



Simba C++ SQL Engine

Reference Guide

Version 10.3

August 2024

Copyright

This document was released in August 2024.

Copyright ©2014–2024 insightsoftware. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written permission from insightsoftware.

The information in this document is subject to change without notice. insightsoftware strives to keep this information accurate but does not warrant that this document is error-free.

Any insightsoftware product described herein is licensed exclusively subject to the conditions set forth in your insightsoftware license agreement.

Simba, the Simba logo, SimbaEngine, and Simba Technologies are registered trademarks of Simba Technologies Inc. in Canada, the United States and/or other countries. All other trademarks and/or servicemarks are the property of their respective owners.

All other company and product names mentioned herein are used for identification purposes only and may be trademarks or registered trademarks of their respective owners.

Information about the third-party products is contained in a third-party-licenses.txt file that is packaged with the software.

Contact Us

insightsoftware

www.insightsoftware.com

About This Guide

Purpose

This guide explains how to use the Simba C++ SQL Engine to document what SQL grammar the C++ engine supports.

Audience

The guide is intended for developers who have created a connector with the Simba C++ SQL Engine. This guide is also intended for end users of the Simba C++ SQL Engine.

Knowledge Prerequisites

To use the Simba C++ SQL Engine, the following knowledge is helpful:

- Familiarity with the platform on which you are using the Simba C++ SQL Engine.
- Ability to use the data store to which the Simba C++ SQL Engine is connecting.
- An understanding of the role of ODBC or JDBC technologies and driver managers in connecting to a data store.
- Experience creating and configuring ODBC or JDBC connections.

Variables Used in this Document

The following variables are used in this document:

Variable	Description
<i>[DRIVER_NAME]</i>	The name of your connector, as used in Windows registry keys and names of configuration files.
<i>[INSTALL_DIR]</i>	Installation directory for the Simba C++ SQL Engine.

Contents

Copyright	2
About This Guide	3
Purpose	3
Audience	3
Knowledge Prerequisites	3
Variables Used in this Document	3
Contents	4
Introducing the Simba C++ SQL Engine	6
About	6
Specifications	8
SQL Engine Parser	9
SQL Engine SQL Commands	10
Functions	11
Aggregate Functions	11
Scalar Function	11
Date and Time Functions	15
Joins	17
SELECT or Subquery Key Words	18
Simba Engine SQL Keywords	20
Appendix A: BNF Grammar for C++ SQL Engine with Railroad Diagram	27
Appendix 2: Token Definitions	137
Reference	146

Contact Us	147
Third-Party Trademarks	148

Introducing the Simba C++ SQL Engine

An ODBC SQL Parsing and Execution Engine for Database Access, Simba C++ SQL Engine is a component of SimbaEngine. It provides SQL processing in data drivers and data providers for non-SQL data stores built with SimbaEngine. Simba C++ SQL Engine can also be used in custom solutions where SQL parsing or execution is required.

About

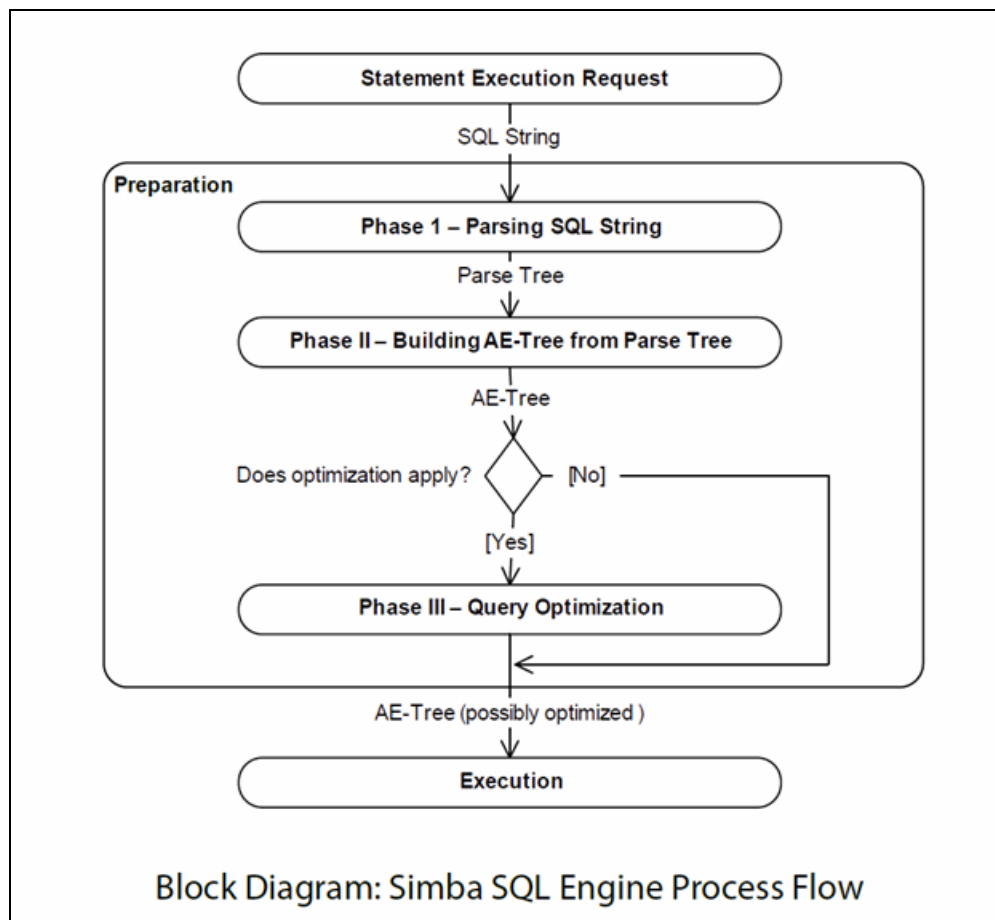
Simba C++ SQL Engine is an easy-to-use, complete, high-performance SQL parser and execution engine for building ODBC, JDBC, ADO.NET or OLE DB data access solutions for non-SQL data stores. You can use it with other SimbaEngine components to create either local or remote data driver access to non-SQL data sources – such as web service, object-oriented, network, real time (e.g. SCADA) and ISAM databases – or you can use it in a custom solution and call it from your own code. Simba SQL Engine's clean and logical interfaces make it easy to work with, and it can either parse, or parse and execute a SQL statement and retrieve the result set. Included with the SimbaEngine is the Quickstart ODBC driver that is a working example of an ODBC driver built using SQL Engine that you can modify to demonstrate reading your own data in just a few days.

