=30 COMMENT ‘by zhuweng’ SCOPE=SPFILE SERVER\_TYPE=ROOTSERVER SERVER\_IP=’10.232.36.188’ SERVER\_PORT=1234

## 查看配置项的值

SHOW PARAMETERS

[LIKE '***pattern***' | WHERE ***expr***]

这个语句用来查询当前**已经生效**的系统参数，输出表的schema如下。

|  |  |  |
| --- | --- | --- |
| 列名 | 类型 | 说明 |
| cluster\_id | Int | server所属的机群ID |
| server\_type | vchar | server的类型：RS,UPS,MS,CS |
| ip | vchar | server的IP地址 |
| port | Int | server的IP地址端口 |
| name | vchar | 参数名称 |
| value | vchar | 参数值 |

like语句用来过滤name，等价于where name like。WHERE子句可以用来对输出表进行过滤，和普通select语句相同。

#### 用例

查询全部配置项：

show parameters

查询当前所有server进程的线程数：

show parameters like ‘%thread\_count%’

查询所有UPS的配置项：

show parameters where server\_type=’updateserver’

查询集群4的所有MS的配置：

show parameters where cluster\_id=4 and server\_type=’mergeserver’

# 其他SQL语句

## show语句

### show columns

SHOW COLUMNS {FROM | IN} ***tbl\_name***

[LIKE '***pattern***' | WHERE ***expr***]

### show create table

SHOW CREATE TABLE ***tbl\_name***

查看可以用来建立指定表格的建表语句。

### show tables

SHOW TABLES

[LIKE '***pattern***' | WHERE ***expr***]

### show variables

SHOW [GLOBAL | SESSION] VARIABLES

[LIKE '***pattern***' | WHERE ***expr***]

获得全局或会话的系统变量，默认为当前会话。

### show parameters

这个语句用来查看各个配置项在各个server实例上的值，见2.3。

### show warnings

SHOW WARNINGS [LIMIT [***offset***,] ***row\_count***]

SHOW COUNT(\*) WARNINGS

获得上一条语句执行过程中产生的warning信息，不包含error。

### show grants

见1.1.8。

## describe语句

{DESCRIBE | DESC} ***tbl\_name*** [***col\_name*** | ***wild***]

这个语句等同于SHOW COLUMNS FROM语句。

## explain语句

EXPLAIN [VERBOSE]

{ ***select\_stmt*** | ***insert\_stmt*** | ***update\_stmt*** | ***delete\_stmt*** | ***delete\_stmt*** }

这个语句可以输出select, insert, update, delete, replace等DML语句内部的物理执行计划。VERBOSE模式也会输出逻辑执行计划。DBA和开发人员可以根据explain的输出来优化SQL语句。

## alter system set语句

这个语句用来修改系统各个server的配置项，见2.2。

# 内部表

OceanBase内部表都以“\_\_”开头，普通用户表请不要使用这种格式的名字。原则上，OB开发人员保留在不同版本见增删内部表和修改它们schema的权利。所以普通用户和应用程序请不要依赖于这些表的任何内容。但是，OB还不成熟，有些特殊的操作需要管理员直接操作内部表。内部表的设计和schema请参考《OceanBase内部表定义》。

# 系统自举和初始化

OceanBase首次部署以后，需要管理员执行一些初始化步骤，然后才能对外提供服务。本章内容是介绍性的，一个包含手把手示例的教程请参考《OceanBase 0.4集群启动文档》。

