



基于PG数据库插件的SQL规范审核工具介绍



嘉宾：陈刚

公司：平安科技





个人简介

陈刚

- ✓ 中国平安集团旗下平安科技数据库架构师
- ✓ 13年工作经验
- ✓ 2011年加入平安科技
- ✓ 先后负责过多种类型数据库的数据库架构设计、新数据库产品的研究引入、数据库的规范制定与规范审核工具开发
- ✓ 对Java、C语言开发及源码研究比较感兴趣





目录

content

1

Hook技术基础简介

2

SQL执行过程分析

3

SQL规范审核插件介绍

4

未来与展望



Hook技术基础简介



HOOK技术简介

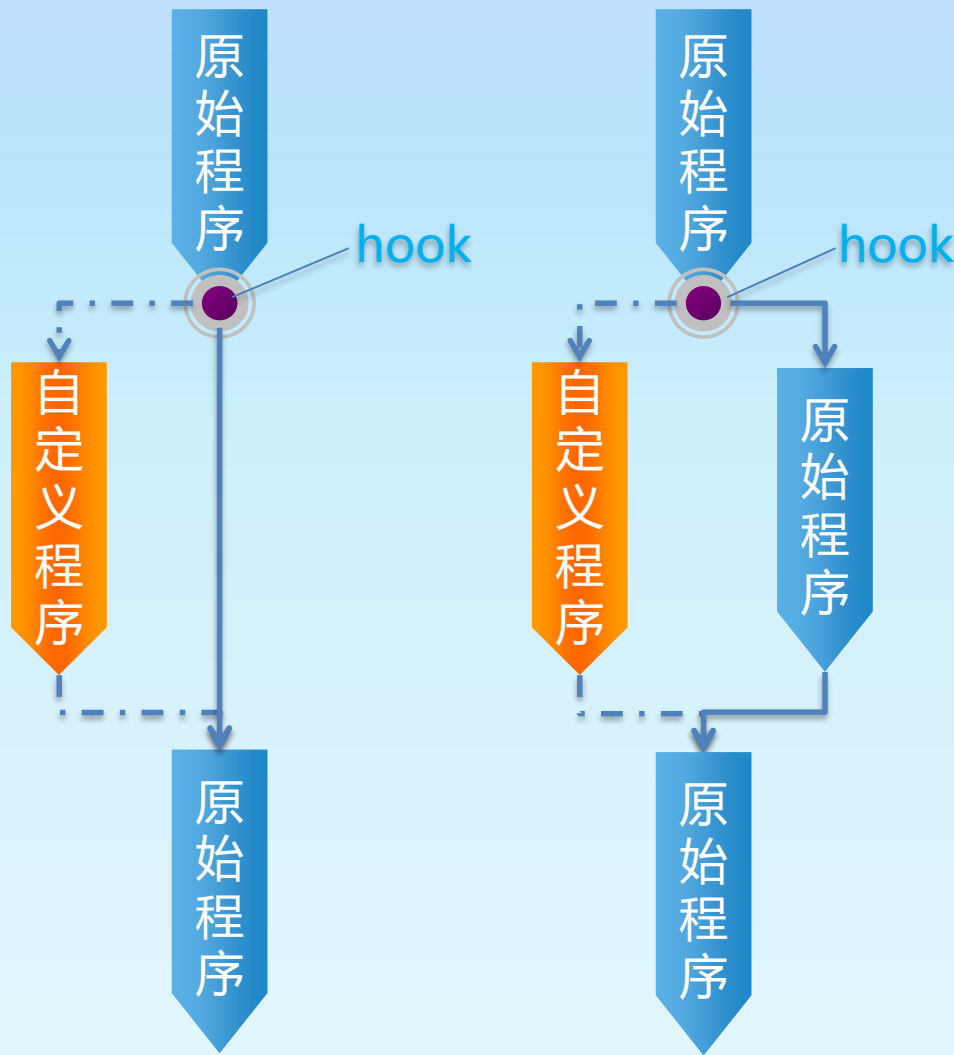
--基本概念

概念

HOOK机制，可以让用户有机会切入到PostgreSQL的内部运行机制中，进行中断、增加或修改原来的程序逻辑，从而实现一些用户自定义的功能。

四要素

- 定义hook
- 自定义程序
- 安装hook
- 卸载hook





HOOK技术简介

--代码样例

```

typedef void (*shmem_startup_hook_type) (void);
shmem_startup_hook_type shmem_startup_hook = NULL;

void createSharedMemoryAndSemaphores (bool
makePrivate, int port)
{
    .....
    if (shmem_startup_hook)
        shmem_startup_hook();
}

```

1

原始程序

```

static shmem_startup_hook_type prev_shmem_startup_hook
= NULL;

void _PG_init(void)
{
    .....
    prev_shmem_startup_hook = shmem_startup_hook;
    shmem_startup_hook = pgss_shmem_startup;
}