The Simba C++ SQL Engine parser conforms to the SQL-92 specification, but it can be extended to implement any SQL function or feature. When Simba C++ SQL Engine prepares a SQL statement, it validates the tables and columns referenced in the statement against the metadata from the underlying data store. Then it creates an intermediate version of the query called the Algebraic Expression Tree (AE-Tree).

The AE-Tree represents the metadata for the query, and is eventually transformed into an Execution Tree (E-Tree), which can be executed. By carefully moving or modifying nodes, the AE-Tree can be updated to change the way the query is executed without changing the result. The underlying data store can modify the AE-Tree, so that when the query is executed, Simba C++ SQL Engine delegates some or all of the processing to the underlying data store. We call this Collaborative Query Execution. The advantage is that any high-performance processing available in the data store is available for use with Simba C++ SQL Engine. If the entire query can be delegated, Simba C++ SQL Engine will only retrieve the result set. This is particularly important for analytical databases with built-in, high-performance aggregation processing that end users expect to see even through ODBC.

Simba C++ SQL Engine is a complete, compatible, high-performance and easy to use solution for SQL processing on non-SQL data stores. Everything you need to prepare and execute SQL queries is included with a clean and logical interface. With compliance to the SQL-92 standard, it is carefully designed to support common reporting and BI tools, so end users can get on with their work. Collaborative Query Execution allows you to use the high-performance processing built into your own data store to boost the performance of Simba C++ SQL Engine. You can use Simba C++ SQL Engine with the other components of SimbaEngine to quickly deploy local or remote data access, or you can use it alone, called from your own code. However you choose to deploy it, Simba C++ SQL Engine is the easiest and most flexible SQL preparation and execution engine available today.

This flow of control is illustrated below:



Specifications

The following list details the specifications for the Simba C++ SQL Engine:

- SQL-92-compliant parser with comprehensive support for ODBC 3.80-compliant data types, as well as scalar and aggregate functions
- ANSI and Unicode UTF-8, -16 and -32 support for both data and metadata.
- Available in 32-bit and 64-bit versions for Windows, Linux and UNIX, OSX, and more.
- Maximum number of simultaneous connections limited only by memory.
- Maximum result set length limited only by memory and swap table space.
- Maximum 65,356 columns supported in a select list.
- Maximum 65,356 columns supported in a table.
- Maximum number of characters in a SQL statement is configurable and is only limited by memory.
- Maximum number of joined tables limited only by memory and swap table space.

SQL Engine Parser

The following is a list of commands that the SQL Engine Parser supports that the Execution component does not support. This means that a parse tree can be generated, however executing using the Execution component will fail.

- ALL PRIVILEGES
- ALTER TABLE
- CASCADE
- CREATE VIEW
- DROP VIEW
- EXCEPT
- EXCEPT ALL
- GRANT
- GRANT OPTION FOR
- INTERSECT
- INTERSECT ALL
- PUBLIC
- REFERENCES
- RESTRICT
- REVOKE
- SET CATALOG
- SET SCHEMA
- USAGE
- WITH GRANT OPTION

SQL Engine SQL Commands

This section discusses commands that are supported by both the Parser and Execution components of the SQL Engine:

- CREATE TABLE
- CREATE VIEW
- DELETE
- DROP TABLE
- DROP VIEW
- INSERT
- SELECT
- UPDATE
- UPSERT

Functions

Simba Engine SQL has many built-in functions for performing calculations on data. There are several basic types and categories of functions. The following sections list some SQL Functions supported by Simba Engine SQL.

