主键自增功能开发文档

修订历史

版本	修订日期	修订描述	作者	备注
Cedar 0.3	2016-12-04	主键自增功能开发文档	刘柏众	创建
Cedar 0.3	2017-09-29	主键自增功能开发文档	黄建伟	修订

1 总体设计

1.1 综述

Cedar是华东师范大学数据科学与工程学院基于OceanBase 0.4.2 研发的可扩展的关系数据库，实现了巨大数据量上的跨行跨表事务。在开源的OceanBase 0.4 版本中虽然存在主键自增功能的语法支持，但是实际功能却没有实现。为了满足某些应用需求，需要在Cedar中实现主键自增功能。

实现之后，当我们定义表格时如果给主键列添加 AUTO_INCREMENT 关键词标记，表示此主键列被定义为自增列，即在插入数据（REPLACE 或 INSERT）时没有给定相应主键值的情况下，会自动生成特定主键列的值（默认从1开始自增）。

1.2 名词解释

主控服务器（RootServer, RS）：Cedar集群的主控节点，负责管理集群中的所有服务器，以及维护tablet信息。
更新服务器（UpdateServer, UPS）：负责存储Cedar系统的增量数据，并提供事务支持。
基准数据服务器（ChunkServer, CS）：负责存储Cedar系统的基线数据。
合并服务器（MergeServer, MS）：负责接收并解析客户端的SQL请求，经过词法分析、语法分析、查询优化等一系列操作后发送到CS和UPS，合并基线数据和增量数据。

1.3 功能

1.3.1基本流程及系统表

新增系统表 __all_auto_increment 记录每张表自增的主键当前最大值，每次插入新数据行时，互斥地读取出该值，并自增1做为新数据行的主键值，同时更新系统表中的记录。

我们假设用户表 user(table_id：3001)的主键列 user_id(column_id: 16)是自增列，则用户表和自增系统表的内容如下所示：

- **用户表 user** ：

user_id	user_name
1	Jack
2	Tom

- **系统表 __all_auto_increment** ：

table_id	column_id	max_value
3001	16	2

1.3.2语法支持

1.3.2.1 CREATE TABLE

定义表格时给主键列添加 AUTO_INCREMENT 关键词标记。

```
CREATE TABLE table_name (id INT PRIMARY KEY AUTO_INCREMENT, col int);
```

AUTO_INCREMENT使用限制：

1. 只能标记主键列。
2. 最多只能标记一个列。
3. 只能标记数值类型 (bigint, int, integer, mediumint, smallint, tinyint, float, double, real)列。

1.3.2.2 DROP TABLE

删除表时将自动删除系统表 __all_auto_increment 中记录最大自增值的表记录。

1.3.2.3 REPLACE

我们默认使用前面提到的user表的schema。

- **情形一** ：使用自增的主键值

```
REPLACE INTO user(user_name) VALUES(Huangjw);
```

- **情形二** ：指定主键值

```
REPLACE INTO user(user_id, user_name) VALUES(3, Huangjw);
```

指定的主键值若大于最大主键，则会更新系统表，下次插入会从新的主键值开始自增。

1.2.2.4 INSERT

若使用自增的主键值，则INSERT实际相当于执行 REPLACE，不检查主键重复，由 AUTO_INCREMENT 功能保证主键的惟一性。若指定主键值，则会之前INSERT流程，检查主键重复。

1.4 性能指标

若一个表的某一个主键值被标记为自增列时，相较于无自增列的插入数据，性能会下降。这是因为我们利用系统表去记录相应最大自增值，在高并发的情况下，系统会竞争这一公共资源，我们需要对此加锁，导致性能降低。

2 模块设计

2.1 系统表设计

2.1.1 在表 __all_all_column中新增 auto_increment 列

__all_all_column表中记录了所有表所有列的元数据，增加auto_increment属性标记该列是否需要自增。
对应函数：

```
 int ObExtraTablesSchema::all_all_column_schema(TableSchema& table_schem
a);
```

2.1.2 新增表 __all_auto_increment

新增系统表 __all_auto_increment 记录所有自增主键的当前最大值。
表结构：

field	type	key
table_id	int	1
column_id	int	2
max_value	int	0

对应函数：

```
 int ObExtraTablesSchema::all_auto_increment_schema(TableSchema& table_sc
hema);
```

没有直接在表 __all_all_column 记录最大主键值的原因是该表以 column_name为主键，UPS在执行 REPLACE 操作时不知道表的名字，不便于直接更新该表。

