**SQL92标准语句Mycat支持测试**

**（基于Oracle和MySQL数据库）**

2015-04-22

测试环境

操作系统：CentOS release 5.10 (Final)

CPU信息：Intel(R) Core(TM) i3-3110M CPU @ 2.40GHz

Oracle版本：10.2.0

MySQL版本：5.5.27，InnoDB 1.1.8

Mycat 版本信息：BuildTime 2015-03-18

MavenVersion 1.4-dev

目录

[一、数据定义命令测试 6](#_Toc417481622)

[1、CREATE TABLE 创建一个数据库表 6](#_Toc417481623)

[（1）Oracle环境测试 6](#_Toc417481624)

[（2）MySQL环境测试 7](#_Toc417481625)

[2、DROP TABLE 从数据库中删除表 8](#_Toc417481626)

[（1）Oracle环境测试 8](#_Toc417481627)

[（2）MySQL环境测试 9](#_Toc417481628)

[3、ALTER TABLE 修改数据库表结构 10](#_Toc417481629)

[（1）Oracle环境测试 11](#_Toc417481630)

[（2）MySQL环境测试 13](#_Toc417481631)

[4、CREATE VIEW 创建一个视图 14](#_Toc417481632)

[（1）Oracle环境测试 14](#_Toc417481633)

[（2）MySQL环境测试 15](#_Toc417481634)

[5、DROP VIEW 从数据库中删除视图 17](#_Toc417481635)

[（1）Oracle环境测试 17](#_Toc417481636)

[（2）MySQL环境测试 18](#_Toc417481637)

[6、CREATE INDEX 为数据库表创建一个索引 18](#_Toc417481638)

[（1）Oracle环境测试 19](#_Toc417481639)

[（2）MySQL环境测试 19](#_Toc417481640)

[7、DROP INDEX 从数据库中删除索引 21](#_Toc417481641)

[（1）Oracle环境测试 22](#_Toc417481642)

[（2）MySQL环境测试 23](#_Toc417481643)

[8、CREATE PROCEDURE 创建一个存储过程 24](#_Toc417481644)

[（1）Oracle环境测试 24](#_Toc417481645)

[（2）MySQL环境测试 25](#_Toc417481646)

[二、数据操作命令测试 25](#_Toc417481647)

[1、SELECT 从数据库表中检索数据行和列 25](#_Toc417481648)

[（1）Oracle环境测试 26](#_Toc417481649)

[（2）MySQL环境测试 26](#_Toc417481650)

[2、INSERT 向数据库表添加新数据行 27](#_Toc417481651)

[（1）Oracle环境测试 27](#_Toc417481652)

[（2）MySQL环境测试 28](#_Toc417481653)

[3、DELETE 从数据库表中删除数据行 29](#_Toc417481654)

[（1）Oracle环境测试 29](#_Toc417481655)

[（2）MySQL环境测试 29](#_Toc417481656)

[4、UPDATE 更新数据库表中的数据 30](#_Toc417481657)

[（1）Oracle 环境测试 30](#_Toc417481658)

[（2）MySQL环境测试 31](#_Toc417481659)

[5、TRUNCATE 删除表中所有行，仍保持表的完整性 32](#_Toc417481660)

[（1）Oracle环境测试 33](#_Toc417481661)

[（2）MySQL环境测试 33](#_Toc417481662)

[6、SELECT 查询条件关键字 34](#_Toc417481663)

[（1）Oracle环境测试 35](#_Toc417481664)

[（2）MySQL环境测试 38](#_Toc417481665)

[7、SELECT子查询与表连接查询 41](#_Toc417481666)

[（1）Oracle环境测试 42](#_Toc417481667)

[（2）MySQL环境测试 43](#_Toc417481668)

[8、FUNCTION 统计函数 45](#_Toc417481669)

[（1）Oracle环境测试 45](#_Toc417481670)

[（2）MySQL环境测试 47](#_Toc417481671)

[9、FUNCTION 字符串函数 49](#_Toc417481672)

[（1）Oracle环境测试 50](#_Toc417481673)

[（2）MySQL环境测试 54](#_Toc417481674)

[10、FUNCTION 日期函数 59](#_Toc417481675)

[（1）Oracle环境测试 60](#_Toc417481676)

[（2）MySQL环境测试 61](#_Toc417481677)

[三、SQL92测试总结 63](#_Toc417481678)

[1、测试问题汇总 63](#_Toc417481679)

[（1）DROP TABLE 从数据库中删除表 63](#_Toc417481680)

[（2）ALTER TABLE 修改数据库表结构 63](#_Toc417481681)

[（3）CREATE VIEW 创建一个视图 64](#_Toc417481682)

[（4）DROP VIEW 从数据库中删除视图 65](#_Toc417481683)

[（5）DROP INDEX 从数据库中删除索引（MySQL也不支持） 65](#_Toc417481684)

[（6）CREATE PROCEDURE 创建一个存储过程（MySQL也不支持） 65](#_Toc417481685)

[（7）DELETE 从数据库表中删除数据行 65](#_Toc417481686)

[（8）SELECT子查询与表连接查询 65](#_Toc417481687)

[（9）FUNCTION 字符串函数 66](#_Toc417481688)

[（10）FUNCTION 日期函数 66](#_Toc417481689)

[2、测试结论 66](#_Toc417481690)

# 一、数据定义命令测试

## 1、CREATE TABLE 创建一个数据库表

测试语句：

create table employee (id int not null primary key,name varchar(100),sharding\_id int not null);

explain create table company(id int not null primary key,name varchar(100));

测试结果：支持该语句

测试过程：

### （1）Oracle环境测试

#### 1）操作employee表

mysql> explain create table employee (id int not null primary key,name varchar(100),sharding\_id int not null);

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

|DATA\_NODE|SQL |

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

| dn1 | create table employee (id int not null primary key,name varchar(100),sharding\_id int not null) |

| dn2 | create table employee (id int not null primary key,name varchar(100),sharding\_id int not null) |

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

2 rows in set (0.00 sec)

mysql> create table employee (id int not null primary key,name varchar(100),sharding\_id int not null);

Query OK, 0 rows affected (0.09 sec)

OK!

#### 2）操作company表

mysql> explain create table company(id int not null primary key,name varchar(100));

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

|DATA\_NODE|SQL |

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

| dn1 | create table company(id int not null primary key,name varchar(100))

| dn2 | create table company(id int not null primary key,name varchar(100))

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

2 rows in set (0.00 sec)

mysql> create table company(id int not null primary key,name varchar(100));

Query OK, 0 rows affected (0.30 sec)

OK!

### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain create table employee (id int not null primary key,name varchar(100),sharding\_id int not null);

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

|DATA\_NODE|SQL |

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

| dn1 | create table employee (id int not null primary key,name varchar(100),sharding\_id int not null) |

| dn2 | create table employee (id int not null primary key,name varchar(100),sharding\_id int not null) |

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

2 rows in set (0.00 sec)

mysql> create table employee (id int not null primary key,name varchar(100),sharding\_id int not null);

Query OK, 0 rows affected (0.17 sec)

#### 2）操作company表

mysql> explain create table company(id int not null primary key,name varchar(100));

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

|DATA\_NODE|SQL |

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

| dn1 | create table company(id int not null primary key,name varchar(100))

| dn2 | create table company(id int not null primary key,name varchar(100))

| dn3 | create table company(id int not null primary key,name varchar(100))

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

3 rows in set (0.00 sec)

mysql> create table company(id int not null primary key,name varchar(100));

Query OK, 0 rows affected (0.32 sec)

## 2、DROP TABLE 从数据库中删除表

测试语句：explain drop table company;

测试结果：支持该语句

测试过程：

### （1）Oracle环境测试

#### 1）操作employee表

mysql> explain drop table employee;

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

| DATA\_NODE | SQL |

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

| dn1 | drop table employee |

| dn2 | drop table employee |

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

2 rows in set (0.00 sec)

mysql> drop table employee;

Query OK, 0 rows affected (0.17 sec)

OK!

#### 2）操作company表

mysql> explain drop table company;

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

| DATA\_NODE | SQL |

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

| dn1 | drop table company |

| dn2 | drop table company |

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

2 rows in set (0.02 sec)

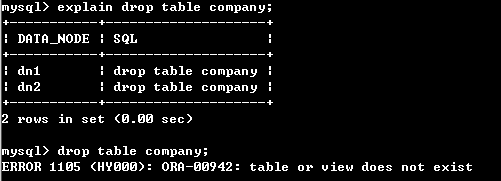
mysql> drop table company;

Query OK, 0 rows affected (3.95 sec)

OK!