Aggregate Functions

An aggregate function takes a set of values (like a column of data) and returns a single value result from the set of values. As of SimbaEngine 9.3, custom aggregates are supported (excluding the ETree), which means that any aggregate function can be supported.

Aggregate Functions	SimbaEngine SQL Supported
AVG	Yes
COUNT	Yes
MAX	Yes
MIN	Yes
STDDEV	Yes
STDDEV_POP	Yes
SUM	Yes
VAR	Yes
VAR_POP	Yes

Scalar Function

A scalar function takes input argument(s) and returns a single value result. Following is a list of scalar function which supported by SimbaEngine SQL. As of SimbaEngine 9.3, custom scalars are supported, which means that any scalar function can be supported.

Scalar Function	SimbaEngine SQL Supported
String Functions	
ASCII	Yes
CHAR	Yes
CONCAT	Yes
LCASE	Yes
LEFT	Yes
LENGTH	Yes
LOCATE	Yes
LTRIM	Yes
INSERT	Yes
REPEAT	Yes
REPLACE	Yes
RIGHT	Yes
RTRIM	Yes
SOUNDEX	Yes
SPACE	Yes
SUBSTRING	Yes
UCASE	Yes

Scalar Function	SimbaEngine SQL Supported
BIT_LENGTH	No
CHAR_LENGTH	No
CHARACTER_LENGTH	No
DIFFERENCE	No
OCTET_LENGTH	No
POSITION(_IN_)	No
Numeric Functions	
ABS	Yes
ACOS	Yes
ASIN	Yes
ATAN	Yes
ATAN2	Yes
CEILING	Yes
COS	Yes
COT	Yes
DEGREES	Yes
EXP	Yes
FLOOR	Yes

Scalar Function	SimbaEngine SQL Supported
LOG	Yes
LOG10	Yes
MOD	Yes
PI	Yes
POWER	Yes
RADIANS	Yes
RAND	Yes
ROUND	Yes
SIGN	Yes
SIN	Yes
SQRT	Yes
TAN	Yes
TRUNCATE	Yes
System Functions	
CAST	Yes
CONVERT	Yes
DATABASE	Yes
IFNULL	Yes

Scalar Function	SimbaEngine SQL Supported
USER	Yes
SimbaEngine SQL	
COALESCE	Yes
NULL	Yes
NULLIF	Yes

Date and Time Functions

The following is a table of all important Date and Time related functions available through Simba Engine SQL.

Date and Time and Interval Functions	SimbaEngine SQL Supported
CURRENT_DATE	Yes
CURRENT_TIME	Yes
CURRENT_TIME(time-precision)	Yes
CURRENT_TIMESTAMP	Yes
CURRENT_TIMESTAMP(time-precision)	Yes
CURDATE	Yes
CURTIME	Yes
DAYNAME	Yes
DAYOFMONTH	Yes
DAYOFWEEK	Yes

Date and Time and Interval Functions SimbaEngine SQL Supported	
DAYOFYEAR	Yes
HOUR	Yes
MINUTE	Yes
MONTH	Yes
MONTHNAME	Yes
NOW	Yes
DAYNAME	Yes
DAYOFMONTH	Yes
DAYOFWEEK	Yes
DAYOFYEAR	Yes
HOUR	Yes
MINUTE	Yes
MONTH	Yes
MONTHNAME	Yes
NOW	Yes
QUARTER	Yes
SECOND	Yes
TIMESTAMPDIFF	Yes

Date and Time and Interval Functions	SimbaEngine SQL Supported
TIMESTAMPADD	Yes
WEEK	Yes
WEEK_ISO	Yes
YEAR	Yes
EXTRACT	Yes

Joins

A SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them. SimbaEngine SQL supports all the join types which show in following table:

Joins	SimbaEngine SQL Supported
CROSS JOIN	Yes
JOIN	Yes
INNER	Yes
LEFT OUTER	Yes
LEFT	Yes
RIGHT OUTER	Yes
RIGHT	Yes
FULL	Yes
FULL OUTER	Yes

SELECT or Subquery Key Words

A subquery is a query within a query. A subquery is usually added in the WHERE Clause of the sql statement, and is usually used when you know how to search for a value using a SELECT statement but do not know the exact value in the database. Subqueries are an alternate way of returning data from multiple tables and can be used with the comparison operators, for example "=", "<>", "<", ">", "<=" and ">=". They may contain any of the keywords used in the SELECT statement.