```

3

```

static void pgss_shmem_startup(void);

static void pgss_shmem_startup(void)
{
    if (prev_shmem_startup_hook)
        prev_shmem_startup_hook();

    //此处为自定义逻辑代码
    .....
}

```

2

自定义程序

```

void _PG_fini(void)
{
    shmem_startup_hook = prev_shmem_startup_hook;
}

```

4

其他辅助操作

- ❑ 参数配置(postgresql.conf) :
shared_preload_libraries = 'xxxx'
- ❑ 或者执行load命令 :
postgres=# LOAD 'xxxx';
- ❑ 必要时重启实例

注意事项

- ❑ 自定义程序性能
- ❑ 是否存在内存泄漏
- ❑ 做好容量评估
- ❑ 有异常处理机制

注：

PG内核程序

插件程序



HOOK技术简介

--auth_delay

```
typedef void (*ClientAuthentication_hook_type) (Port *, int);
ClientAuthentication_hook_type ClientAuthentication_hook =
NULL;
```

```
void ClientAuthentication(Port *port)
{
    .....
    if (ClientAuthentication_hook)
        (*ClientAuthentication_hook) (port, status);
}
```

1

原始程序

```
static ClientAuthentication_hook_type
original_client_auth_hook = NULL;
```

```
void _PG_init(void)
{
    .....
    original_client_auth_hook =
ClientAuthentication_hook;
    ClientAuthentication_hook = auth_delay_checks;
}
```

3

注：

PG内核程序

插件程序

```
static void
auth_delay_checks(Port *port, int status)
{
    if (original_client_auth_hook)
        original_client_auth_hook(port, status);

    if (status != STATUS_OK)
    {
        pg_usleep(1000L * auth_delay_milliseconds);
    }
}
```

2

自定义程序

原始程序



史上最简单的插件



HOOK技术简介

--常用HOOK

序号	HOOK名	源码位置
1	object_access_hook_type	src/backend/catalog/objectaccess.c
2	ExplainOneQuery_hook_type	src/backend/commands/explain.c
3	explain_get_index_name_hook_type	src/backend/commands/explain.c
4	check_password_hook_type	src/backend/commands/user.c
5	ExecutorStart_hook_type	src/backend/executor/execMain.c
6	ExecutorRun_hook_type	src/backend/executor/execMain.c
7	ExecutorFinish_hook_type	src/backend/executor/execMain.c
8	ExecutorEnd_hook_type	src/backend/executor/execMain.c
9	ExecutorCheckPerms_hook_type	src/backend/executor/execMain.c
10	ClientAuthentication_hook_type	src/backend/libpq/auth.c
11	set_rel_pathlist_hook_type	src/backend/optimizer/path/allpaths.c
12	join_search_hook_type	src/backend/optimizer/path/allpaths.c
13	set_join_pathlist_hook_type	src/backend/optimizer/path/joinpath.c
14	planner_hook_type	src/backend/optimizer/plan/planner.c
15	create_upper_paths_hook_type	src/backend/optimizer/plan/planner.c
16	get_relation_info_hook_type	src/backend/optimizer/util/plancat.c
17	post_parse_analyze_hook_type	src/backend/parser/analyze.c
18	row_security_policy_hook_type	src/backend/rewrite/rowsecurity.c
19	shmem_startup_hook_type	src/backend/storage/ipc/ipci.c
20	ProcessUtility_hook_type	src/backend/tcop/utility.c
21	get_relation_stats_hook_type	src/backend/utils/adt/selffuncs.c
22	get_index_stats_hook_type	src/backend/utils/adt/selffuncs.c
23	get_attavgwidth_hook_type	src/backend/utils/cache/lscache.c
24	emit_log_hook_type	src/backend/utils/error/elog.c
25	needs_fmgr_hook_type	src/backend/utils/fmgr/fmgr.c
26	fmgr_hook_type	src/backend/utils/fmgr/fmgr.c