注意：如果只有一个节点dn1有表company，另一个节点dn2没有（可能意外删除），执行该语句报错，但实际已经删除dn1上的company表。

java.sql.SQLException: ORA-00942: table or view does not exist





### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain drop table employee;

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

| DATA\_NODE | SQL |

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

| dn1 | drop table employee |

| dn2 | drop table employee |

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

2 rows in set (0.06 sec)

mysql> drop table employee;

Query OK, 0 rows affected (0.11 sec)

#### 2）操作company表

mysql> explain drop table company;

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

| DATA\_NODE | SQL |

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

| dn1 | drop table company |

| dn2 | drop table company |

| dn3 | drop table company |

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

3 rows in set (0.02 sec)

mysql> drop table company;

Query OK, 0 rows affected (0.36 sec)

## 3、ALTER TABLE 修改数据库表结构

测试语句：

--添加索引

explain alter table employee add index IDX\_EMP\_NAME (name);

create index IDX\_EMP\_NAME on employee(name);

--删除索引

explain alter table employee drop index IDX\_EMP\_NAME;

drop index IDX\_EMP\_NAME;

--添加唯一约束

explain alter table employee add unique EMP\_NAME(name);（MySQL写法）

explain alter table employee add constraint EMP\_UNIQUE unique(name);（Oracle写法）

--添加字段

explain alter table employee add age int;

--修改列名

explain alter table employee change name newname varchar(100);（MySQL写法）

explain alter table employee change name newname varchar2(100);

explain alter table employee rename column name to newname;（Oracle写法）

explain alter table employee rename name to newname;

--删除字段

explain alter table employee drop age;

--添加非空约束

explain alter table employee modify name constraint NAME\_NOT\_NULL not null; （Oracle写法）

explain alter table employee add constraint name check(name is not null); （MySQL语法）

explain alter table employee modify name varchar(2) not null;  （MySQL语法）

测试结果：

MySQL支持以上功能，Oracle只支持少部分功能，不支持添加唯一约，不支持修改列名，不支持删除字段

测试过程：

### （1）Oracle环境测试

mysql> alter table employee add index IDX\_EMP\_NAME (name);

ERROR 1105 (HY000): ORA-00904: invalid identifier

mysql> create index IDX\_EMP\_NAME on employee(name);

Query OK, 0 rows affected (0.09 sec)

OK!

mysql> alter table employee add unique EMP\_NAME(name);

ERROR 1105 (HY000): ORA-00906: missing left parenthesis

mysql> explain alter table employee add constraint EMP\_UNIQUE unique(name);

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: TODO UNIQUE unique

mysql> explain alter table employee add age int;

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

| DATA\_NODE | SQL |

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

| dn1 | alter table employee add age int |

| dn2 | alter table employee add age int |

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

2 rows in set (0.00 sec)

mysql> alter table employee add age int;

Query OK, 0 rows affected (0.01 sec)

OK!

mysql> alter table employee change name newname varchar2(100);

ERROR 1105 (HY000): ORA-01735: invalid ALTER TABLE option

mysql> alter table employee change name newname varchar(100);

ERROR 1105 (HY000): ORA-01735: invalid ALTER TABLE option

mysql> alter table employee rename column name to newname;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: error COLUMN

mysql> alter table employee rename name to newname;

ERROR 14155 (HY000): ORA-14155: missing PARTITION or SUBPARTITION keyword

mysql> alter table employee drop age;

ERROR 1105 (HY000): ORA-00905: missing keyword

mysql> explain alter table employee modify name constraint NAME\_NOT\_NULL not null;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: error CONSTRAINT

mysql> explain alter table employee add constraint name check(name is not null);

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: TODO CHECK check

mysql> explain alter table employee modify name varchar(2) not null;

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

| DATA\_NODE | SQL |

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

| dn1 | alter table employee modify name varchar(2) not null |

| dn2 | alter table employee modify name varchar(2) not null |

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

2 rows in set (0.00 sec)

mysql> alter table employee modify name varchar(2) not null;

Query OK, 0 rows affected (0.03 sec)

OK!

### （2）MySQL环境测试

mysql> alter table employee add index IDX\_EMP\_NAME (name);

Query OK, 0 rows affected (0.31 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> create index IDX\_EMP\_NAME3 on employee(name);

Query OK, 0 rows affected (0.26 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table employee drop index IDX\_EMP\_NAME;

Query OK, 0 rows affected (0.23 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> drop index IDX\_EMPLOYEE\_NAME;

ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' at line 1

mysql> alter table employee add unique EMP\_NAME(name);

Query OK, 0 rows affected (0.23 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> explain alter table employee add constraint EMP\_UNIQUE unique(name);

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: TODO UNIQUE unique

mysql> alter table employee add age int;

Query OK, 1 row affected (0.46 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table employee change name newname varchar(100);

Query OK, 1 row affected (0.39 sec)

Records: 1 Duplicates: 0 Warnings: 0

mysql> alter table employee drop age;

Query OK, 1 row affected (0.42 sec)

Records: 1 Duplicates: 0 Warnings: 0

mysql> explain alter table employee modify name constraint NAME\_NOT\_NULL not null;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: error CONSTRAINT

mysql> alter table employee modify name varchar(2) not null;

Query OK, 1 row affected (0.59 sec)

Records: 0 Duplicates: 0 Warnings: 0

## 4、CREATE VIEW 创建一个视图

测试语句：

explain create view v\_comp as select \* from company;

explain create view v\_emp as select \* from employee;

测试结果：支持该语句, 但须在schema.xml添加该视图（使用table标签）

测试过程：

### （1）Oracle环境测试

mysql> explain create view v\_comp as select \* from company;

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

| DATA\_NODE | SQL |

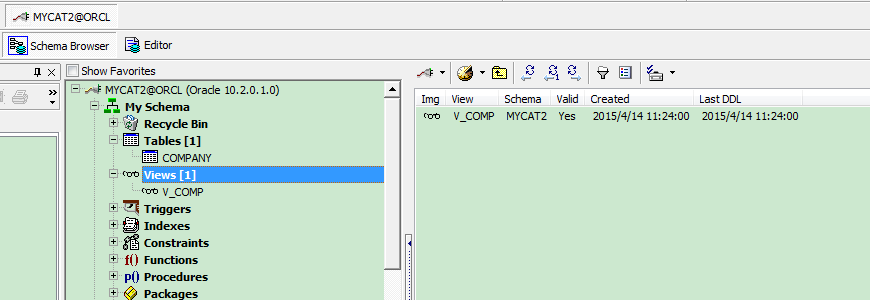
+-----------------+-----------------------------------------------------------+

| dn1 | create view v\_comp as select \* from company |

| dn2 | create view v\_comp as select \* from company |

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

1. rows in set (0.00 sec)



### （2）MySQL环境测试

在schema.xml中添加标签

<table name="v\_comp" primaryKey="ID" type="global" dataNode="dn1,dn2,dn3" />

mysql> explain create view v\_comp as select \* from company;

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

| DATA\_NODE | SQL |

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

| dn1 | create view v\_comp as select \* from company |

| dn2 | create view v\_comp as select \* from company |

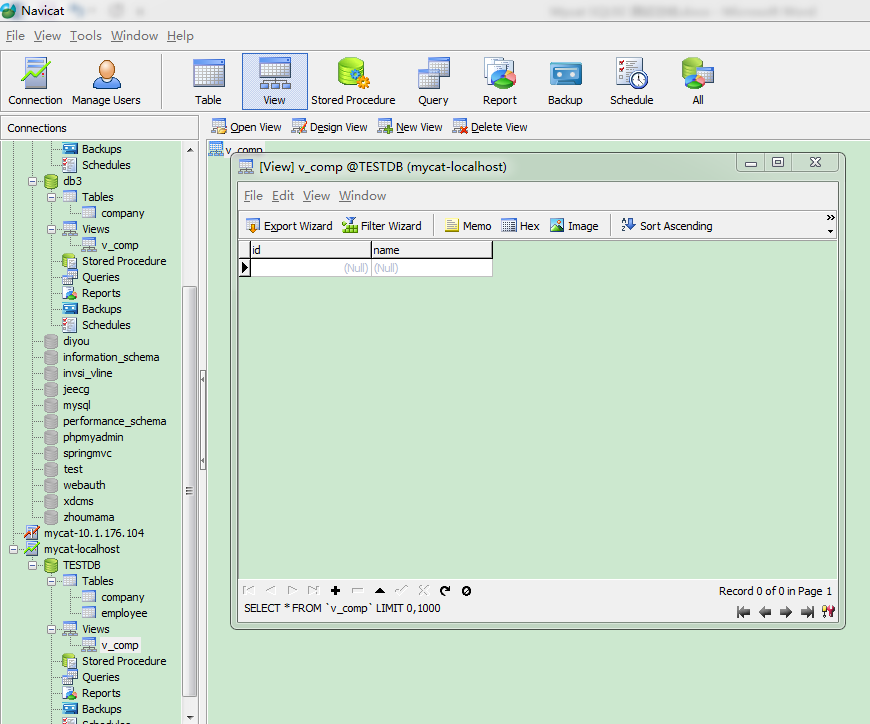
| dn3 | create view v\_comp as select \* from company |

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

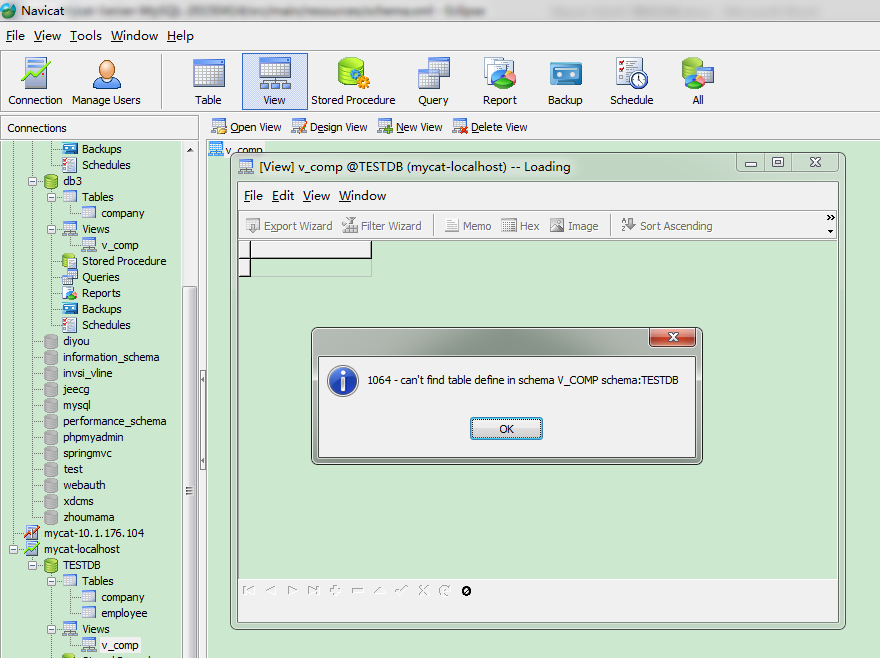
3 rows in set (0.01 sec)

mysql> create view v\_comp as select \* from company;

Query OK, 0 rows affected (0.14 sec)



若不加标签，也可以执行成功，但打开视图时报错，如下图所示：



## 5、DROP VIEW 从数据库中删除视图

测试语句：

explain drop view v\_comp;

测试结果：支持该语句，但须在schema.xml添加该视图（使用table标签），方法见下文

测试过程：

### （1）Oracle环境测试

在schema.xml中添加以下语句：

<table name="v\_comp" primaryKey="ID" type="global" dataNode="dn1,dn2" />

mysql> explain drop view v\_comp;

