Decimal使用手册

修订历史

版本	日期	修改描述	作者	备 注
Cedar 0.3	2017-9-29	DECIMAL 使用手册	徐石磊	无

1 Decimal的使用精度。

Decimal类型的格式为decimal(p,s)，其中p表示多少位有效数字，s表示小数位个数。
（1）设计的decimal精度为（38,37）。即，最多可以有38个有效数字（p<=38），37位小数(s<=37)，并且要求p>s。

> 建表语句
> 例：
> CREATE TABLE 表名(列名1 类型名 PRIMARY KEY,…,列名2 decimal(p,s));
> create table t1(c1 int, c2 decimal(38,37), primary key(c1));

2 Decimal的类型转换

四则运算时的类型转换：
（1）float/double {+、-、*、/} decimal - > double;
（2）int/ DateTime/ PreciseDateTime/ Varchar/ CreateTime/ ModifyTime/ Bool/ {+、-、*、/} decimal - > decimal
说明：除了float和double与decimal进行四则运算时转换成double类型之外，其他目前支持的数据类型与decimal一起运算时都转换为decimal。

3 Decimal的四则运算精度

其中，表格中各参数含义为：
p：第一个数的精度 s：第一个数的小数位数
p’：第二个数的精度 s’：第二个数的小数位数
int_len：运算结果整数有效位
Cedar 中Decimal的p的最大精度为38，s的最大精度是37。运算结果要优先满足p-s<shema_p-schema_s。所以最终运算结果根据整型范围动态调整小数有效位数。

运算	Precision	scale
ADD	Min(38,max(p-s,p’-s’)+max(s,s’)+1)	Max(s,s’)
SUB	Min(38,max(p-s,p’-s’)+max(s,s’)+1)	Max(s,s’)
MUL	Min(37,s+s’)+ int_len	Min(37,s+s’)
DIV	38	38-int_len

注：
（1）在计算时，如果整数部分有效数字大于最高有效数字38，则输出结果为最大值，即38个9。
（2）在一个多个运算符的运算中，当前面的运算结果整数有效位已经超过38时，再进行运算时就不符合我们运算精度表。在下面溢出处理中已经说明，对于这种在select中进行的计算，并不会产生数据入库行为，因此这类运算失真不影响数据库正确。

示例：

```
mysql> select * from t1;
+---+-----------------+-----------------------------------------------------------------------------------------------+
| k | c2              | c3                                                                                            |
+---+-----------------+-----------------------------------------------------------------------------------------------+
| 1 | 9999999999.99999 | 99999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999 |
+---+-----------------+-----------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql> select c2+c3 from t1;
+-----------------------------------------------------------------------------------+
| c2+c3                                                                             |
+-----------------------------------------------------------------------------------+
| 1000000000000000000000000000000000000000000000000000000009999999998.99999 |
+-----------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql> select c2+c3+1 from t1;
+-----------------------------------------------------------------------------------+
| c2+c3+1                                                                           |
+-----------------------------------------------------------------------------------+
| 1000000000000000000000000000000000000000000000000000000009999999999.99999 |
+-----------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

（3）对于MOD取余运算，目前Decimal的设计，是先截取整数部分，然后对整数部分进行取余操作。

4 Decimal的功能支持

4.1 支持insert、replace、update、delete、select等操作

示例：

```
create table t1(c1 int, c2 decimal(30,5), primary key(c1));
insert into t1 values (1, 100.001);
insert into t1 values (2, 200.001);
replace into t1 values (2, 300.001);
delete from t1 where c2 = 100.001;
update t1 set c2 = c2 + 100.001 where c1 = 2;
select * from t1;
```

4.2 支持聚合函数avg( )、count( )、sum( )、max( )、min( )等操作

示例：

```
create table t1(c1 int, c2 decimal(30,5), primary key(c1));
insert into t1 values (1, 100.001);
insert into t1 values (2, 100.001);
insert into t1 values (3, 100.001);
select count(c2) from t1;
select max(c2) from t1;
select min(c2) from t1;
select sum(c2) from t1;
select avg(c2) from t1;
```

4.3 支持系统函数cast( )、coalesce( )

示例：

```
create table t2(c1 int, c2 decimal(10,3), primary key(c1));
insert into t2 values (1, 1000000.4444);
select cast(c2 as decimal(12,1)) from t2;
select cast(c2 as decimal(12,5)) from t2;
select cast(c2 as decimal(12,6)) from t2;
select cast(coalesce(null,c2) as decimal(12,5)) from t1;
insert into t1 (c1) values (2);
select coalesce(c2, 0.11) from t1;
```

注：当违反了整数部分（p-s）精度 > 定义的整数部分（p-s）精度，会报错。

4.4 支持逻辑运算符（=、>=、>、<=、<、!=(<>)）

示例：

```
create table t1(c1 int, c2 decimal(4,2), primary key(c1));
insert into t1 values (1, 23.12);
select c2 > 23.12 from t1;
select c2 >= 23.120 from t1;
select c2 < 23.13 from t1;
select c2 <= 23.12 from t1;
select c2 = 23.120 from t1;
select c2 <> 23.12 from t1;
select c2 != 23.12 from t1;
```

4.5 支持运算符（between and、in、is null、is not null、not、and、or）

示例：