注：基于V10.0版本统计

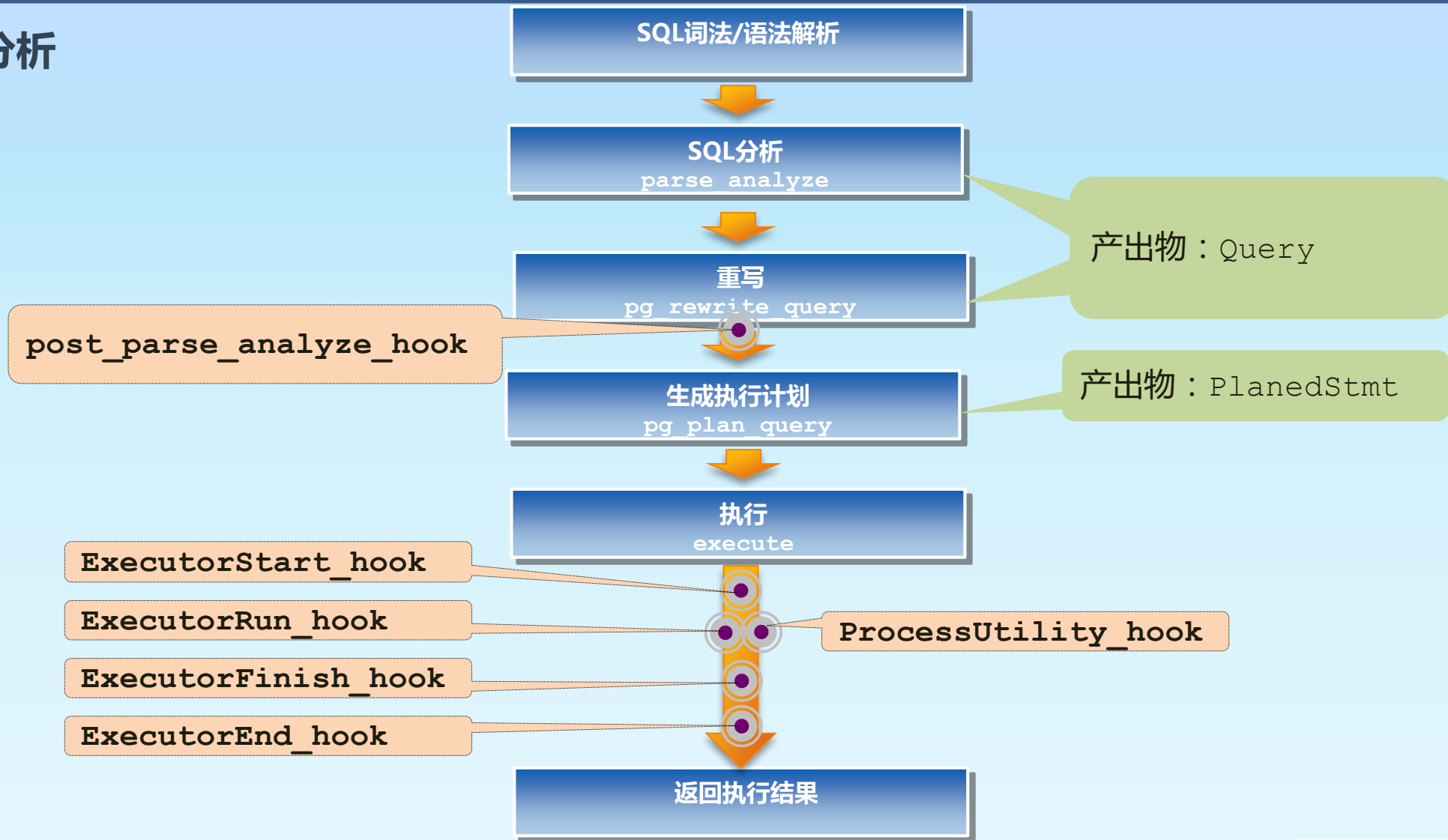


SQL执行过程分析



SQL执行过程分析

--执行过程分解





SQL执行过程分析

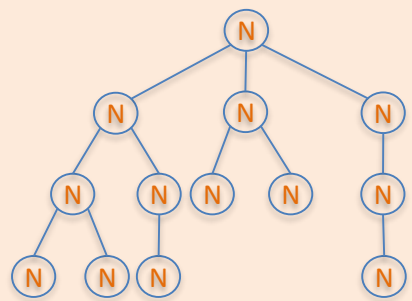
select id,col1 from t1 where id=1 order by id

--parse tree

```

{ QUERY
:commandType 1
:querySource 0
:canSetTag true
:utilityStmt <>
:resultRelation 0
:hasAggs false
:hasWindowFuncs false
:hasSubLinks false
:hasDistinctOn false
:hasRecursive false
:hasModifyingCTE false
:hasForUpdate false
:hasRowSecurity false
:cteList <>
:rtable (
  { RTE
  :alias <>
  :eref
  { ALIAS
  :aliasname t1
  :colnames ("id" "col1" "col2")
  }
  :rtekind 0
  :relid 123056
  :relkind r
  :tablesample <>
  :lateral false
  :inh true
  :inFromCl true
  :requiredPerms 2
  :checkAsUser 0
  :selectedCols (b 9 10)
  :insertedCols (b)
  :updatedCols (b)
  :securityQuals <>
  )
:jointree
  { FROMEXPR
  :fromlist (
    { RANGETBLREF
    :rtindex 1
    }
  )
  :quals
  { OEXPR
  :opno 96
  :opfuncid 65
  :opresulttype 16
  :opretset false
  :opcollid 0
  :inputcollid 0
  :args (
    { VAR
    :varno 1
    :varattno 1
    :vartype 23
    :vartypmod -1
    :varcollid 0
    :varlevelsup 0
    :varnoold 1
    :varoattno 1
    :location 29
    }
  )
  :targetList (
    { TARGETENTRY
    :expr
    { VAR
    :varno 1
    :varattno 1
    :vartype 23
    :vartypmod -1
    :varcollid 0
    :varlevelsup 0
    :varnoold 1
    :varoattno 1
    :location 7
    }
    :resno 1
    :resname id
    :ressortgroupref 1
    }
  )
  :resorigtbl 123056
  :resorigcol 1
  :resjunk false
  }
  { TARGETENTRY
  :expr
  { VAR
  :varno 1
  :varattno 2
  :vartype 23
  :vartypmod -1
  :varcollid 0
  :varlevelsup 0
  :varnoold 1
  :varoattno 2
  :location 10
  }
  :resno 2
  :resname col1
  :ressortgroupref 0
  :resorigtbl 123056
  :resorigcol 2
  :resjunk false
  }
  :onConflict <>
  :returningList <>
  :groupClause <>
  :groupingSets <>
  :havingQual <>
  :windowClause <>
  :distinctClause <>
  :sortClause (
    { SORTGROUPCLAUSE
    :tleSortGroupRef 1
    :eqop 96
    :sortop 97
    :nulls_first false
    :hashable true
    }
  )
  :limitOffset <>
  :limitCount <>
  :rowMarks <>
  :setOperations <>
  :constraintDeps <>
}