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

| DATA\_NODE | SQL |

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

| dn1 | drop view v\_comp |

| dn2 | drop view v\_comp |

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

2 rows in set (0.00 sec)



### （2）MySQL环境测试

在schema.xml中添加以下语句：

<table name="v\_comp" primaryKey="ID" type="global" dataNode="dn1,dn2,dn3" />

mysql> explain drop view v\_comp;

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

| DATA\_NODE | SQL |

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

| dn1 | drop view v\_comp |

| dn2 | drop view v\_comp |

| dn3 | drop view v\_comp |

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

3 rows in set (0.17 sec)

mysql> drop view v\_comp;

Query OK, 0 rows affected (0.01 sec)

## 6、CREATE INDEX 为数据库表创建一个索引

测试语句：

explain create index IDX\_COMPANY\_NAME on company(name);

explain create index IDX\_EMPLOYEE\_NAME on employee(name);

测试结果：支持该语句

测试过程：

### （1）Oracle环境测试

#### 1）操作employee表

mysql> explain create index IDX\_EMPLOYEE\_NAME on employee(name);

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

| DATA\_NODE | SQL |

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

| dn1 | create index IDX\_EMPLOYEE\_NAME on employee(name) |

| dn2 | create index IDX\_EMPLOYEE\_NAME on employee(name) |

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

2 rows in set (0.00 sec)

mysql> create index IDX\_EMPLOYEE\_NAME on employee(name);

Query OK, 0 rows affected (0.06 sec)

OK!

#### 2）操作company表

mysql> explain create index IDX\_COMPANY\_NAME on company(name);

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

| DATA\_NODE | SQL |

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

| dn1 | create index IDX\_COMPANY\_NAME on company(name) |

| dn2 | create index IDX\_COMPANY\_NAME on company(name) |

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

2 rows in set (0.00 sec)

mysql> create index IDX\_COMPANY\_NAME on company(name);

Query OK, 0 rows affected (0.05 sec)

OK!

### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain create index IDX\_EMPLOYEE\_NAME on employee(name);

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

| DATA\_NODE | SQL |

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

| dn1 | create index IDX\_EMPLOYEE\_NAME on employee(name) |

| dn2 | create index IDX\_EMPLOYEE\_NAME on employee(name) |

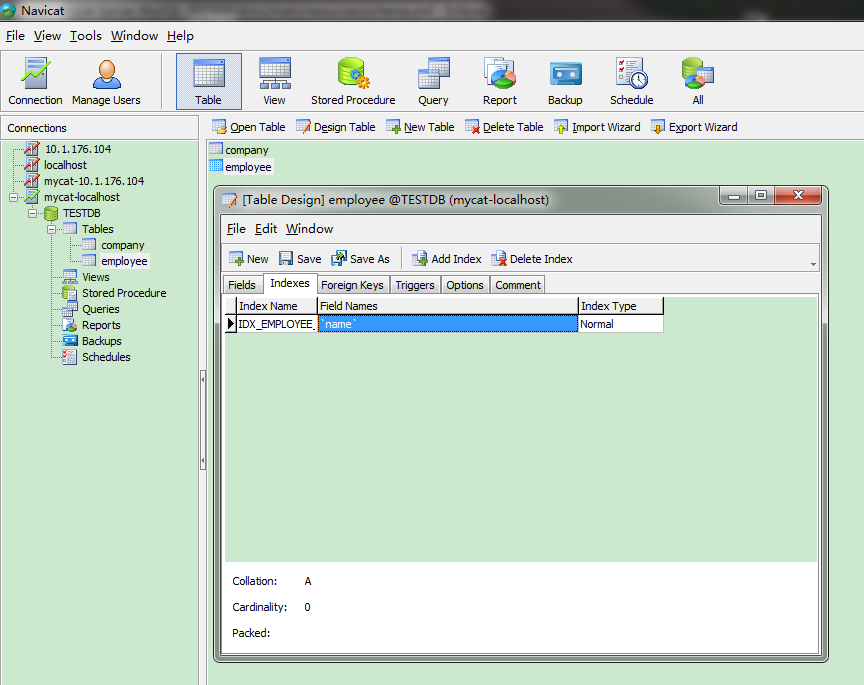
+-----------+--------------------------------------------------+

2 rows in set (0.00 sec)

mysql> create index IDX\_EMPLOYEE\_NAME on employee(name);

Query OK, 0 rows affected (0.30 sec)

Records: 0 Duplicates: 0 Warnings: 0



#### 2）操作company表

mysql> explain create index IDX\_COMPANY\_NAME on company(name);

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

| DATA\_NODE | SQL |

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

| dn1 | create index IDX\_COMPANY\_NAME on company(name) |

| dn2 | create index IDX\_COMPANY\_NAME on company(name) |

| dn3 | create index IDX\_COMPANY\_NAME on company(name) |

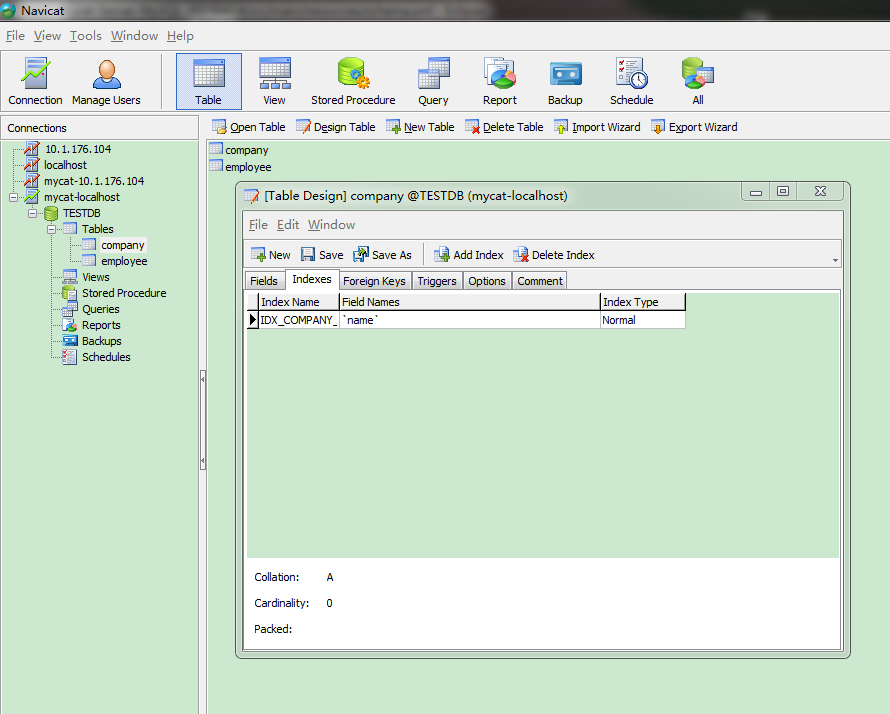
+-----------+------------------------------------------------+

3 rows in set (0.00 sec)

mysql> create index IDX\_COMPANY\_NAME on company(name);

Query OK, 0 rows affected (0.41 sec)

Records: 0 Duplicates: 0 Warnings: 0



## 7、DROP INDEX 从数据库中删除索引

测试语句：

explain drop index IDX\_COMPANY\_NAME;

explain drop index IDX\_EMPLOYEE\_NAME;

测试结果：存在异常，无法在全部节点同时执行删除，执行计划不确定。第一次执行命令在一个节点执行了删除索引，再次执行命令在另一个节点执行。Oracle和MySQL都存在该问题。

测试过程：

### （1）Oracle环境测试

mysql> explain drop index IDX\_COMPANY\_ID;

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

| DATA\_NODE | SQL |

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

| dn2 | drop index IDX\_COMPANY\_ID |

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

1 row in set (0.00 sec)

mysql> explain drop index IDX\_COMPANY\_ID;

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

| DATA\_NODE | SQL |

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

| dn1 | drop index IDX\_COMPANY\_ID |

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

1 row in set (0.00 sec)

mysql> drop index IDX\_COMPANY\_ID;

Query OK, 0 rows affected (0.10 sec)

OK!

在另一个节点执行

mysql> explain drop index IDX\_COMPANY\_ID;

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

| DATA\_NODE | SQL |

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

| dn1 | drop index IDX\_COMPANY\_ID |

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

1 row in set (0.00 sec)

mysql> explain drop index IDX\_COMPANY\_ID;

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

| DATA\_NODE | SQL |

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

| dn2 | drop index IDX\_COMPANY\_ID |

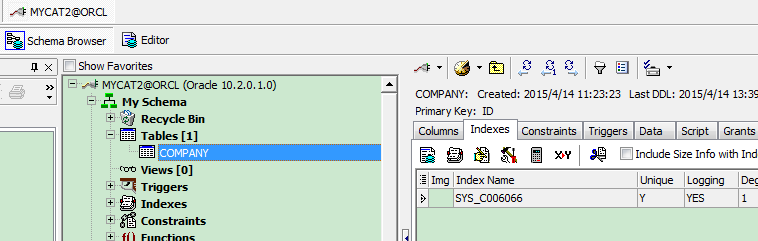
+-----------+---------------------------+

1 row in set (0.00 sec)

mysql> drop index IDX\_COMPANY\_ID;

Query OK, 0 rows affected (0.07 sec)

OK!