The following are other keywords that can be used within a SELECT statement, in any of the associated clauses:

Subquery Key Words	SimbaEngine SQL Supported
ALL	Yes
ANY	Yes
DISTINCT	Yes
GROUP BY	Yes
HAVING	Yes
LIMIT	Yes
TOP	Yes
UNION	Yes
UNION ALL	Yes
VALUES	Yes
AS	Yes
ASC	Yes
DATATYPE (one of the SQL_* types)	Yes
TSIDATATYPE (one of the SQL_TSI_* types)	Yes

Subquery Key Words	SimbaEngine SQL Supported
DESC	Yes
DEFAULT	Yes
WHERE	Yes
OR	Yes
AND	Yes
NOT	Yes
EQ (=)	Yes
NE (<>)	Yes
LT (<)	Yes
GT (>)	Yes
LE (<=)	Yes
GE (>=)	Yes
NOT BETWEEN	Yes
BETWEEN	Yes
NOT IN	Yes
IN	Yes
NOT LIKE	Yes
LIKE	Yes

Subquery Key Words	SimbaEngine SQL Supported
ESCAPE	Yes
IS NOT NULL	Yes
IS NULL	Yes
EXISTS	Yes

Simba Engine SQL Keywords

Keywords cannot be used as identifiers (names of database objects such as columns and tables) without quoting. Simba Engine SQL reserved keywords for defining, manipulating, and accessing databases.

Key Words	SimbaEngine SQL Supported
ADD *	Yes
ALL	Yes
ALTER *	Yes
AND	Yes
ANY	Yes
AS	Yes
ASC	Yes
AVG	Yes
BETWEEN	Yes
BY	Yes

Key Words	SimbaEngine SQL Supported
CALL	Yes
CASCADE *	Yes
CASE	Yes
CAST	Yes
CATALOG *	Yes
CHECK	Yes
COALESCE	Yes
COLUMN	Yes
CONVERT	Yes
COUNT	Yes
CREATE	Yes
CROSS	Yes
DATE	Yes
DAY	Yes
DEFAULT	Yes
DELETE	Yes
DESC	Yes
DISTINCT	Yes

Key Words	SimbaEngine SQL Supported
DROP	Yes
ELSE	Yes
END	Yes
ESCAPE	Yes
EXCEPT *	Yes
EXISTS	Yes
FN	Yes
FOR	Yes
FOREIGN *	Yes
FROM	Yes
FULL	Yes
GRANT *	Yes
GROUP	Yes
HAVING	Yes
HOUR	Yes
IF	Yes
IN	Yes
INDEX	Yes

Key Words	SimbaEngine SQL Supported
INNER	Yes
INSERT	Yes
INTERVAL	Yes
INTO	Yes
IS	Yes
JOIN	Yes
KEY	Yes
LEFT	Yes
LIKE	Yes
LIMIT	Yes
MAX	Yes
MIN	Yes
MINUTE	Yes
MONTH	Yes
NOT	Yes
NULL	Yes
NULLIF	Yes
ON	Yes

Key Words	SimbaEngine SQL Supported
OPTION *	Yes
OR	Yes
ORDER	Yes
OUTER	Yes
PERCENT	Yes
PRIMARY	Yes
PRIVILEGE *	Yes
PROCEDURE	Yes
PUBLIC *	Yes
REFERENCES *	Yes
RESTRICT *	Yes
REVOKE *	Yes
RIGHT	Yes
SCHEMA *	Yes
SECOND	Yes
SELECT	Yes
SET	Yes
SOME *	Yes

Key Words	SimbaEngine SQL Supported
STDDEV	Yes
STDDEV_POP	Yes
SUM	Yes
SVBEGIN	Yes
SVEND	Yes
TABLE *	Yes
THEN	Yes
TIME	Yes
TIMESTAMP	Yes
TIMESTAMPADD	Yes
TIMESTAMPDIFF	Yes
TO	Yes
TOP	Yes
UNION	Yes
UNIQUE	Yes
UPDATE	Yes
USAGE *	Yes
USER	Yes

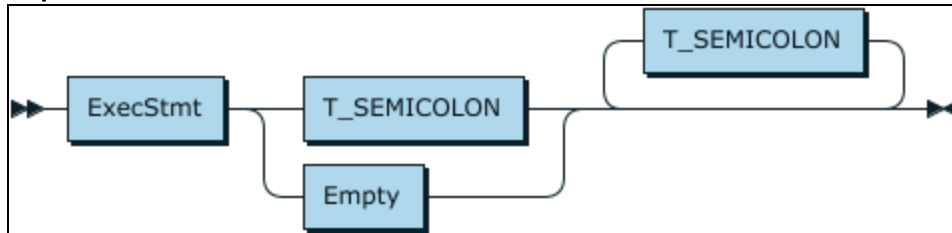
Key Words	SimbaEngine SQL Supported
VALUES	Yes
VAR	Yes
VAR_POP	Yes
VIEW *	Yes
WHEN	Yes
WHERE	Yes
WITH	Yes
YEAR	Yes

* Parser Support only. This is not supported by the AETree or ETree components.

Appendix A: BNF Grammar for C++ SQL Engine with Railroad Diagram

The following is the Rail Diagrams for the SQL BNF Grammar with their relations.