```





SQL执行过程分析

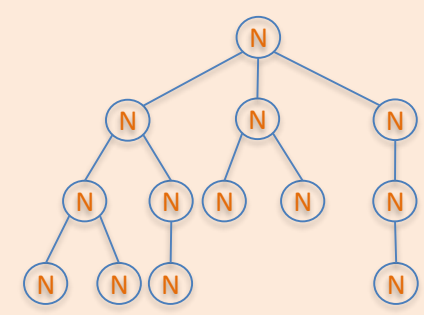
`select id,col1 from t1 where id=1 order by id`

--plan

```

{ PLANNEDSTMT
:commandType 1
:queryId 4258770327
:hasReturning false
:hasModifyingCTE false
:canSetTag true
:transientPlan false
:dependsOnRole false
:parallelModeNeeded false
:planTree
{SEQSCAN
:startup_cost 0.00
:total_cost 72755.05
:plan_rows 1
:plan_width 8
:parallel_aware false
:plan_node_id 0
:targetlist (
{TARGETENTRY
:expr
{VAR
:varno 1
:varattno 1
:vartype 23
:vartypmod -1
:varcollid 0
:varlevelsup 0
:varnoold 1
:varoattno 1
:location 7
}
}
:resno 1
:resname id
:ressortgroupref 1
:resorigtbl 123056
:resorigcol 1
:resjunk false
}
:resno 2
:resname col1
:ressortgroupref 0
:resorigtbl 123056
:resorigcol 2
:resjunk false
}
:qual (
{OPEXPR
:opno 96
:opfuncid 65
:opresulttype 16
:opretset false
:opcollid 0
:inputcollid 0
:args (
{VAR
:varno 1
:varattno 1
:vartype 23
:vartypmod -1
:varcollid 0
:varlevelsup 0
:varnoold 1
:varoattno 1
:location 29
}
{CONST
:consttype 23
:consttypmod -1
:constcollid 0
:constlen 4
:constbyval true
:constisnull false
:location 32
:constvalue 4 [ 1 0 0 0 0 0 0 ]
}
)
:location 31
}
)
:location 31
)
:extParam (b)
:allParam (b)
:scanrelid 1
}
:relationOids (o 123056)
:invalItems <>
:nParamExec 0
}
}
}
:rtable (
{RTE
:alias <>
:eref
{ALIAS
:aliasname t1
:colnames ("id" "col1" "col2")
}
:rtekind 0
:relid 123056
:relkind r
:tablesample <>
:lateral false
:inh false
:inFromCI true
:requiredPerms 2
:checkAsUser 0
:selectedCols (b 9 10)
:insertedCols (b)
:updatedCols (b)
:securityQuals <>
}
)
:resultRelations <>
:utilityStmt <>
:subplans <>
:rewindPlanIDs (b)
:rowMarks <>
}

```





SQL规范审核插件介绍



SQL规范审核插件

--定义规范项

RULE_NO	RULE_CONTENT	RULE_HINT	STATUS	LEVEL	EXTRA_INFO
PG-DML-001	UPDATE禁止出现ORDER BY子句	请取消UPDATE语句中的ORDER BY子句	ENABLED	WARNING	
PG-DML-002	UPDATE必须出现WHERE子句	请增加WHERE子句	ENABLED	WARNING	
PG-DML-003	可能UPDATE全表数据	请检查WHERE条件, 是否没有做行过滤	ENABLED	WARNING	
PG-DML-004	DELETE必须出现WHERE子句	请增加WHERE子句	ENABLED	WARNING	
PG-DML-005	DELETE禁止出现ORDER BY子句	请取消DELETE语句中的ORDER BY子句	ENABLED	WARNING	
PG-DML-006	可能DELETE全表数据	请检查WHERE条件, 是否没有做行过滤	ENABLED	WARNING	
PG-DML-007	禁止嵌套SELECT子句	请不要在SELECT查询的字段中再嵌套SELECT子查询	ENABLED	WARNING	
PG-DCL-001	用户名不符合规范	请使用规范的用户名	ENABLED	ERROR	postgres,dbmgr,dbmon,dbmonopr,deployop,edsop,edsqry,repmgr,repuser,pub_test,dtcoll,dmlbak,devsup*,*data,*opr,*tjs,*k tl,*sqp,*etl
PG-DCL-002	r*_qry角色授权违规	请不要将insert/update/delete/execute或all权限授权给QRY角色	ENABLED	ERROR	select,usage
PG-DCL-003	devsup用户授权违规	只能将r*_dev_qry权限授权给devsup用户	ENABLED	ERROR	r*_dev_qry
PG-DCL-004	edsqry用户授权违规	只能将r*_qry权限授权给edsqry用户	ENABLED	ERROR	r*_qry
PG-DDL-001	DATE类型字段命名违规	Date 类型的字段, 需要以"date_"作为前缀或者"_date"做为后缀	ENABLED	WARNING	
PG-DDL-002	索引名不符合规范	索引命名规则为: IDX_表名称_字段名	ENABLED	WARNING	