### （2）MySQL环境测试

mysql> explain drop index IDX\_EMPLOYEE\_NAME;

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

| DATA\_NODE | SQL |

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

| dn1 | drop index IDX\_EMPLOYEE\_NAME |

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

1 row in set (0.00 sec)

mysql> explain drop index IDX\_EMPLOYEE\_NAME;

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

| DATA\_NODE | SQL |

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

| dn2 | drop index IDX\_EMPLOYEE\_NAME |

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

1 row in set (0.00 sec)

mysql> explain drop index IDX\_COMPANY\_NAME;

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

| DATA\_NODE | SQL |

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

| dn1 | drop index IDX\_COMPANY\_NAME |

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

1 row in set (0.00 sec)

mysql> explain drop index IDX\_COMPANY\_NAME;

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

| DATA\_NODE | SQL |

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

| dn2 | drop index IDX\_COMPANY\_NAME |

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

1 row in set (0.00 sec)

## 8、CREATE PROCEDURE 创建一个存储过程

测试语句：

create or replace procedure mycat2.pro\_insertComp is

begin

insert into company(id,name) values(3,'tiens');

commit;

end pro\_insertComp;

CREATE PROCEDURE GreetWorld() SELECT CONCAT(@greeting,'World');

SET @greeting='Hello';

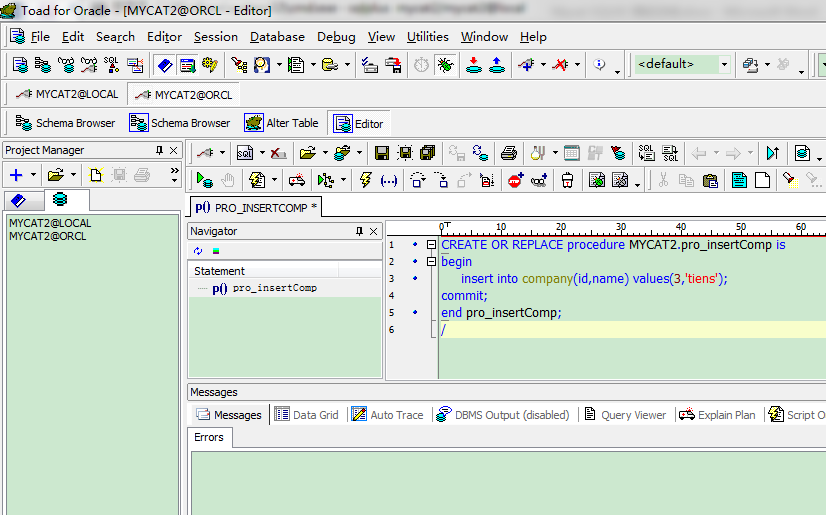
CALL GreetWorld();

测试结果：不支持创建存储过程

测试过程：

### （1）Oracle环境测试

单独在一个数据库节点中，执行存储计划可以成功，但在Mycat中执行失败。



mysql> CREATE OR REPLACE procedure MYCAT2.pro\_insertComp is

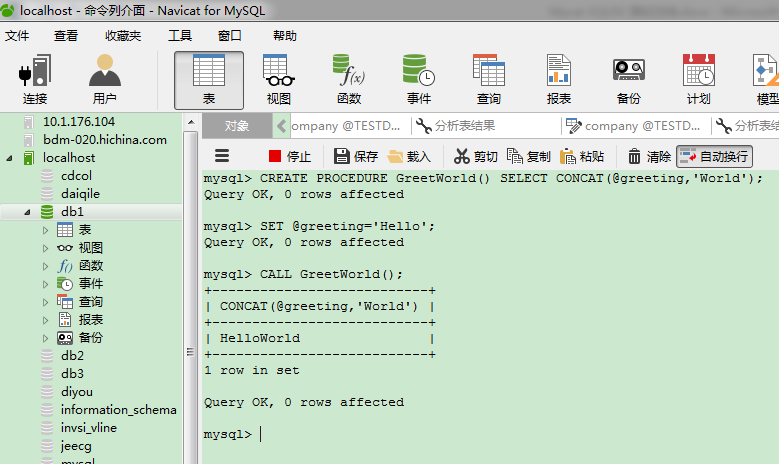
-> begin

-> insert into company(id,name) values(3,'tiens');

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: TODO PROCEDURE

### （2）MySQL环境测试

单独在一个数据库节点中，执行存储计划可以成功，但在Mycat中执行失败。



mysql> CREATE PROCEDURE GreetWorld() SELECT CONCAT(@greeting,'World');

-> SET @greeting='Hello';

-> CALL GreetWorld();

-> CALL GreetWorld();

# 二、数据操作命令测试

## 1、SELECT 从数据库表中检索数据行和列

测试语句：

explain select \* from employee;

explain select \* from company;

测试结果：支持该语句

测试过程：

### （1）Oracle环境测试

#### 1）操作employee表

mysql> explain select \* from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee WHERE ROWNUM <= 100 |

| dn2 | SELECT \* FROM employee WHERE ROWNUM <= 100 |

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

2 rows in set (0.36 sec)

mysql> select \* from employee;

Empty set (0.18 sec)

#### 2）操作company表

全局表在每一个节点的数据相同，因此可在任意节点执行查询。

mysql> explain select \* from company;

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

| DATA\_NODE | SQL |

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

| dn2 | SELECT \* FROM company LIMIT 100 |

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

1 row in set (0.00 sec)

mysql> select \* from company;

Empty set (0.00 sec)

### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain select \* from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee LIMIT 100 |

| dn2 | SELECT \* FROM employee LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee;

Empty set (0.00 sec)

#### 2）操作company表

mysql> explain select \* from company;

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

| DATA\_NODE | SQL |

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

| dn2 | SELECT \* FROM company WHERE ROWNUM <= 100 |

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

1 row in set (0.00 sec)

mysql> select \* from company;

Empty set (0.01 sec)

## 2、INSERT 向数据库表添加新数据行

测试语句：

explain insert into employee(id,name,sharding\_id) values(1,'znl',10000);

explain insert into company(id,name) values(1,'znl');

测试结果：支持该语句

测试过程：

### （1）Oracle环境测试

#### 1）操作employee表

mysql> explain insert into employee(id,name,sharding\_id) values(1,'znl',10000);

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

| DATA\_NODE | SQL |

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

| dn1 | insert into employee(id,name,sharding\_id) values(1,'znl',10000) |

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

1 row in set (0.09 sec)

#### 2）操作company表

mysql> explain insert into company(id,name) values(1,'znl');

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

| DATA\_NODE | SQL |

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

| dn1 | insert into company(id,name) values(1,'znl') |

| dn2 | insert into company(id,name) values(1,'znl') |

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

2 rows in set (0.00 sec)

### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain insert into employee(id,name,sharding\_id) values(1,'znl',10000);

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

| DATA\_NODE | SQL |

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

| dn1 | insert into employee(id,name,sharding\_id) values(1,'znl',10000) |

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

1 row in set (0.00 sec)

mysql> insert into employee(id,name,sharding\_id) values(1,'znl',10000);

Query OK, 1 row affected (0.08 sec)

#### 2）操作company表

mysql> explain insert into company(id,name) values(1,'znl');

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

| DATA\_NODE | SQL |

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

| dn1 | insert into company(id,name) values(1,'znl') |

| dn2 | insert into company(id,name) values(1,'znl') |

| dn3 | insert into company(id,name) values(1,'znl') |

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

3 rows in set (0.00 sec)

mysql> insert into company(id,name) values(1,'znl');

Query OK, 3 rows affected (0.10 sec)

## 3、DELETE 从数据库表中删除数据行

测试语句：explain delete from employee where id=1;

explain delete from company where id=1;

测试结果：Oracle环境不支持该语句，MySQL环境支持该语句

测试过程：

### （1）Oracle环境测试

mysql> explain delete from employee where id=1;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: ERROR. token : DELETE, pos :6



### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain delete from employee where id=1;

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

| DATA\_NODE | SQL |

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

| dn2 | delete from employee where id=1 |

| dn1 | delete from employee where id=1 |

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

2 rows in set (0.04 sec)

mysql> delete from employee where id=1;

Query OK, 1 row affected (0.06 sec)

#### 2）操作company表

mysql> explain delete from company where id=1;

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

| DATA\_NODE | SQL |

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

| dn1 | delete from company where id=1 |

| dn2 | delete from company where id=1 |

| dn3 | delete from company where id=1 |

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

3 rows in set (0.00 sec)

mysql> delete from company where id=1;

Query OK, 3 rows affected (0.10 sec)

## 4、UPDATE 更新数据库表中的数据

测试语句：

explain update employee set name='newname' where id=1;

explain update company set name='newname' where id=1;

测试结果：支持该语句

测试过程：

### （1）Oracle 环境测试

#### 1）操作employee表

mysql> explain update employee set name='newname' where id=1;

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

| DATA\_NODE | SQL |

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

| dn2 | update employee set name='newname' where id=1 |

| dn1 | update employee set name='newname' where id=1 |

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

2 rows in set (0.00 sec)

mysql> update employee set name='newname' where id=1;

Query OK, 0 rows affected (0.31 sec)

OK!

#### 2）操作company表

mysql> explain update company set name='newname' where id=1;

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

| DATA\_NODE | SQL |

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

| dn1 | update company set name='newname' where id=1 |

| dn2 | update company set name='newname' where id=1 |

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

2 rows in set (0.00 sec)

mysql> update company set name='newname' where id=1;

Query OK, 0 rows affected (0.02 sec)

OK!

### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain update employee set name='newname' where id=1;

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

| DATA\_NODE | SQL |

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

| dn2 | update employee set name='newname' where id=1 |

| dn1 | update employee set name='newname' where id=1 |

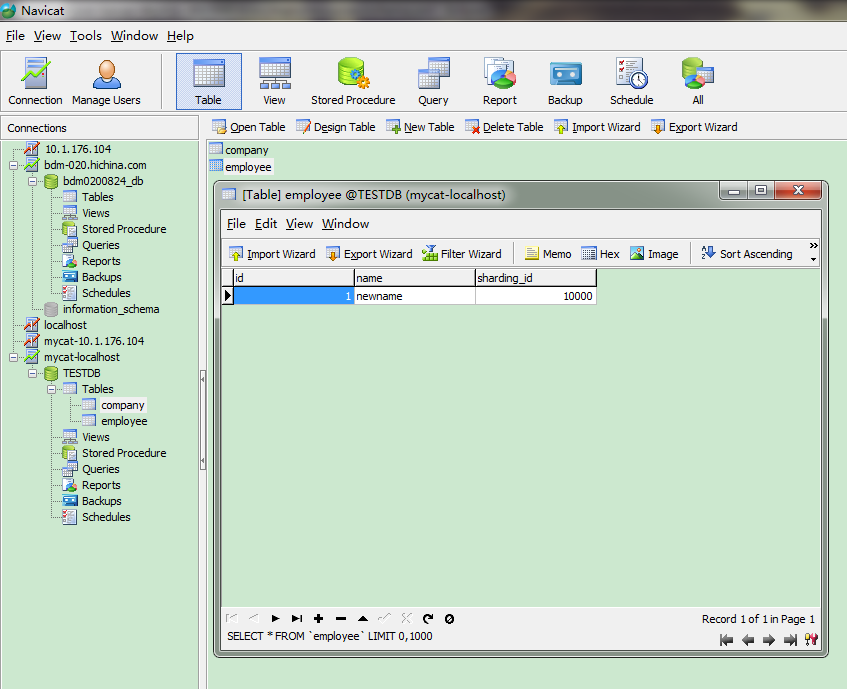
+-----------+-----------------------------------------------+

2 rows in set (0.04 sec)

mysql> update employee set name='newname' where id=1;

Query OK, 1 row affected (0.04 sec)

Rows matched: 1 Changed: 1 Warnings: 0



#### 2）操作company表

mysql> explain update company set name='newname' where id=1;

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

| DATA\_NODE | SQL |

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

| dn1 | update company set name='newname' where id=1 |

| dn2 | update company set name='newname' where id=1 |

| dn3 | update company set name='newname' where id=1 |

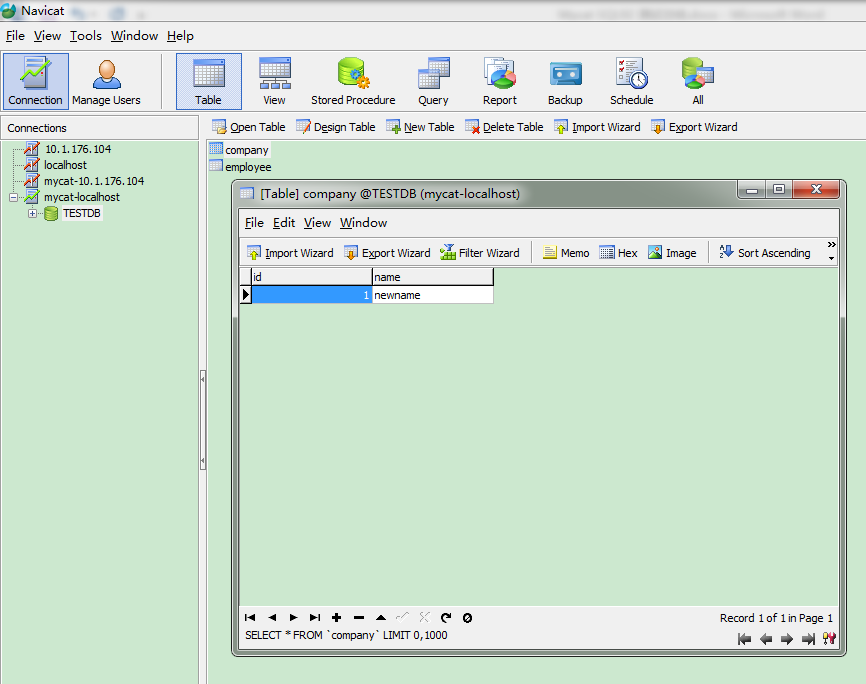
+-----------+----------------------------------------------+

3 rows in set (0.00 sec)

mysql> update company set name='newname' where id=1;

Query OK, 3 rows affected (0.11 sec)

Rows matched: 1 Changed: 1 Warnings: 0



## 5、TRUNCATE 删除表中所有行，仍保持表的完整性

测试语句：explain truncate table employee;

explain truncate table company;

测试结果：支持该语句

测试过程：

### （1）Oracle环境测试

#### 1）操作employee表

mysql> explain truncate table employee;

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

| DATA\_NODE | SQL |

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

| dn1 | truncate table employee |

| dn2 | truncate table employee |

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

2 rows in set (0.00 sec)

mysql> truncate table employee;

Query OK, 0 rows affected (0.04 sec)

OK!

#### 2）操作company表

mysql> explain truncate table company;

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

| DATA\_NODE | SQL |

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

| dn1 | truncate table company |

| dn2 | truncate table company |

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

2 rows in set (0.00 sec)

mysql> truncate table company;

Query OK, 0 rows affected (0.01 sec)

OK!

### （2）MySQL环境测试

#### 1）操作employee表

mysql> explain truncate table employee;

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

| DATA\_NODE | SQL |

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

| dn1 | truncate table employee |

| dn2 | truncate table employee |

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

2 rows in set (0.00 sec)

mysql> truncate table employee;

Query OK, 0 rows affected (0.30 sec)

#### 1）操作company表

mysql> explain truncate table company;

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

| DATA\_NODE | SQL |

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

| dn1 | truncate table company |

| dn2 | truncate table company |

| dn3 | truncate table company |

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

3 rows in set (0.00 sec)

mysql> truncate table company;

Query OK, 0 rows affected (0.21 sec)

## 6、SELECT 查询条件关键字

测试语句：

explain select \* from employee where id=1;

explain select \* from employee where name='a';

explain select \* from employee where name like '%a%';

explain select \* from employee where name='a' or name='b';

explain select \* from employee where id<100;

explain select \* from employee where id between 1 and 100;

explain select \* from employee where id in (1,3,5);

explain select \* from employee order by id asc;

explain select distinct name from employee;

explain select \* from employee group by name;

explain select \* from employee group by name having count(\*) = 2;

测试结果：支持以上命令，“distinct”命令转化为“group by”

测试过程：

### （1）Oracle环境测试

mysql> explain select \* from employee where id=1;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee where id=1 |

| dn1 | select \* from employee where id=1 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id=1;

Empty set (0.11 sec)

mysql> explain select \* from employee where name='a';

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

| DATA\_NODE | SQL |

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

| dn2 | SELECT \* FROM employee WHERE name = 'a' AND ROWNUM <= 100 |

| dn1 | SELECT \* FROM employee WHERE name = 'a' AND ROWNUM <= 100 |

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

2 rows in set (0.06 sec)

mysql> select \* from employee where name='a';

Empty set (0.12 sec)

mysql> explain select \* from employee where name like '%a%';

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee WHERE name LIKE '%a%'AND ROWNUM <= 100 |

| dn2 | SELECT \* FROM employee WHERE name LIKE '%a%'AND ROWNUM <= 100 |

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

2 rows in set (0.01 sec)

mysql> select \* from employee where name like '%a%';

Empty set (0.04 sec)

mysql> explain select \* from employee where name='a' or name='b';

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

|DATA\_NODE |SQL |

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

| dn2 | SELECT \* FROM employee WHERE (name = 'a' OR name = 'b') AND ROWNUM <= 100 |

| dn1 | SELECT \* FROM employee WHERE (name = 'a' OR name = 'b')AND ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where name='a' or name='b';

Empty set (0.01 sec)

mysql> explain select \* from employee where id<100;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee WHERE id < 100 AND ROWNUM <= 100 |

| dn2 | SELECT \* FROM employee WHERE id < 100 AND ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id<100;

Empty set (0.06 sec)

mysql> explain select \* from employee where id between 1 and 100;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee where id between 1 and 100 |

| dn1 | select \* from employee where id between 1 and 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id between 1 and 100;

Empty set (0.01 sec)

mysql> explain select \* from employee where id in (1,3,5);

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee where id in (1,3,5) |

| dn1 | select \* from employee where id in (1,3,5) |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id in (1,3,5);

Empty set (0.01 sec)

mysql> explain select \* from employee order by id asc;

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

|DATA\_NODE |SQL |

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

| dn1 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT \* FROM employee ORDER BY id ASC) XX WHERE ROWNUM <= 100 |

| dn2 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT \* FROM employee ORDER BY id ASC) XX WHERE ROWNUM <= 100 |

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

2 rows in set (0.01 sec)

mysql> select \* from employee order by id asc;

Empty set (0.01 sec)

mysql> explain select \* from employee where id=(select id from employee where name='a');

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

|DATA\_NODE |SQL |

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

| dn2 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') AND ROWNUM <= 100 |

| dn1 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') AND ROWNUM <= 100 |

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

2 rows in set (0.01 sec)

mysql> select \* from employee where id=(select id from employee where name='a');

Empty set (0.04 sec)

mysql> explain select distinct name from employee;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT name FROM employee GROUP BY name) XX WHERE ROWNUM <= 100 |

| dn2 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT name FROM employee GROUP BY name) XX WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select distinct name from employee;

Empty set (0.05 sec)

mysql> explain select name from employee group by name;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT name FROM employee GROUP BY name) XX WHERE ROWNUM <= 100 |

| dn2 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT name FROM employee GROUP BY name) XX WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select name from employee group by name;

Empty set (0.01 sec)

mysql> explain select name from employee group by name having count(\*)=2;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT name FROM employee GROUP BY name HAVING COUNT(\*) = 2) XX WHERE ROWNUM <= 100 |

| dn2 | SELECT XX.\*, ROWNUM AS RN FROM (SELECT name FROM employee GROUP BY name HAVING COUNT(\*) = 2) XX WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select name from employee group by name having count(\*)=2;

Empty set (0.01 sec)

### （2）MySQL环境测试

mysql> explain select \* from employee where id=1;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee where id=1 |

| dn1 | select \* from employee where id=1 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id=1;

Empty set (0.00 sec)

mysql> explain select \* from employee where name='a';

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

| DATA\_NODE | SQL |

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

| dn2 | SELECT \* FROM employee WHERE name = 'a' LIMIT 100 |

| dn1 | SELECT \* FROM employee WHERE name = 'a' LIMIT 100 |

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

2 rows in set (0.05 sec)

mysql> select \* from employee where name='a';

Empty set (0.00 sec)

mysql> explain select \* from employee where name like '%a%';

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee WHERE name LIKE '%a%' LIMIT 100 |

| dn2 | SELECT \* FROM employee WHERE name LIKE '%a%' LIMIT 100 |

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

2 rows in set (0.01 sec)

mysql> select \* from employee where name like '%a%';