```
create table t1(c1 int, c2 decimal(4,2), primary key(c1));
insert into t1 values (1, 23.12);
select * from t1 where c2 between 23.11 and 23.13;
select * from t1 where c2 in (23.12, -22);
insert into t1 (c1) values (2);
select * from t1 where c2 is null;
select * from t1 where c2 is not null;
select * from t1 where not c2;
select * from t1 where c2 and true;
select * from t1 where c2 and false;
select * from t1 where c2 or true;
```

注：between and两边都包含

4.6 Decimal类型数据支持集合操作（union、except、intersect）

示例：

```
select c2 from t1 union select c2 from t2;
select c2 from t1 except select c2 from t2;
select c2 from t1 intersect select c2 from t2;
```

4.7 Decimal类型数据可以作为连接属性的支持情况

Decimal支持Semi join，Hash join，BloomFilterJoin。

示例：
select * from t1 inner join t2 on t1.c2 = t2.c2;
select * from t1,t2 where t1.c2 = t2.c2;
select /+*JOIN(si)*/ * from t1 inner join t2 on t1.c2 = t2.c2;
select /+*JOIN(si)*/ * from t1,t2 where t1.c2 = t2.c2;
select /+ *join(bloomfilter_join)* /* from t1 inner join t2 on t1.c2 = t2.c2;
注：目前系统中对于BloomFilterJoin尚且只支持相同数据类型(即两个都是decimal数据类型)的Join。

4.8 Decimal类型数据可以作为数据表和二级索引的主键

示例：
drop table t1;
create table t1 (c1 decimal(5,2) primary key, c2 int);
insert into t1 values (345.23, 4);
insert into t1 values (546.5, 4);
insert into t1 values (22.09, 4);
insert into t1 values (921.56, 4);
delete from t1 where c1 = 22.09;
alter system set index_immediate_effect=1 server_type=rootserver;
drop table t1;
create table t1 (c1 int primary key, c2 decimal(5,2));
create index i1 on t1(c2);
insert into t1 values (345, 412.2);
insert into t1 values (45, 12.24);
insert into t1 values (111, 412.2);
insert into t1 values (37, 12.33);
delete from t1 where c1 = 45;
select * from t1 where c2 = 412.2;

4.9 Decimal类型支持特殊格式的输入

示例：
create table t1(c1 int, c2 decimal(2,1), primary key(c1));
insert into t1 values (1, 3.5);
insert into t1 values (2, 3.);
insert into t1 values (3, .5);
select * from t1;

5 Decimal的溢出机制

5.1 Decimal 插入、更新溢出机制

当插入含有小数的有效数字超过38位，如果存在小数部分，则会对小数部分进行截取，将整个数据截取成38位。如果是整数部分已经超过38位，则系统会报错

> 示例：
> create table t1(c1 int, c2 decimal(38,37), primary key(c1));
> insert into t1 values (1, 1.00000000000000000000000000000000000011);

当输入的小数点前的有效数字大于建表时定义的 p - s 时，会报溢出错误。

> 示例：
> create table t1(c1 int, c2 decimal(10,3), primary key(c1));
> insert into t1 values (1, 10000000);

当输入的小数点后有效数字超过 s 时，直接截取处理

> 示例：
> create table t1(c1 int, c2 decimal(10,3), primary key(c1));
> insert into t1 values (1, 0.1111);

当使用update语句时，如果修改列（修改值）的整数部分（p-s）大于被修改列定义的整数部分（p-s），会报overflow错误。

> 示例：
> create table t1(c1 int, c2 decimal(3,1), c3 double, primary key(c1));
> insert into t1 values (1, 34.2, 231.444);
> insert into t1 values (2, 34.2, 21.444);
> update t1 set c2 = c3 where c1 = 1;

5.2 Decimal类型读取机制

当对Decimal数据类型进行读取操作时，无论查询进行的计算有效数字为多少都不会报溢出错误。

示例

```
mysql> create table t1(k int primary key, c2 decimal(15,5), c3 decimal(38,0));
Query OK, 0 rows affected (0.86 sec)

mysql> insert into t1 values (1, 9999999999.99999, 99999999999999999999999999999999999991.99);
Query OK, 1 row affected (0.00 sec)

mysql> select c2* c3 from t1;
+--------------------------------------+
| c2* c3                               |
+--------------------------------------+
| 99999999999999999999999999999999999999 |
+--------------------------------------+
1 row in set (0.00 sec)

mysql>
```

5.3 Decimal的数据Insert限制

decimal支持插入的最大int型整数为9223372036854775807。这是受限于一个int64的数值范围限制。如果需要插入99999999999999999999，可以改写为插入99999999999999999999.0。

示例

```
mysql> create table t1(c1 int, c2 decimal(38,8), primary key(c1));
Query OK, 2 rows affected (0.67 sec)

mysql> insert into t1 values (1, 99999999999999999999);
Query OK, 1 row affected (0.01 sec)

mysql> select * from t1;
+------+--------------------------------+
| c1   | c2                             |
+------+--------------------------------+
|    1 | 9223372036854775807.00000000 |
+------+--------------------------------+
1 row in set (0.00 sec)

mysql> create table t1(c1 int, c2 decimal(38,0), primary key(c1));
Query OK, 2 rows affected (0.81 sec)

mysql> insert into t1 values (1, 99999999999999999999999999999999999999.0);
Query OK, 1 row affected (0.00 sec)

mysql> select * from t1;
+------+--------------------------------------+
| c1   | c2                                   |
+------+--------------------------------------+
|    1 | 99999999999999999999999999999999999999 |
+------+--------------------------------------+
1 row in set (0.00 sec)
```