SQL规范审核插件

--规范审核案例1

- 利用post_parse_analyze_hook这个hook
- 对Query查询树进行分析
- 判断where条件Node是否为空

RULE_NO	RULE_CONTENT	RULE_HINT	STATUS	LEVEL	EXTRA_INFO
PG-DML-002	UPDATE必须出现WHERE子句	请增加WHERE子句	ENABLED	WARNING	

update t1 set col1=0 where id=1;

update t1 set col1=0;

```

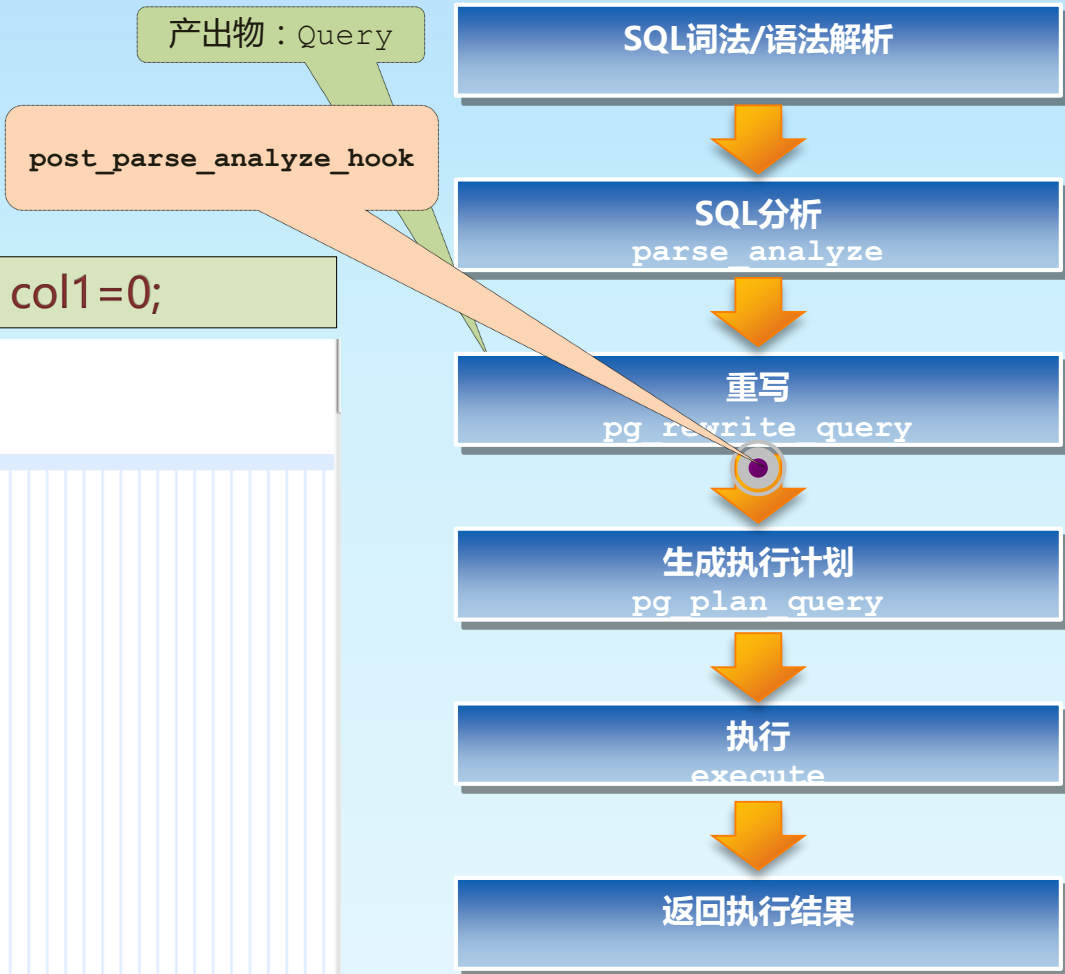
40  :jointree
41  {FROMEXPR
42  :fromlist (
43  {RANGETBLREF
44  :rtindex 1
45  }
46  )
47  :quals
48  {OPEXPR
49  :opno 96
50  :opfuncid 65
51  :opresulttype 16
52  :opretset false
53  :opcollid 0
54  :inputcollid 0
55  :args (
56  {VAR
57  :varno 1
58  :varattno 1
59  :vartype 23
60  :vartypmod -1
61  :varcollid 0
62  :varlevelsup 0
63  :varnoold 1
64  :varoattno 1
65  :location 27
66  }
67  {CONST
68  :consttype 23
69  :consttypmod -1
70  :constcollid 0
71  :constlen 4
72  :constbyval true
73  :constisnull false
74  :location 30
75  :constvalue 4 [ 1 0 0 0 0 0 0 0
76  ]
77  }
78  :location 29
79  )
80  }
81  :targetList (
82  {TARGETENTRY
83  :expr