Empty set (0.00 sec)

mysql> explain select \* from employee where name='a' or name='b';

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

|DATA\_NODE | SQL |

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

| dn2 | SELECT \* FROM employee WHERE name = 'a' OR name = 'b' LIMIT 100 |

| dn1 | SELECT \* FROM employee WHERE name = 'a' OR name = 'b' LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where name='a' or name='b';

Empty set (0.00 sec)

mysql> explain select \* from employee where id<100;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee WHERE id < 100 LIMIT 100 |

| dn2 | SELECT \* FROM employee WHERE id < 100 LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id<100;

Empty set (0.00 sec)

mysql> explain select \* from employee where id between 1 and 100;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee where id between 1 and 100 |

| dn1 | select \* from employee where id between 1 and 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id between 1 and 100;

Empty set (0.00 sec)

mysql> explain select \* from employee where id in (1,3,5);

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee where id in (1,3,5) |

| dn1 | select \* from employee where id in (1,3,5) |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id in (1,3,5);

Empty set (0.00 sec)

mysql> explain select \* from employee order by id asc;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT \* FROM employee ORDER BY id ASC LIMIT 100 |

| dn2 | SELECT \* FROM employee ORDER BY id ASC LIMIT 100 |

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

2 rows in set (0.01 sec)

mysql> select \* from employee order by id asc;

Empty set (0.01 sec)

mysql> explain select \* from employee where id=(select id from employee where name='a');

+-----------+------------------------------------------------------------------|DATA\_NODE |SQL |

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

| dn2 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') LIMIT 100 |

| dn1 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id=(select id from employee where name='a');

Empty set (0.00 sec)

mysql> explain select distinct name from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT name FROM employee GROUP BY name LIMIT 100 |

| dn2 | SELECT name FROM employee GROUP BY name LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select distinct name from employee;

Empty set (0.00 sec)

mysql> explain select \* from employee group by name having count(\*) = 2;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT \* FROM employee GROUP BY name HAVING COUNT(\*) = 2 LIMIT 100 | dn2 | SELECT \* FROM employee GROUP BY name HAVING COUNT(\*) = 2 LIMIT 100

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

2 rows in set (0.00 sec)

mysql> select \* from employee group by name having count(\*) = 2;

Empty set (0.00 sec)

## 7、SELECT子查询与表连接查询

测试语句：

explain select \* from employee e, company c where e.id += c.id;

explain select \* from employee e left join company c on e.id = c.id;

explain select \* from employee e, company c where e.id =+ c.id;

explain select \* from employee e right join company c on e.id = c.id;

explain select \* from employee union select \* from employee;

explain select \* from employee union all select \* from employee;

explain select \* from employee where id=(select id from employee where name='a');

测试结果：支持以上表连接和子查询功能，符号“+=”、符号“=+”不支持，可使用left join和right join 代替

测试过程：

### （1）Oracle环境测试

mysql> explain select \* from employee where id=(select id from employee where name='a');

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

|DATA\_NODE |SQL |

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

| dn2 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') AND ROWNUM <= 100 |

| dn1 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') AND ROWNUM <= 100 |

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

2 rows in set (0.01 sec)

mysql> select \* from employee where id=(select id from employee where name='a');

Empty set (0.04 sec)

mysql> explain select \* from employee e, company c where e.id += c.id;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: ERROR. token : EQ, pos : 49

mysql> explain select \* from employee e left join company c on e.id = c.id;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee e left join company c on e.id = c.id |

| dn1 | select \* from employee e left join company c on e.id = c.id |

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

2 rows in set (0.25 sec)

mysql> select \* from employee e left join company c on e.id = c.id;

Empty set (0.31 sec)

mysql> explain select \* from employee e, company c where e.id =+ c.id;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: TODO

mysql> explain select \* from employee e right join company c on e.id = c.id;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee e right join company c on e.id = c.id |

| dn1 | select \* from employee e right join company c on e.id = c.id |

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

2 rows in set (0.00 sec)

mysql> select \* from employee e right join company c on e.id = c.id;

Empty set (0.01 sec)

mysql> explain select \* from employee union all select \* from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | select \* from employee union all select \* from employee |

| dn2 | select \* from employee union all select \* from employee |

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

2 rows in set (0.00 sec)

mysql> select \* from employee union all select \* from employee;

Empty set (0.05 sec)

### （2）MySQL环境测试

mysql> explain select \* from employee where id=(select id from employee where name='a');

+-----------+------------------------------------------------------------------|DATA\_NODE |SQL |

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

| dn2 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') LIMIT 100 |

| dn1 | SELECT \* FROM employee WHERE id = (SELECT id FROM employee WHERE name = 'a') LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select \* from employee where id=(select id from employee where name='a');

Empty set (0.00 sec)

mysql> explain select \* from employee e, company c where e.id += c.id;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: ERROR. token : EQ, pos : 49



mysql> explain select \* from employee e left join company c on e.id = c.id;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee e left join company c on e.id = c.id |

| dn1 | select \* from employee e left join company c on e.id = c.id |

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

2 rows in set (0.00 sec)

mysql> select \* from employee e left join company c on e.id = c.id;

Empty set (0.09 sec)

mysql> explain select \* from employee e, company c where e.id =+ c.id;

ERROR 1064 (HY000): com.alibaba.druid.sql.parser.ParserException: TODO



mysql> explain select \* from employee e right join company c on e.id = c.id;

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

| DATA\_NODE | SQL |

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

| dn2 | select \* from employee e right join company c on e.id = c.id |

| dn1 | select \* from employee e right join company c on e.id = c.id |

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

2 rows in set (0.00 sec)

mysql> select \* from employee e right join company c on e.id = c.id;

Empty set (0.00 sec)

mysql> explain select \* from employee union select \* from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | select \* from employee union select \* from employee |

| dn2 | select \* from employee union select \* from employee |

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

2 rows in set (0.00 sec)

mysql> select \* from employee union select \* from employee;

Empty set (0.00 sec)

mysql> explain select \* from employee union all select \* from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | select \* from employee union all select \* from employee |

| dn2 | select \* from employee union all select \* from employee |

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

2 rows in set (0.00 sec)

mysql> select \* from employee union all select \* from employee;

Empty set (0.00 sec)

mysql> explain select \* from employee union select \* from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | select \* from employee union select \* from employee |

| dn2 | select \* from employee union select \* from employee |

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

2 rows in set (0.01 sec)

mysql> select \* from employee union select \* from employee;

Empty set (0.04 sec)

## 8、FUNCTION 统计函数

测试语句：

explain select avg(id) as avgId from employee;

explain select max(id) as maxId from employee;

explain select min(id) as minId from employee;

explain select count(id) from employee;

explain select sum(id) as sumId from employee;

测试结果：支持以上测试统计函数

测试过程：

### （1）Oracle环境测试

mysql> explain select avg(id) as avgId from employee;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT SUM(id) AS avgIdSUM, COUNT(id) AS avgIdCOUNT FROM employee

WHERE ROWNUM <= 100 |

| dn2 | SELECT SUM(id) AS avgIdSUM, COUNT(id) AS avgIdCOUNT FROM employee

WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select avg(id) as avgId from employee;

Query OK (21.00 sec)

mysql> explain select max(id) as maxId from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT MAX(id) AS maxId FROM employee WHERE ROWNUM <= 100 |

| dn2 | SELECT MAX(id) AS maxId FROM employee WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select max(id) as maxId from employee;

Query OK (21.00 sec)

mysql> explain select min(id) as minId from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT MIN(id) AS minId FROM employee WHERE ROWNUM <= 100 |

| dn2 | SELECT MIN(id) AS minId FROM employee WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select min(id) as minId from employee;

Query OK (21.01 sec)

mysql> explain select count(id) from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT COUNT(id) AS COUNT0 FROM employee WHERE ROWNUM <= 100 |

| dn2 | SELECT COUNT(id) AS COUNT0 FROM employee WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select count(id) from employee;

Query OK (21.02 sec)

mysql> explain select sum(id) as sumId from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT SUM(id) AS sumId FROM employee WHERE ROWNUM <= 100 |

| dn2 | SELECT SUM(id) AS sumId FROM employee WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select sum(id) as sumId from employee;

Query OK (21.00 sec)

### （2）MySQL环境测试

mysql> explain select avg(id) as avgId from employee;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT SUM(id) AS avgIdSUM, COUNT(id) AS avgIdCOUNT FROM employee

LIMIT 100 |

| dn2 | SELECT SUM(id) AS avgIdSUM, COUNT(id) AS avgIdCOUNT FROM employee

LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select avg(id) as avgId from employee;

+-------+

| AVGID |

+-------+

| 0 |

+-------+

1 row in set (0.00 sec)

mysql> explain select max(id) as maxId from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT MAX(id) AS maxId

FROM employee

LIMIT 100 |

| dn2 | SELECT MAX(id) AS maxId

FROM employee

LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select max(id) as maxId from employee;

+-------+

| maxId |

+-------+

| NULL |

+-------+

1 row in set (0.00 sec)

mysql> explain select min(id) as minId from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT MIN(id) AS minId FROM employee LIMIT 100 |

| dn2 | SELECT MIN(id) AS minId FROM employee LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select min(id) as minId from employee;

+-------+

| minId |

+-------+

| NULL |

+-------+

1 row in set (0.00 sec)

mysql> explain select count(id) from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT COUNT(id) AS COUNT0 FROM employee LIMIT 100 |

| dn2 | SELECT COUNT(id) AS COUNT0 FROM employee LIMIT 100 |

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

2 rows in set (0.00 sec)

mysql> select count(id) from employee;

+--------+

| COUNT0 |

+--------+

| 0 |

+--------+

1 row in set (0.05 sec)

mysql> explain select sum(id) as sumId from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT SUM(id) AS sumId FROM employee LIMIT 100 |

| dn2 | SELECT SUM(id) AS sumId FROM employee LIMIT 100 |

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

2 rows in set (0.01 sec)

mysql> select sum(id) as sumId from employee;