2.2 CREATE TABLE 语句

2.2.1 AUTO_INCREMENT 语法支持

在开源的OceanBase 0.4 版本中支持在 CREATE TABLE 中添加 AUTO_INCREMENT 关键词, 在构建物理计划时可以直接使用到这个属性，已存在对列类型的检查。
对应函数：

```
 int ObTransformer::gen_physical_create_table(ObLogicalPlan logical_plan,
 ObPhysicalPlan,            *physical_plan, ErrStat& err_stat, const uint
64_t& query_id,int32_t index);
```

2.2.2 增加 AUTO_INCREMENT 限制

在构建CREATE TABLE的物理计划时，检查AUTO_INCREMENT是否定义正确，即只支持最多一个主键列自增。

2.2.3 给表 __all_all_column中 auto_increment 属性赋值

对应函数：

```
int ObExtraTablesSchema::all_all_column_schema(TableSchema& table_schem
a);
```

2.2.4 在表 __all_auto_increment 插入新记录

新建表格时，在__all_auto_increment 中插入一行新记录，设置最大主键值（max_value）从OB_DEFAULT_AUTO_INCREMENT_VALUE （默认为0）+ 1 开始自增。

2.3 DROP TABLE 语句

2.3.1 删除系统表__all_auto_increment 记录

先从 **all_all_column系统表中查询出待删除表中自增的主键列ID(column_id)，再删除**all_auto_increment中对应表记录。
对应函数：

```
int ObSchemaServiceImpl::delete_auto_increment(const int64_t table_id);
```

2.4 REPLACE 语句

2.4.1 MS 取消对自增主键必须定义值的限制

原REPLACE在执行时会检查是否设置了主键值，修改后自增类型的主键值可以不设置。
对应函数：

```
int ObTransformer::gen_physical_replace_new(ObLogicalPlan logical_plan, O
bPhysicalPlan *physical_plan, ErrStat& err_stat, const uint64_t& query_i
d, int32_t index);
```

2.4.2 UPS 查询 max_value

MS发送给UPS不含自增主键值的物理计划，UPS查询增量数据，获取max_value。
对应函数：

```
template <class T>
int MemTableModifyTmpl<T>::get_auto_increment_value(const int64_t table_i
d, const uint64_t column_id, int64_t &auto_value);
```

2.4.3 MS 加载 max_value 到 UPS

在执行每日合并后，系统表**all_auto_increment 的增量数据合并到CS中，此时 UPS 查询不到 max_value 值，返回OB_ERR_AUTO_VALUE_NOT_SERVE 的错误，MS识别到该错误，执行一条更新** all_auto_increment 表的SQL语句，将 max_value 值加载到UPS增量数据中。
对应函数：

```
int ObMySQLServer::load_auto_value_and_retry(const common::ObString& q, O
bMySQLResultSet &result, ObSqlContext &context, ObMySQLCommandPacket*& pa
cket, int &err)
```

我们设置最大重试数次为10。

2.4.4 UPS 构造新主键值

在迭代器ObCellIterAdaptor中定义新数据结构AutoIncrementInfo用于保存主键自增相关的数据，若未定义主键值，则需要在原行基本上插入一个主键列。
对应函数：

```
int ObCellIterAdaptor::add_auto_increment_column(const ObRow *&row);
```

2.4.5 UPS 构造新主键值

UPS在插入新记录后，需要更新__all_auto_increment中记录的max_value值。
对应函数：

```
template <class T>
int MemTableModifyTmpl<T>::update_auto_increment_value(const ObCellIterAd
aptor &cia,
const uint64_t table_id, const uint64_t auto_column_id, const int64_t old
_value);
```

新增新类型的迭代器 ObAutoIncrementCellIterAdaptor 用于构造系统表__all_auto_increment中的行。

2.5 INSERT 语句

在构建 INSERT 语句的物理计划时，判断当前表格的主键是否是自增类型，若需要自增，则构建 REPLACE 语句的物理计划，即此时 INSERT 操作使用 REPLACE 语义。
对应函数：

```
int ObTransformer::generate_physical_plan(ObLogicalPlan logical_plan, ObP
hysicalPlan& physical_plan, ErrStat& err_stat, const uint64_t& query_id,
int32_t* index);
```

3 使用限制条件和注意事项

- 暂不支持插入NULL或0时自增。
- 不支持设置自增开始值和步长。
- 未对数值越界做处理（float, double等非int类型主键值越界未定义，应使用通用的特殊的数据结构表示max_value）。