```

```

40  :jointree
41  {FROMEXPR
42  :fromlist (
43  {RANGETBLREF
44  :rtindex 1
45  }
46  )
47  :quals <>
48  }
49  :targetList (
50  {TARGETENTRY
51  :expr

```



产出物: Query

post_parse_analyze_hook





SQL规范审核插件

--规范审核案例2

- 利用ExecutorStart_hook这个hook
- 对PlannedStmt树进行分析
- 判断(scan)node->plan.qual == NULL?

RULE_NO	RULE_CONTENT	RULE_HINT	STATUS	LEVEL	EXTRA_INFO
PG-DML-003	可能UPDATE全表数据	请检查WHERE条件，是否没有做行过滤	ENABLED	WARNING	

update t1 set col1=0 where id=1;

update t1 set col1=0 where 1=1;
update t1 set col1=0 where true;

产出物: PlannedStmt

ExecutorStart_hook

SQL词法/语法解析

SQL分析
parse analyze

重写

pg rewrite query

生成执行计划

pg plan query

执行

execute

返回执行结果

```

373         :resjunk true
374     }
375 }
376 :qual {
377     {OPEXPR
378         :opno 96
379         :opfuncid 65
380         :opresulttype 16
381         :opretset false
382         :opcollid 0
383         :inputcollid 0
384         :args {
385             {VAR
386                 :varno 1
387                 :varattno 1
388                 :vartype 23
389                 :vartypmod -1
390                 :varcollid 0
391                 :varlevelsup 0
392                 :varnoold 1
393                 :varoattno 1
394                 :location 27
395             }
396             {CONST
397                 :consttype 23
398                 :consttypmod -1
399                 :constcollid 0
400                 :constlen 4
401                 :constbyval true
402                 :constisnull false
403                 :location 30
404                 :constvalue 4 [ 1 0 0 0 0 0 0 0 ]
405             }
406         }
407         :location 29
408     }
409 }
410 :lefttree <>
411 :righttree <>

```

```

372         :resjunk true
373     }
374 }
375 :qual <>
376 :lefttree <>
377 :righttree <>

```




SQL规范审核插件

--规范审核案例3

RULE_NO	RULE_CONTENT	RULE_HINT	STATUS	LEVEL	EXTRA_INFO
PG-DCL-002	r_*_qry角色授权违规	请不要将insert/update/delete/execute或all权限授权给QRY角色	ENABLED	ERROR	select, usage

grant insert,update on t1 to r_demodata_qry;

➤ 利用ProcessUtility_hook这个hook

➤ 对GrantStmt进行分析

➤ 对(GrantStmt) stmt->grantee-> rolename

和(GrantStmt) stmt->privilege->priv_name的内容进行判断

```

foreach(cell, stmt->grantees)
{
    RoleSpec *grantee = (RoleSpec *) lfirst(cell);
    if (is_begin_with(grantee->rolename, "r_") == 1
        && is_end_with(grantee->rolename, "_qry") == 1)
        existsQRYrule = true;
}
if (!existsQRYrule)
    return;
if (stmt->privileges == NIL) {
    flag = true;
} else {
    foreach(cell, stmt->privileges)
    {
        AccessPriv *privnode = (AccessPriv *) lfirst(cell);
        flag1 = false;
        foreach(cell1, privilegeList)
        {
            char *priv = (char *) lfirst(cell1);
            if (strcmp(privnode->priv_name, priv) != 0) {
                flag1 = true;
                break;
            }
        }
        if (flag1) {
            flag = true;
            break;
        }
    }
}
}