+-------+

| sumId |

+-------+

| NULL |

+-------+

1 row in set (0.00 sec)

## 9、FUNCTION 字符串函数

测试语句：

--函数返回字符表达式最左端字符的 ASCII 码值

explain select ASCII(name) as ascii from employee where id = 1;

--函数用于将 ASCII 码转换为字符

explain select name, CHAR(ASCII(name)) from employee e where id = 1;

--函数把字符串全部转换为小写

explain select LOWER(name) from employee e where id = 1;

--函数把字符串全部转换为大写

explain select UPPER(name) from employee e where id = 1;

--函数把数值型数据转换为字符型数据

explain select STR(id) from employee e where id = 1;

--函数把字符串头部的空格去掉

explain select name, LTRIM (name) from employee e where id = 1;

--函数把字符串尾部的空格去掉

explain select name, RTRIM (name) from employee e where id = 1;

--函数返回部分字符串

explain select name, LEFT (name, 2) from employee e where id = 1;

explain select name, RIGHT(name, 2) from employee e where id = 1;

explain select name, SUBSTRING (name, 1, 3) from employee e where id = 1;

explain select name, SUBSTRING (name, 2) from employee e where id = 1;

--函数返回被替换了指定子串的字符串

explain select replace('abc123g', '123', 'def');

--数据类型转换函数

explain select CAST(100+99 as char);

explain select CONVERT('abc' USING utf8);

测试结果：

Oracle不支持CHAR函数，该函数用于将 ASCII 码转换为字符；不支持STR函数，该函数把数值型数据转换为字符型数据；不支持LEFT()函数,Right()函数；不支持SUBSTRING，支持SUBSTR，用法与SUBSTRING相同;不支持REPLACE函数；不支持CONVERT，支持to\_char()函数。

MySQL不支持函数STR()，该函数把数值型数据转换为字符型数据。

测试过程：

### （1）Oracle环境测试

mysql> explain select ASCII(name) as ascii from employee where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select ASCII(name) as ascii from employee where id = 1 |

| dn1 | select ASCII(name) as ascii from employee where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select ASCII(name) as ascii from employee where id = 1;

+-------+

| ASCII |

+-------+

| 122 |

+-------+

1 row in set (0.04 sec)

mysql> select name, CHAR(ASCII(name)) from employee e where id = 1;

ERROR 1105 (HY000): ORA-00936: missing expression

mysql> explain select LOWER(name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select LOWER(name) from employee e where id = 1 |

| dn1 | select LOWER(name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select LOWER(name) from employee e where id = 1;

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

| LOWER(NAME) |

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

| znl |

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

1 row in set (0.03 sec)

mysql> explain select UPPER(name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select UPPER(name) from employee e where id = 1 |

| dn1 | select UPPER(name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select UPPER(name) from employee e where id = 1;

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

| UPPER(NAME) |

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

| ZNL |

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

1 row in set (0.01 sec)

mysql> select STR(id) from employee e where id = 1;

ERROR 1105 (HY000): ORA-00904: "STR": invalid identifier

mysql> explain select name, LTRIM (name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, LTRIM (name) from employee e where id = 1 |

| dn1 | select name, LTRIM (name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, LTRIM (name) from employee e where id = 1;

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

| NAME | LTRIM(NAME) |

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

| znl | znl |

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

1 row in set (0.04 sec)

mysql> explain select name, RTRIM (name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, RTRIM (name) from employee e where id = 1 |

| dn1 | select name, RTRIM (name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, RTRIM (name) from employee e where id = 1;

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

| NAME | RTRIM(NAME) |

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

| znl | znl |

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

1 row in set (0.02 sec)

mysql> select name, LEFT (name, 2) from employee e where id = 1;

ERROR 1105 (HY000): ORA-00904: "LEFT": invalid identifier

mysql> select name, RIGHT(name, 2) from employee e where id = 1;

ERROR 1105 (HY000): ORA-00904: "RIGHT": invalid identifier

mysql> select name, SUBSTRING (name, 1, 3) from employee e where id = 1;

ERROR 1105 (HY000): ORA-00904: "SUBSTRING": invalid identifier

mysql> select name, SUBSTR (name, 1, 3) from employee e where id = 1;

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

| NAME | SUBSTR(NAME,1,3) |

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

| znl | znl |

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

1 row in set (0.01 sec)

mysql> explain select name, SUBSTR (name, 2) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, SUBSTR (name, 2) from employee e where id = 1 |

| dn1 | select name, SUBSTR (name, 2) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, SUBSTR (name, 2) from employee e where id = 1;

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

| NAME | SUBSTR(NAME,2) |

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

| znl | nl |

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

1 row in set (0.01 sec)

mysql> explain select replace('abc123g', '123', 'def');

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

| DATA\_NODE | SQL |

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

| dn2 | select replace('abc123g', '123', 'def') |

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

1 row in set (0.00 sec)

mysql> select replace('abc123g', '123', 'def');

ERROR 923 (HY000): ORA-00923: FROM keyword not found where expected

mysql> explain select CAST(100+99 as int) from employee;

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

| DATA\_NODE | SQL |

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

| dn1 | SELECT CAST(100 + 99 AS int) FROM employee WHERE ROWNUM <= 100 |

| dn2 | SELECT CAST(100 + 99 AS int) FROM employee WHERE ROWNUM <= 100 |

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

2 rows in set (0.00 sec)

mysql> select CAST(100+99 as int) from employee;

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

| CAST(100+99ASINT) |

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

| 199 |

| 199 |

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

2 rows in set (0.01 sec)

mysql> explain select CONVERT('abc' USING utf8) from employee;

ERROR 1064 (HY000): syntax error, expect RPAREN, actual USING USING

mysql> select CONVERT('abc' USING utf8) from employee;

ERROR 1064 (HY000): syntax error, expect RPAREN, actual USING USING

mysql> explain select to\_char(1234,'99999999') from company;

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

|DATA\_NODE| SQL |

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

| dn2 | SELECT to\_char(1234, '99999999') FROM company WHERE ROWNUM <= 100

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

1 row in set (0.00 sec)

mysql> select to\_char(1234,'99999999') from company;

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

| TO\_CHAR(1234,'99999999') |

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

| 1234 |

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

1 row in set (0.06 sec)

mysql> explain select to\_char(1234,'99999999') from employee;

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

|DATA\_NODE|SQL |

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

| dn1 | SELECT to\_char(1234, '99999999') FROM employee WHERE ROWNUM <= 100

| dn2 | SELECT to\_char(1234, '99999999') FROM employeeWHERE ROWNUM <= 100

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

2 rows in set (0.00 sec)

mysql> select to\_char(1234,'99999999') from employee;

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

| TO\_CHAR(1234,'99999999') |

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

| 1234 |

| 1234 |

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

2 rows in set (0.01 sec)

### （2）MySQL环境测试

mysql> explain select ASCII(name) as ascii from employee where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select ASCII(name) as ascii from employee where id = 1 |

| dn1 | select ASCII(name) as ascii from employee where id = 1 |

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

2 rows in set (0.01 sec)

mysql> select ASCII(name) as ascii from employee where id = 1;

+-------+

| ascii |

+-------+

| 108 |

+-------+

1 row in set (0.01 sec)

mysql> explain select name, CHAR(ASCII(name)) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, CHAR(ASCII(name)) from employee e where id = 1 |

| dn1 | select name, CHAR(ASCII(name)) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, CHAR(ASCII(name)) from employee e where id = 1;

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

| name | CHAR(ASCII(name)) |

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

| leader us | l |

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

1 row in set (0.01 sec)

mysql> explain select LOWER(name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select LOWER(name) from employee e where id = 1 |

| dn1 | select LOWER(name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select LOWER(name) from employee e where id = 1;

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

| LOWER(name) |

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

| leader us |

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

1 row in set (0.00 sec)

mysql> explain select UPPER(name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select UPPER(name) from employee e where id = 1 |

| dn1 | select UPPER(name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select UPPER(name) from employee e where id = 1;

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

| UPPER(name) |

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

| LEADER US |

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

1 row in set (0.00 sec)

mysql> explain select STR(id) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select STR(id) from employee e where id = 1 |

| dn1 | select STR(id) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select STR(id) from employee e where id = 1;

ERROR 1105 (HY000): FUNCTION db1.STR does not exist

mysql> explain select name, LTRIM (name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, LTRIM (name) from employee e where id = 1 |

| dn1 | select name, LTRIM (name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, LTRIM (name) from employee e where id = 1;

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

| name | LTRIM (name) |

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

| leader us | leader us |

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

1 row in set (0.00 sec)

mysql> explain select name, RTRIM (name) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, RTRIM (name) from employee e where id = 1 |

| dn1 | select name, RTRIM (name) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, RTRIM (name) from employee e where id = 1;

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

| name | RTRIM (name) |

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

| leader us | leader us |

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

1 row in set (0.00 sec)

mysql> explain select name, LEFT (name, 2) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, LEFT (name, 2) from employee e where id = 1 |

| dn1 | select name, LEFT (name, 2) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, LEFT (name, 2) from employee e where id = 1;

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

| name | LEFT (name, 2) |

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

| leader us | le |

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

1 row in set (0.00 sec)

mysql> explain select name, RIGHT(name, 2) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, RIGHT(name, 2) from employee e where id = 1 |

| dn1 | select name, RIGHT(name, 2) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, RIGHT(name, 2) from employee e where id = 1;

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

| name | RIGHT(name, 2) |

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

| leader us | us |

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

1 row in set (0.00 sec)

mysql> explain select name, SUBSTRING (name, 1, 3) from employee e where id = 1;

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

|DATA\_NODE | SQL |

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

| dn2 | select name, SUBSTRING (name, 1, 3) from employee e where id = 1 |

| dn1 | select name, SUBSTRING (name, 1, 3) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, SUBSTRING (name, 1, 3) from employee e where id = 1;

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

| name | SUBSTRING (name, 1, 3) |

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

| leader us | lea |

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

1 row in set (0.00 sec)

mysql> explain select name, SUBSTRING (name, 2) from employee e where id = 1;

