# update语句的解析与执行

# update语句的解析

sql\_stmt\_update \*parse\_sql\_stmt\_update(ParserT \*parser) 函数用于update语句的解析。

该函数接收一个解析器，返回一个sql\_stmt\_update结构体，这个结构体中包含三个字段tableName,fields，fieldsExpr,where，其中tableName表示操作的表名，fields表示要修改的字段，fieldsExpr表示要修改的字段的值或者表达式，where表示update中where子句的选择关系。

update <tableName> set <field1>=<expr1>[,<field2>=<expr2>,...] where <logicla\_expr>

1. 首先使用解析器获得Token，使用matchToken()函数进行判断第一个词是否是TOKEN\_REVERSE\_WORD 类型的update,如果不是，返回空；否则继续进行
2. 再次使用解析器获得Token，判断是否是TOKEN\_WORD类型，获得表名。否则就提示缺少表名，返回空
3. 因为这里的update语句比较简单，没有使用from，所以不必考虑别名，直接使用TableReference\_make函数传入表名创建一个空的别名变量，在使用SRATable()函数创建一个SRA\_T类型表示这个表
4. 使用解析器获得Token，再次使用matchToken函数解析set词
5. 接下来使用循环结构，分别解析字段名，等于号，字段值（或表达式），分别放入两个arraylist中，保存起来，留意解析逗号。循环的结束条件是token->type!=TOKEN\_WORD，因为一旦解析到where，就证明没有要修改的字段了，所以可以结束字段的解析了
6. 使用matchToken函数解析where关键词
7. 解析where后面的子句，直接使用parseExpressionRD解析出where后面的表达式，保存下来
8. 如果有where子句，就利用SRASelect函数传入前面的SRATable构造一个select的SRA\_T类型。否则保留之前的SRATable
9. 给sql\_stmt\_update结构体的各字段赋值，返回sql\_stmt\_update结构体

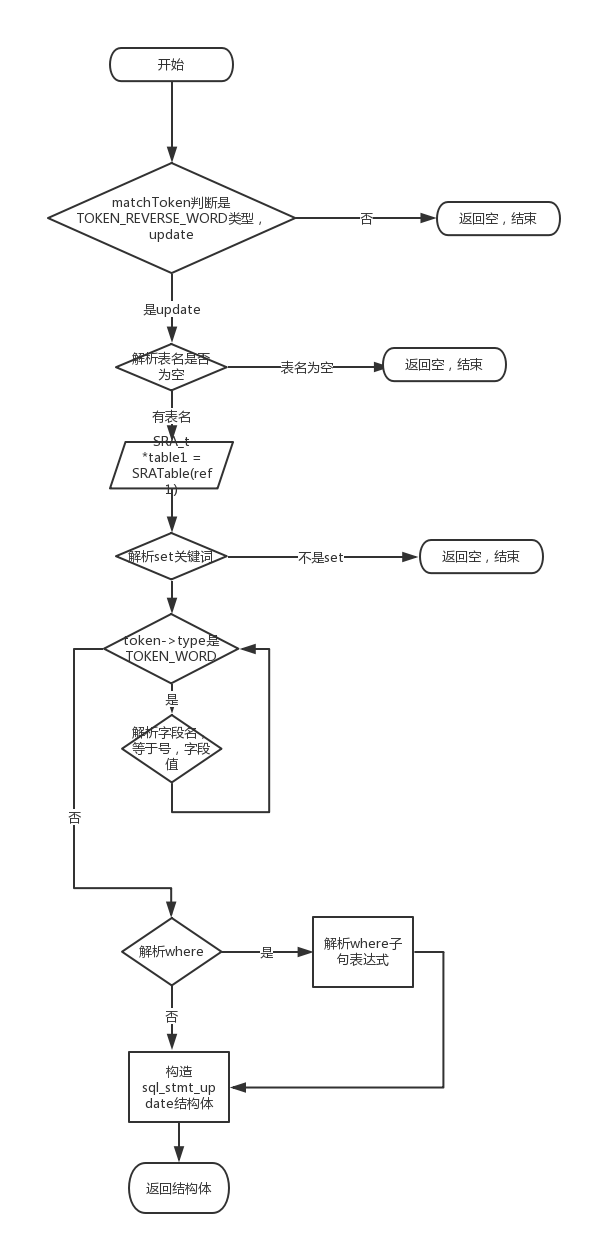
# update语句的执行

update语句的物理执行：首先使用physical\_scan\_generate函数，这个函数会根据传入的sqlStmtUpdate判断sra->t类型，然后创建不同的物理扫描，这个函数是递归函数，假如有where子句的话：sra\_t 的结构应该是SRASelect<-SRATable，所以会调用physical\_scan\_table\_create创建一个全部的物理计划，然后调用physical\_scan\_select\_create函数创建一个带有选择的物理扫描计划,并且获得选择的条件。

之后使用物理扫描获得的记录条目全部都是满足where子句条件的。然后计算出字段表达式的值，调用setInt/setString函数直接修改字段的值就可以了

1. 调用physical\_scan\_generate 函数创建一个物理扫描
2. 调用beforeFirst函数获得最底部的物理扫描，也就是全盘扫描，最底层的扫描，将读取的位置指向recordfile的开头位置
3. 使用循环结构不断获得下一条满足条件的记录plan->next(plan)。使用physical\_scan\_evaluate\_expression函数计算字段表达式的值，判断字段值的类型是整数类型还是字符串类型，分别调用SetInt和SetString函数修改字段的值
4. 最后调用plan->close(plan)关闭物理扫描

update语句解析流程图



update执行流程图