```

产出物: GrantStmt

ProcessUtility_hook

SQL词法/语法解析



SQL分析

parse analyze



重写

pg rewrite query



生成执行计划

pg plan query



执行

execute



返回执行结果

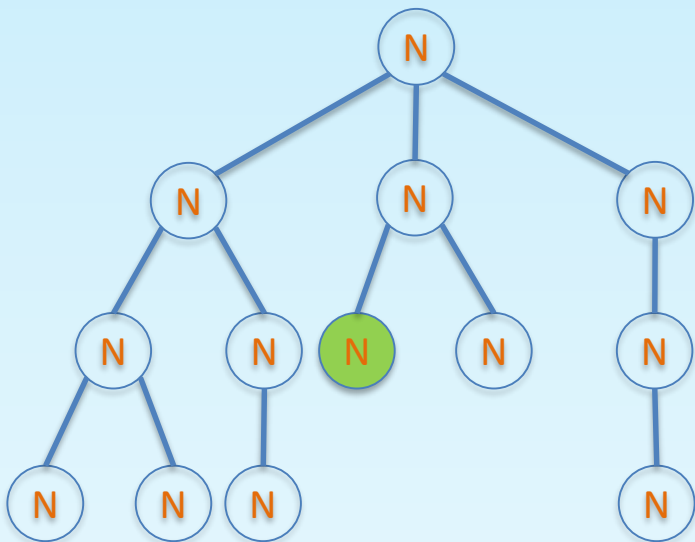




SQL规范审核插件

-- 查询树遍历

- 递归遍历树中的每个Node
- 逐一判断每个Node的NodeTag
- 根据审核规则判断对应Node的内容是
否符合某种条件



```

plan:
{PLANNEDSTMT
:commandType 2
:queryId 2148518803
:hasReturning false
.....
:planTree
{MODIFYTABLE
:startup_cost 0.00
:total_cost 72744.75
.....
:resultRelations (i 1)
:resultRelIndex 0
:plans (
{SEQSCAN
:startup_cost 0.00
:total_cost 72744.75
:plan_rows 1575
.....
:resorigcol 0
:resjunk true
}
)
:qual <>
:lefttree <>
:righttree <>
:initPlan <>
:extParam (b 0)
:allParam (b 0)
:scanrelid 1
}
)

```

```

void
outNode(StringInfo str, const void *obj)
{
    if (obj == NULL)
        appendStringInfoString(str, "<>");
    else if (IsA(obj, List) || IsA(obj, IntList) || IsA(obj,
OidList))
        _outList(str, obj);
    else if (IsA(obj, Integer) || IsA(obj, Float) ||
IsA(obj, String) || IsA(obj, BitString))
        { _outValue(str, obj); }
    else {
        appendStringInfoChar(str, '{');
        switch (nodeTag(obj))
        {
            case T_PlannedStmt:
                _outPlannedStmt(str, obj);
                break;
            case T_Plan:
                _outPlan(str, obj);
                break;
            .....
            case T_XmlSerialize:
                _outXmlSerialize(str, obj);
                break;
            case T_ForeignKeyCacheInfo:
                _outForeignKeyCacheInfo(str, obj);
                break;
            default:
                break;
        }
        appendStringInfoChar(str, '}');
    }
}

```

源码位置 : [src/backend/nodes/outfuncs.c](#)

```

bool traversalNode(const char *ruleNo, const void *obj)
{
    if (obj == NULL) return false;
    else if (IsA(obj, List) || IsA(obj, IntList) || IsA(obj,
OidList))
        return traversalNode_List(ruleNo, obj);
    else {
        switch (nodeTag(obj)) {
            case T_Query:
                return traversalNode_Query(ruleNo, obj);
                break;
            case T_SubLink:
                return traversalNode_SubLink(ruleNo, obj);
                break;
            .....
            case T_SeqScan:
                return traversalNode_SeqScan(ruleNo, obj);
                break;
            default:
                break;
        }
    }
    return false;
}

```

```

static bool traversalNode_SeqScan(const char *ruleNo,
const SeqScan *node) {
    if (strcmp(ruleNo, "PG-DML-003") == 0
        || strcmp(ruleNo, "PG-DML-006") == 0) {
        if (node->plan.qual == NULL) {
            return true;
        }
    }
    return false;
}

```





SQL规范审核插件

--两张辅助表

1 pg_sql_check_rule 规则表

```
postgres=# select * from pg_sql_check_rule;
```

rule_no	rule_content	rule_hint	status	level
PG-DML-001	UPDATE禁止出现ORDER BY子句	请取消UPDATE语句中的ORDER BY子句	ENABLED	WARNING
PG-DML-002	UPDATE必须出现WHERE子句	请增加WHERE子句	ENABLED	WARNING
PG-DML-003	可能UPDATE全表数据	请检查WHERE条件, 是否没有做行过滤	ENABLED	WARNING
PG-DML-004	DELETE必须出现WHERE子句	请增加WHERE子句	ENABLED	WARNING
PG-DML-005	DELETE禁止出现ORDER BY子句	请取消DELETE语句中的ORDER BY子句	ENABLED	WARNING
PG-DML-006	可能DELETE全表数据	请检查WHERE条件, 是否没有做行过滤	ENABLED	WARNING
PG-DML-007	禁止嵌套SELECT子句	请不要在SELECT查询的字段中再嵌套SELECT子查询	ENABLED	WARNING