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

| DATA\_NODE | SQL |

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

| dn2 | select name, SUBSTRING (name, 2) from employee e where id = 1 |

| dn1 | select name, SUBSTRING (name, 2) from employee e where id = 1 |

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

2 rows in set (0.00 sec)

mysql> select name, SUBSTRING (name, 2) from employee e where id = 1;

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

| name | SUBSTRING (name, 2) |

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

| leader us | eader us |

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

1 row in set (0.00 sec)

mysql> explain select replace('abc123g', '123', 'def');

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

| DATA\_NODE | SQL |

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

| dn2 | select replace('abc123g', '123', 'def') |

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

1 row in set (0.00 sec)

mysql> select replace('abc123g', '123', 'def');

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

| replace('abc123g', '123', 'def') |

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

| abcdefg |

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

1 row in set (0.00 sec)

mysql> explain select CAST(100+99 as char);

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

| DATA\_NODE | SQL |

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

| dn2 | select CAST(100+99 as char) |

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

1 row in set (0.00 sec)

mysql> select CAST(100+99 as char);

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

| CAST(100+99 as char) |

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

| 199 |

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

1 row in set (0.00 sec)

mysql> explain select CONVERT('abc' USING utf8);

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

| DATA\_NODE | SQL |

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

| dn2 | select CONVERT('abc' USING utf8) |

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

1 row in set (0.00 sec)

mysql> select CONVERT('abc' USING utf8);

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

| CONVERT('abc' USING utf8) |

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

| abc |

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

1 row in set (0.00 sec)

## 10、FUNCTION 日期函数

测试语句：

--函数返回YYYY-MM-DD中的日期值

select day('1998-02-03');（MySQL写法）

select to\_char(sysdate,'dd') as nowDay from dual;（Oracle写法）

--函数返回YYYY-MM-DD中的月份值

select MONTH('1998-02-03');（MySQL写法）

select to\_char(sysdate,'mm') as nowMonth from dual;（Oracle写法）

--函数返回YYYY-MM-DD中的年份值

select YEAR('1998-02-03');（MySQL写法）

select to\_char(sysdate,'yyyy') as nowYear from dual;（Oracle写法）

--函数返回指定日期 date 加上指定的额外日期间隔 number 产生的新日期

select date\_add(now(), interval 2 day);

--函数返回两个指定日期的差值

select datediff(now(), '2015-04-19'), datediff(now(), '2015-04-20');

--函数以字符串的形式返回日期的指定部分

select DATENAME(day, now()) as 'YEAR'

--函数以 DATETIME 的缺省格式返回系统当前的日期和时间

select GETDATE();

select to\_date(sysdate,'yyyy-mm-dd hh24:mi:ss') from dual

测试结果：Mycat对函数部分支持（包括字符串函数），分别与Oracle和MySQL的SQL语法规则相同。

测试过程：

### （1）Oracle环境测试

mysql> select to\_char(sysdate,'dd') as nowDay from dual;

+--------+

| NOWDAY |

+--------+

| 22 |

+--------+

1 row in set (0.01 sec)

mysql> select to\_char(sysdate,'mm') as nowMonth from dual;

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

| NOWMONTH |

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

| 04 |

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

1 row in set (0.01 sec)

mysql> select to\_char(sysdate,'yyyy') as nowYear from dual;

+---------+

| NOWYEAR |

+---------+

| 2015 |

+---------+

1 row in set (0.01 sec)

mysql> select to\_date(sysdate,'yyyy-mm-dd hh24:mi:ss') from dual;

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

| TO\_DATE(SYSDATE,'YYYY-MM-DDHH24:MI:SS') |

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

| 2015-04-22 15:42:31.0 |

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

1 row in set (0.00 sec)

### （2）MySQL环境测试

mysql> select day('1998-02-03');

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

| day('1998-02-03') |

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

| 3 |

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

1 row in set (0.00 sec)

mysql> select MONTH('1998-02-03');

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

| MONTH('1998-02-03') |

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

| 2 |

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

1 row in set (0.00 sec)

mysql> select YEAR('1998-02-03');

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

| YEAR('1998-02-03') |

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

| 1998 |

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

1 row in set (0.00 sec)

mysql> select date\_add(now(), interval 2 day);

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

| date\_add(now(), interval 2 day) |

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

| 2015-04-22 14:49:19 |

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

1 row in set (0.00 sec)

mysql> select datediff(now(), '2015-04-19'), datediff(now(), '2015-04-20');

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

| datediff(now(), '2015-04-19') | datediff(now(), '2015-04-20') |

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

| 1 | 0 |

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

1 row in set (0.00 sec)

mysql> select DATENAME(day, now()) as 'YEAR';

ERROR 1305 (42000): FUNCTION db2.DATENAME does not exist

mysql> select GETDATE();

ERROR 1305 (42000): FUNCTION db1.GETDATE does not exist

mysql> select NOW();

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

| NOW() |

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

| 2015-04-21 13:47:58 |

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

1 row in set (0.00 sec)

# 三、SQL92测试总结

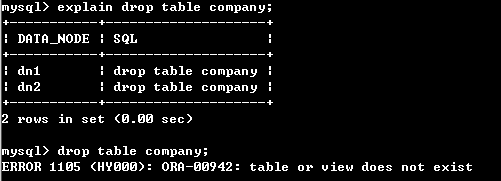
## 1、测试问题汇总

### （1）DROP TABLE 从数据库中删除表

Oracle和MySQL都支持该功能，有一个需要注意的地方：

如果只有一个节点dn1有表company，另一个节点dn2没有（可能意外删除），执行该语句报错，但实际已经删除dn1上的company表。

java.sql.SQLException: ORA-00942: table or view does not exist





### （2）ALTER TABLE 修改数据库表结构

**1）添加索引功能，Oracle只支持B写法，MySQL两种写法都支持**

A、explain alter table employee add index IDX\_EMP\_NAME (name);

B、explain create index IDX\_EMP\_NAME on employee(name);

**2）删除索引功能，Oracle只支持B写法，MySQL只支持A写法**

A、explain alter table employee drop index IDX\_EMP\_NAME;

B、explain drop index IDX\_EMP\_NAME;

**3）添加唯一约束功能，Oracle 两种写法都不支持，MySQL只支持A写法**

A、explain alter table employee add unique EMP\_NAME(name);（MySQL写法）

B、explain alter table employee add constraint EMP\_UNIQUE unique(name);（Oracle写法）

**4）修改列名功能，Oracle 两种写法都不支持，MySQL只支持A写法**

A、explain alter table employee change name newname varchar(100);（MySQL写法）

B、explain alter table employee rename column name to newname;（Oracle写法）

**5）删除字段功能，Oracle不支持，MySQL支持A写法**

A、explain alter table employee drop age;

**6）添加非空约束，Oracle和MySQL都只支持B写法，都不支持constraint关键字**

A、explain alter table employee modify name constraint NAME\_NOT\_NULL not null; （Oracle写法）

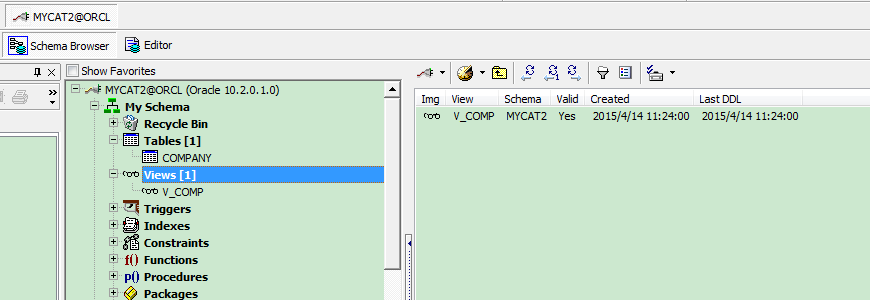
B、explain alter table employee modify name varchar(2) not null;

### （3）CREATE VIEW 创建一个视图

Oracle和MySQL都支持该功能，但须在schema使用table标签添加该视图

在schema.xml中添加标签

<table name="v\_comp" primaryKey="ID" type="global" dataNode="dn1,dn2,dn3" />



### （4）DROP VIEW 从数据库中删除视图

Oracle和MySQL都支持该功能，但须在schema使用table标签添加该视图

在schema.xml中添加标签

<table name="v\_comp" primaryKey="ID" type="global" dataNode="dn1,dn2,dn3" />

### （5）DROP INDEX 从数据库中删除索引（MySQL也不支持）

Oracle和MySQL都存在异常，无法在全部节点同时执行删除，执行计划不确定。第一次执行命令在一个节点执行了删除索引，再次执行命令在另一个节点执行。Oracle和MySQL都存在该问题。

### （6）CREATE PROCEDURE 创建一个存储过程（MySQL也不支持）

Oracle和MySQL都不支持创建存储过程。

### （7）DELETE 从数据库表中删除数据行

Oracle环境不支持该功能，MySQL环境支持该功能

### （8）SELECT子查询与表连接查询

Oracle和MySQL都支持子查询和表连接查询，但符号“+=”、符号“=+”不支持，可使用left join和right join 代替

### （9）FUNCTION 字符串函数

Oracle不支持CHAR函数，该函数用于将 ASCII 码转换为字符；不支持STR函数，该函数把数值型数据转换为字符型数据；不支持LEFT()函数,Right()函数；不支持SUBSTRING，支持SUBSTR，用法与SUBSTRING相同;不支持REPLACE函数；不支持CONVERT，支持to\_char()函数。

MySQL不支持函数STR()，该函数把数值型数据转换为字符型数据。

### （10）FUNCTION 日期函数

Mycat对函数部分支持（包括字符串函数），分别与Oracle和MySQL的SQL语法规则相同。

## 2、测试结论

在本次测试中MySQL环境只有删除索引和创建存储过程问题。Oracle环境比较严重的问题是不支持DELETE命令，另外对ALTER命令支持较差。Mycat对函数部分支持，分别与Oracle和MySQL的SQL语法规则相同。

有任何问题请与我联系，欢迎与大家交流

王金剑 微信：life3721com QQ：1526330277 邮件：acsebt@qq.com