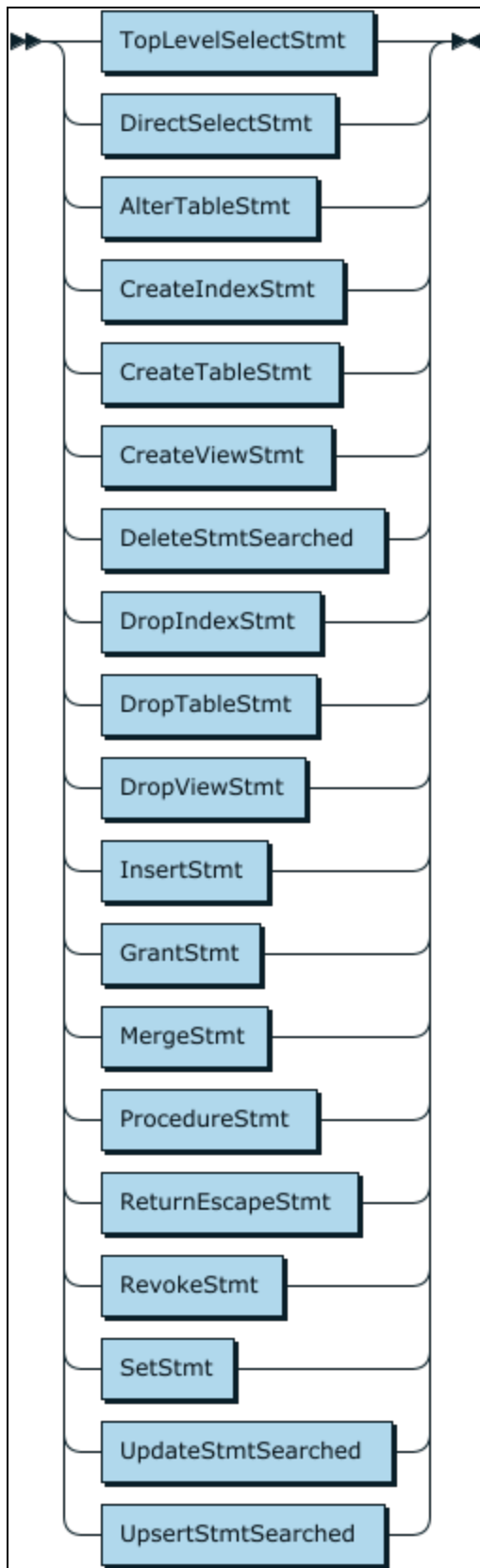
TopLevel:



`TopLevel ::= ExecStmt ( T_SEMICOLON | Empty ) T_SEMICOLON*`

no references

ExecStmt:



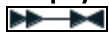
ExecStmt ::= TopLevelSelectStmt

- | DirectSelectStmt
- | AlterTableStmt
- | CreateIndexStmt
- | CreateTableStmt
- | CreateViewStmt
- | DeleteStmtSearched
- | DropIndexStmt
- | DropTableStmt
- | DropViewStmt
- | InsertStmt
- | GrantStmt
- | MergeStmt
- | ProcedureStmt
- | ReturnEscapeStmt
- | RevokeStmt
- | SetStmt
- | UpdateStmtSearched
- | UpsertStmtSearched

referenced by:

- TopLevel

Empty:



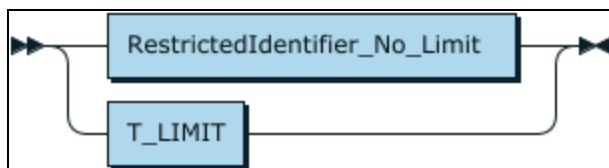
Empty ::=

referenced by:

- AS_Clause_Opt
- AS_Opt
- CastFormat_Opt
- ColumnConstraintDefinition_Opt
- ColumnDefinitionList_Opt
- ColumnNameList_Opt
- CorrelationSpec_Opt
- CreateTableStmt
- CreateViewStmt
- DataTypeAttributeList_Opt
- DataType_Opt
- DerivedColumnList_Opt
- ElseClause_Opt
- EscapeCharacter_Opt
- GrantOptionFor_Opt
- GroupBy_Opt
- Having_Opt
- Index_Type_Opt
- IntervalFractionalSecondsPrecision_Opt
- IntervalLeadingPrecision_Opt
- IntervalSign_Opt
- Limit_Opt
- Nulls_Opt
- OrderingSpecification_Opt
- ParamList_Opt
- Params_Opt

- PassingClauseName_Opt
- PassingClause_Opt
- Percent_Opt
- Pivot_Opt
- PrivilegeColumnList_Opt
- ProcedureParams_Opt
- SelectLimit_Opt
- SetQuantifier_Opt
- ToDay_Opt
- ToHour_Opt
- ToMinute_Opt
- ToMonth_Opt
- ToSecond_Opt
- ToYear_Opt
- TopLevel
- Unique_Opt
- WhereSearchCond_Opt
- WithGrantOption_Opt

RestrictedIdentifier:



RestrictedIdentifier

```

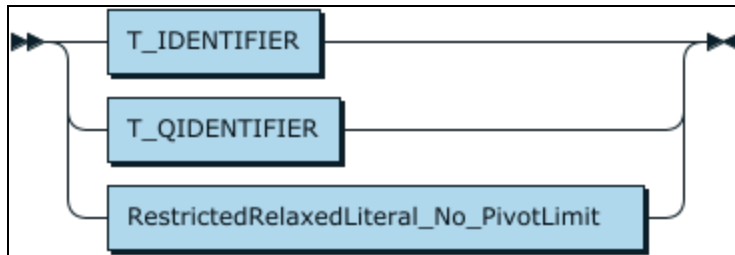
::= RestrictedIdentifier_No_Limit
| T_LIMIT