2 pg_sql_check_log 审核违规日志表

```
postgres=# select * from pg_sql_check_log;
```

application_name	check_date	queryid	rule_no	rule_content	query	rule_hint	level	datid	datname	usesysid	username
psql	2017-06-16 10:33:39.614251+08	4068198720	PG-DML-006	可能DELETE全表数据	delete from t2 where true;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:33:46.424295+08	390658016	PG-DML-003	可能UPDATE全表数据	update t2 set coll=1 where 1=1;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.479283+08	3247814783	PG-DML-001	UPDATE禁止出现ORDER BY子句	update t2 set coll=1 where id in (select id from t1 where coll=1 order by col2);	请取消UPDATE语句中的ORDER BY子句	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.490815+08	165823437	PG-DML-002	UPDATE必须出现WHERE子句	update t2 set coll=1;	请增加WHERE子句	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.492552+08	165823437	PG-DML-003	可能UPDATE全表数据	update t2 set coll=1;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.496084+08	390658016	PG-DML-003	可能UPDATE全表数据	update t2 set coll=1 where 1=1;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.50081+08	257380939	PG-DML-003	可能UPDATE全表数据	update t2 set coll=1 where true;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.504195+08	1528877246	PG-DML-005	DELETE禁止出现ORDER BY子句	delete from t2 where id in (select id from t1 where coll=1 order by col2);	请取消DELETE语句中的ORDER BY子句	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.511674+08	307860557	PG-DML-006	可能DELETE全表数据	delete from t2 where 1=1;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.5157+08	4068198720	PG-DML-006	可能DELETE全表数据	delete from t2 where true;	请检查WHERE条件, 是否没有做行过滤	WARNING	13269	postgres	123069	demo
psql	2017-06-16 10:35:49.517257+08	1999229256	PG-DML-007	禁止嵌套SELECT子句	select id,(select coll from t1 where t1.id=t2.id) from t2;	请不要在SELECT查询的字段中再嵌套SELECT子查询	WARNING	13269	postgres	123069	demo





SQL规范审核插件

--三个辅助存储过程

1 **Setsqlcheckrulestatus()**
设置某规则是否审核

```
postgres=# select setsqlcheckrulestatus('PG-DML-001','DISABLED');
setsqlcheckrulestatus
-----
DISABLED
(1 row)
```

2 **Setsqlcheckrulelevel()**
设置某规则警告级别

```
postgres=# select setsqlcheckrulelevel('PG-DML-001','WARNING');
setsqlcheckrulelevel
-----
WARNING
(1 row)
```

3 **Clearsqlchecklog()**
清理审核违规日志表

```
postgres=# select clearsqlchecklog();
clearsqlchecklog
-----
TABLE pg_sql_check TRUNCATED
(1 row)
```



SQL规范审核插件

--最终效果

```
postgres=# select id,(select coll from t1 where t1.id=t2.id) from t2;
WARNING:  (PG-DML-007)禁止嵌套select子句
HINT:  请不要在select查询的字段中再嵌套select子查询
```

```
postgres=# update t1 set coll=0 where 1=1;
WARNING:  (PG-DML-003)可能update全表数据
HINT:  请检查where条件, 是否没有做行过滤
UPDATE 1000
```

```
postgres=# update t1 set coll=0 where true;
WARNING:  (PG-DML-003)可能update全表数据
HINT:  请检查where条件, 是否没有做行过滤
UPDATE 1000
```

```
postgres=# update t1 set coll=0;
WARNING:  (PG-DML-002)update必须出现where子句
HINT:  请增加where子句
WARNING:  (PG-DML-003)可能update全表数据
HINT:  请检查where条件, 是否没有做行过滤
UPDATE 1000
```

```
postgres=# delete from t2 ;
WARNING:  (PG-DML-004)delete必须出现where子句
HINT:  请增加where子句
WARNING:  (PG-DML-006)可能delete全表数据
HINT:  请检查where条件, 是否没有做行过滤
DELETE 999
```

```
postgres=# delete from t2 where id in (select id from t1 where coll=1 order by col2);
WARNING:  (PG-DML-005)DELETE禁止出现ORDER BY子句
HINT:  请取消delete语句中的order by子句
```

```
postgres=# update t2 set coll=1 where id in (select id from t1 where coll=0 order by col2);
WARNING:  (PG-DML-001)update禁止出现order by子句
HINT:  请取消update语句中的order by子句
UPDATE 1000
```




SQL规范审核插件

--特点总结

□ 轻量级

使用原生的SQL解析器，不需要额外对SQL语句进行分析解析，所以对数据库内核性能影响极小

□ 只读、安全

对内核没有任何修改操作，只是读取SQL解析后的查询树等中间产物，所以安全可靠

□ 兼容性强

目前兼容的版本包括：

✓ PG 9.4.x / 9.5.x / 9.6.x / 10.0

✓ EDB 9.4.x / 9.5.x / 9.6.x

□ 灵活可控

可以随时启用或禁用某些审核项，并且可以灵活控制违规级别（警告或拦截）



未来与展望



增加更多的SQL规范审核项

增加SQL智能优化建议的功能

开发出更多实用的插件





Thanks!