## 步骤一：Bootstrap

系统自举（bootstrap）过程中，需要建立并初始化所有内部表。

* 启动整个集群的所有server
* **管理员使用rs\_admin向主集群的主RS发送boot\_strap指令**

为了满足各位看官的好奇心，一次典型的自举过程一般要经历如下事件：

1. 所有server向各自所在集群的主RS注册，要求主集群有MS、CS、UPS各至少一个
2. 过一段时间，RS会从注册过的UPS中选择一个UPS为主UPS
3. RS新建\_\_first\_tablet\_entry, \_\_all\_all\_column, \_\_all\_join\_info这三张核心表的tablet
4. RS通知所有server更新schema
5. RS新建\_\_all\_sys\_param和\_\_all\_sys\_stat等其余所有内部表的schema，并新建tablet
6. RS通知所有server更新schema
7. 如果有schema.ini文件，从中读取其他表格，新建它们的表schema。Schema.ini中的表格被认为都是有静态sstable数据的，不会自动新建tablet
8. 通知所有CS汇报tablet，这是为了导入有静态sstable数据的表格数据
9. 往\_\_all\_sys\_param, \_\_all\_sys\_stat, \_\_users等三张表中写入初始数据
10. 往\_\_trigger\_event中插入一个事件，通知备集群的RS bootstrap事件

## 步骤二：修改初始配置

第一次启动OB以后，所有server都是使用程序默认的配置值启动。自举完成以后，需要管理员修改默认的配置，以适配特定的部署关键。修改方法见4.2小节。

由于某些配置项无法动态生效（例如某个server线程池中线程数thread\_count），修改完成以后需要重启整个集群。

## 步骤三：账户管理

* 修改admin账户的初始密码
* 给应用创建新用户，授予恰当的权限

# 实现上的限制

由于历史和实现的原因，当前版本的OceanBase内部实现会对外表现出很多硬性的限制。注意，这些限制值可能随着OceanBase版本不同而改变，请查阅相应版本的手册。

|  |  |  |
| --- | --- | --- |
| 名称 | 值 | 说明 |
| MAX\_COLUMN\_NAME\_LENGTH | 127 | 列名最大长度 |
| MAX\_TABLE\_NAME\_LENGTH | 255 | 表名最大长度 |
| MAX\_VARCHAR\_LENGTH | 64K-512 | Varchar类型数据最大允许长度 |
| MAX\_ROW\_LENGTH | 2M-512K | 一行数据中varchar类型值的总最大长度 |
| MAX\_PACKET\_LENGTH | 2M | 网络包最大长度 |
| MAX\_USER\_DEFINED\_COLUMNS\_COUNT | 496 | 表可以定义的最大列数 |
| ROW\_MAX\_COLUMNS\_COUNT | 512 | 查询结果最大列数 |
| MAX\_ROWKEY\_COLUMN\_NUMBER | 16 | 组成主键的最大列数 |
| MAX\_PREPARE\_STMT\_NUM\_PER\_SESSION | 512 | 一个SQL会话中允许的prepared statement最大个数 |
| MAX\_VAR\_NUM\_PER\_SESSION | 1024 | 一个SQL会话中可以用SET语句定义的用户变量最大个数 |

# SQL优化

## 执行计划

Explain语句，可以查看一个DML语句的执行计划。执行计划是一个物理运算符组成的树状结构。常见的物理运算符有TableScan, Filter, Sort, GroupBy, Join, Project等。

例如，执行语句

explain select name, value1, value2 from \_\_all\_sys\_config\_stat where name = 'location\_cache\_timeout';

可以得到如下一个plan：

**Project**(columns=[expr<NULL,65519>=[COL<12,25>|],expr<NULL,65518>=[COL<12,27>|],expr<NULL,65517>=[COL<12,28>|]])

**TableRpcScan**(rpc\_scan=<RpcScan(tid=12 Filter(filters=[expr<NULL,65516>=[COL<12,25>|varchar:location\_cache\_timeout|EQ<2>|]])

Project(columns=[expr<12,25>=[COL<12,25>|],expr<12,27>=[COL<12,27>|],expr<12,28>=[COL<12,28>|]])

这个plan由两个物理运算符组成，Project运算符负责投影和表达式计算，它的输入数据由下层的TableRpcScan提供。Columns参数列出了三个表达式，表示这个投影操作会产生3列数据。expr<NULL,65519>=[COL<12,25>|]代表一个后缀表达式，这个表达式会产生一个TABLE\_ID为NULL，COLUMN\_ID为65519的cell。这个cell的值根据后缀表达式[COL<12,25>|]运算后产生，这里这个表达式是一个列引用，就是取TABLE\_ID为12，COLUMN\_ID为25的数据值。

第二个物理运算符是TableRpcScan，它实际上由RpcScan操作符实现。这个物理运算符实际上是在ChunkServer上执行。它读取TABLE\_ID为12的表中通过Project(columns=[expr<12,25>=[COL<12,25>|],expr<12,27>=[COL<12,27>|],expr<12,28>=[COL<12,28>|]]中指定COLUMN\_ID为25,27,28的3列数据。然后经过一个Filter操作符进行过滤，过滤条件即后缀表达式[COL<12,25>|varchar:location\_cache\_timeout|EQ<2>|]，它表示过滤TABLE\_ID为12，COLUMN\_ID为25的cell，等于varchar:location\_cache\_timeout的行。

## 内部优化规则

OceanBase为SQL语句产生执行计划的时候，实现了一些基于规则的优化策略，这里简单描述一些目前实现的规则。

### 主键索引

OceanBase表格中数据存储都是按照主键排序的。如果一个SELECT查询的WHERE条件中限定了主键的所有列，那么查询可以使用主键索引进行优化。

假设有一个表t1：create table t1 (c1 int, c2 int, c3 int, c4 int, primary key(c1, c2, c3))，下面举例说明优化规则：

1. 使用等值条件限定所有主键的查询，会使用MultiGet操作进行单行查询。如”select \* from t1 where c1 = 1 and c2 = 2 and c3 = 3”。注意，这里的等值条件必须写成列在等号左边，例如1 = c1则不满足本规则。
2. 使用IN表达式限定所有主键的查询，会使用MultiGet操作进行多行查询。如”select \* from t1 where (c1, c2, c3) IN ((1,2,3), (4,5,6))”。
3. 使用简单比较操作限定主键前缀列取值范围的查询，会转换为对某些特定Tablet的扫描。如”select \* from t1 where c1 >= 1 and c1 < 10 and c2 > 100 and c2 <= 200 and c3 >300”，会转换为对主键范围(<1,100,MIN>, <10, 200, MAX>)所有tablet的扫描。
4. 不满足上面三种情况的，则需要对整个表进行**全表扫描**。特别的，目前执行计划生产过程没有做表达式变化。所以”select \* from t1 where (c1 = 1 and c2 = 2 and c3 = 3) OR (c1 = 10 and c2 = 20 and c3 = 30)”这个语句不会使用MultiGet优化，应用应该使用IN表达式执行这种多行查询。

### 并发执行

MS向CS读取基本表数据的时候，是并发查询多个CS的。除此之外，如果满足某些条件，MS会把聚合操作分发到拥有数据的相关CS上执行，然后把CS汇总后的结果再做汇总。我们把这个优化叫做“聚合操作下压”，它需要查询满足以下一些条件：

1. 单表查询
2. 没有union, intersect, except等集合操作
3. 没有order by子句
4. 有group by子句，或者有聚集函数，且任何聚集函数不能有distinct修饰符
5. 最后，系统变量开关ob\_group\_agg\_push\_down\_param为true（默认值）

此外，limit操作也可以下压到CS端执行，以减少需要网络传输的数据。适用这个优化的查询需要满足一下条件：

1. 单表查询
2. 没有union, intersect, except等集合操作
3. 没有order by子句
4. 没有group by子句以及聚集函数
5. 最后，当然需要有limit子句

1. 适用与OceanBase 0.4.1版本 [↑](#footnote-ref-1)
2. 报错功能在0.4.1中尚未实现。 [↑](#footnote-ref-2)