```

referenced by:

- [PassingClauseName](#)
- [RestrictedColumnName](#)

RestrictedIdentifier_No_PivotLimit:



RestrictedIdentifier_No_PivotLimit

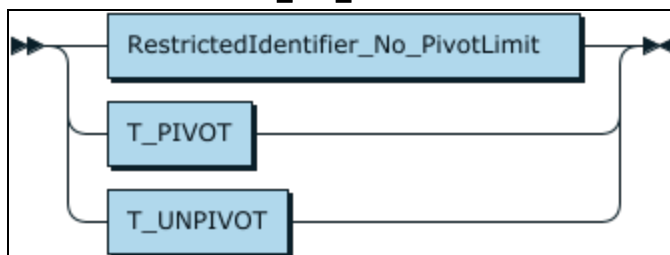
```

::= T_IDENTIFIER
| T_QIDENTIFIER
| RestrictedRelaxedLiteral_No_PivotLimit
  
```

referenced by:

- [Identifier_No_PivotLimit](#)
- [RestrictedIdentifier_No_Limit](#)

RestrictedIdentifier_No_Limit:



RestrictedIdentifier_No_Limit

```

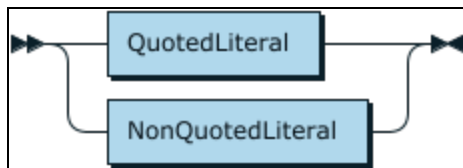
::= RestrictedIdentifier_No_PivotLimit
| T_PIVOT
  
```

| T_UNPIVOT

referenced by:

- Identifier_No_Limit
- RestrictedIdentifier

IdentLiteral:



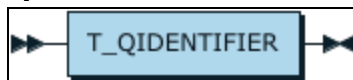
IdentLiteral

::= QuotedLiteral
| NonQuotedLiteral

referenced by:

- DateTimeTSGuidValueVT
- OuterJoinVT
- PredVT
- ScalarOrAggrFn

QuotedLiteral:



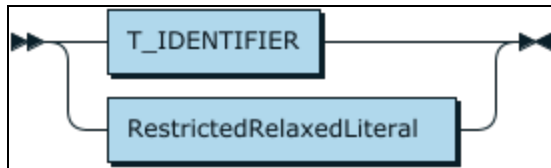
QuotedLiteral

::= T_QIDENTIFIER

referenced by:

- [DataType](#)
- [IdentLiteral](#)

NonQuotedLiteral:



NonQuotedLiteral

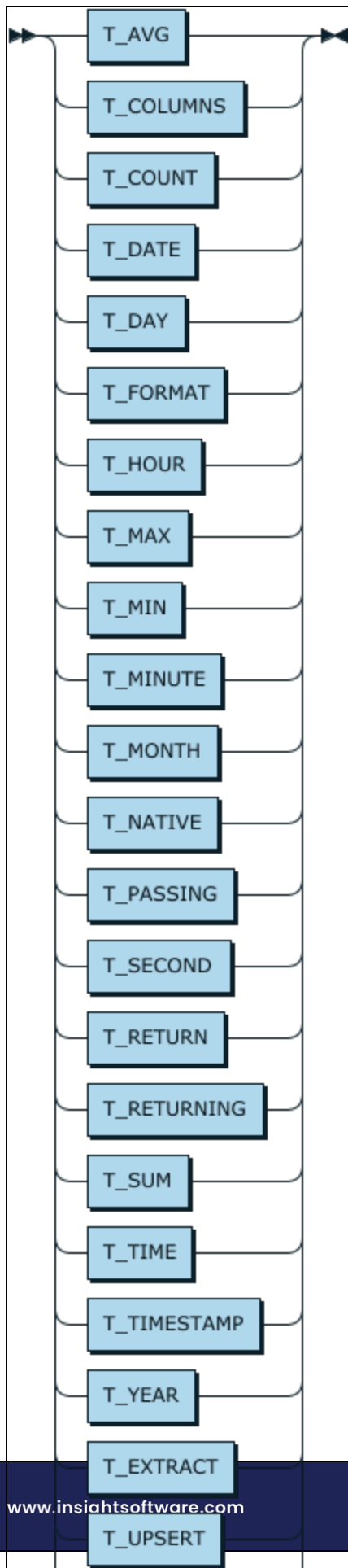
```

::= T_IDENTIFIER
   | RestrictedRelaxedLiteral
  
```

referenced by:

- [IdentLiteral](#)
- [NonQuotedDataTypeLiteral](#)

RestrictedRelaxedLiteral_No_PivotLimit:



RestrictedRelaxedLiteral_No_PivotLimit

```

::= T_AVG
    | T_COLUMNS
    | T_COUNT
    | T_DATE
    | T_DAY
    | T_FORMAT
    | T_HOUR
    | T_MAX
    | T_MIN
    | T_MINUTE
    | T_MONTH
    | T_NATIVE
    | T_PASSING
    | T_SECOND
    | T_RETURN
    | T_RETURNING
    | T_SUM
    | T_TIME
    | T_TIMESTAMP
    | T_YEAR
    | T_EXTRACT
    | T_UPSERT
    | T_INCLUDE
    | T_EXCLUDE
    | T_NULLS
    | T_CHECK
    | T_END
    | T_PERCENT
    | T_WEEK
    | T_QUARTER
    | T_DAYOFWEEK

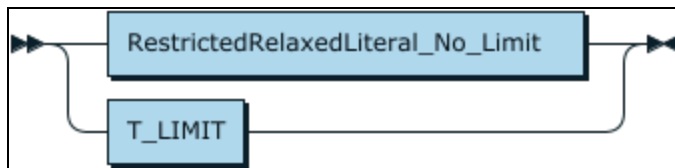
```

| T_POSITION

referenced by:

- RestrictedIdentifier_No_PivotLimit
- RestrictedRelaxedLiteral_No_Limit

RestrictedRelaxedLiteral:



RestrictedRelaxedLiteral

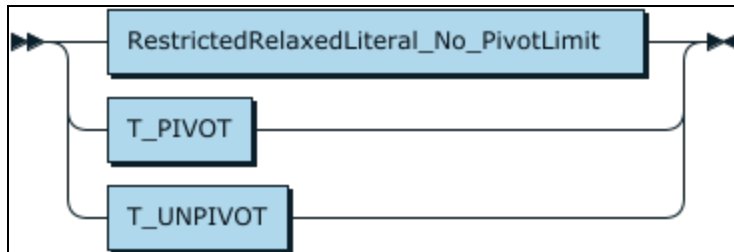
::= RestrictedRelaxedLiteral_No_Limit

| T_LIMIT

referenced by:

- NonQuotedLiteral

RestrictedRelaxedLiteral_No_Limit:



RestrictedRelaxedLiteral_No_Limit

::= RestrictedRelaxedLiteral_No_PivotLimit

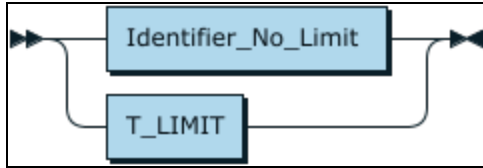
| T_PIVOT

| T_UNPIVOT

referenced by:

- [RestrictedRelaxedLiteral](#)

Identifier:



Identifier

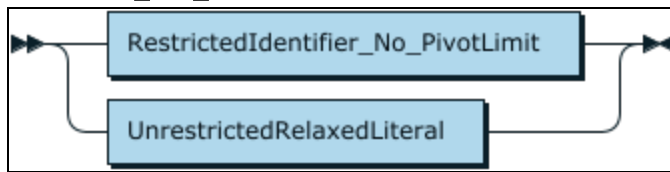
```

::= Identifier_No_Limit
| T_LIMIT
  
```

referenced by:

- [AuthorizationIdentifier](#)
- [ColumnReference](#)
- [CreateIndexStmt](#)
- [DataTypeAttributeValue](#)
- [DropIndexStmt](#)
- [Index_Type_Opt](#)
- [MeasureColumnList](#)
- [OrderColumn](#)
- [ParameterName](#)
- [QualifiedIdentifier](#)
- [QualifiedName](#)
- [QualifiedProcedureName](#)
- [SelectSubListItem](#)
- [UnpivotColumnList](#)

Identifier_No_PivotLimit:



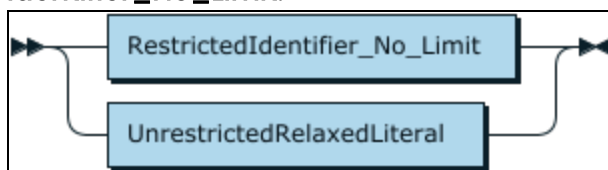
```

Identifier_No_PivotLimit
  ::= RestrictedIdentifier_No_PivotLimit
  | UnrestrictedRelaxedLiteral
  
```

referenced by:

- [CorrelationName](#)

Identifier_No_Limit:



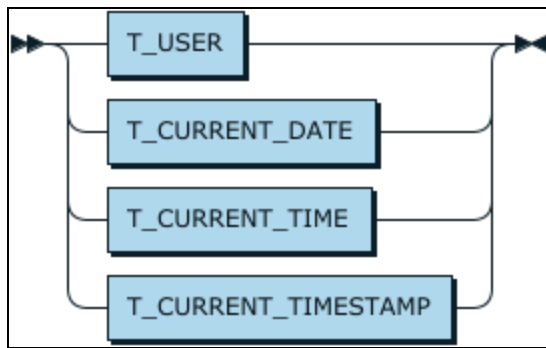
```

Identifier_No_Limit
  ::= RestrictedIdentifier_No_Limit
  | UnrestrictedRelaxedLiteral
  
```

referenced by:

- [ColumnName_No_Limit](#)
- [Identifier](#)

UnrestrictedRelaxedLiteral:



UnrestrictedRelaxedLiteral

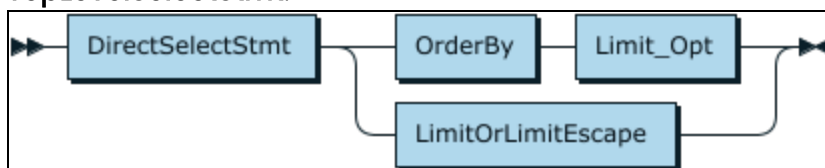
```

::= T_USER
   | T_CURRENT_DATE
   | T_CURRENT_TIME
   | T_CURRENT_TIMESTAMP
  
```

referenced by:

- Identifier_No_Limit
- Identifier_No_PivotLimit
- NonQuotedDataTypeLiteral

TopLevelSelectStmt:



TopLevelSelectStmt

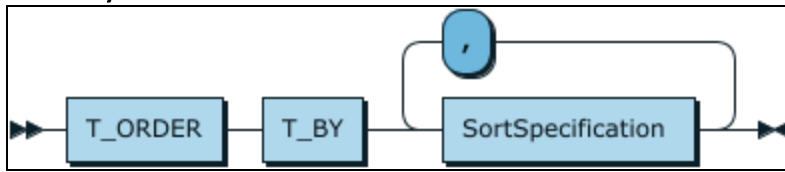
```

::= DirectSelectStmt ( OrderByLimit_Opt | LimitOrLimitEscape )
  
```

referenced by:

- ExecStmt

OrderBy:

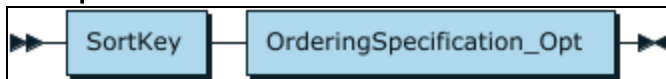


```
OrderBy ::= T_ORDER T_BY SortSpecification ( ',' SortSpecification )*
```

referenced by:

- [InsertList](#)
- [SubQuery](#)
- [TopLevelSelectStmt](#)

SortSpecification:



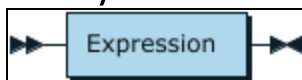
SortSpecification

```
::= SortKey OrderingSpecification_Opt
```

referenced by:

- [OrderBy](#)

SortKey:

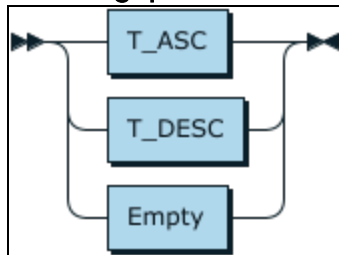


```
SortKey ::= Expression
```

referenced by:

- SortSpecification

OrderingSpecification_Opt:



OrderingSpecification_Opt

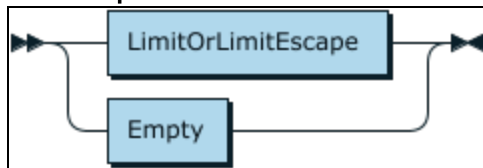
```

::= T_ASC
| T_DESC
| Empty
  
```

referenced by:

- OrderColumn
- SortSpecification

Limit_Opt:



Limit_Opt

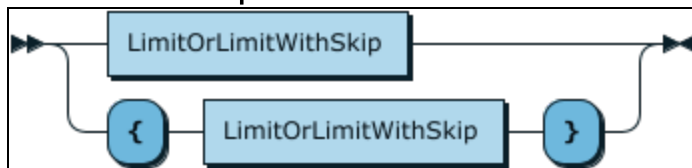
```

::= LimitOrLimitEscape
| Empty
  
```

referenced by:

- InsertList
- NonJoinQueryExpression
- NonJoinQueryPrimaryLimitOpt
- SubQuery
- TopLevelSelectStmt

LimitOrLimitEscape:



LimitOrLimitEscape

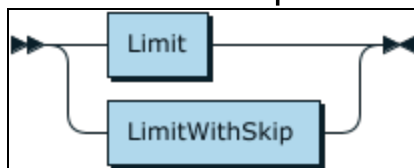
```

::= LimitOrLimitWithSkip
| '{ LimitOrLimitWithSkip }'
  
```

referenced by:

- Limit_Opt
- SubQuery
- TopLevelSelectStmt

LimitOrLimitWithSkip:



LimitOrLimitWithSkip

```

::= Limit
| LimitWithSkip
  
```

referenced by:

- [LimitOrLimitEscape](#)

LimitWithSkip:



LimitWithSkip

`::= T_LIMIT UnsignedValueSpecification ',' UnsignedValueSpecification`

referenced by:

- [LimitOrLimitWithSkip](#)

Limit:

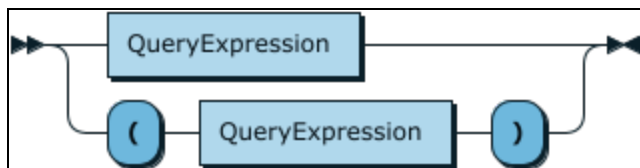


Limit ::= T_LIMIT UnsignedValueSpecification

referenced by:

- [LimitOrLimitWithSkip](#)

DirectSelectStmt:



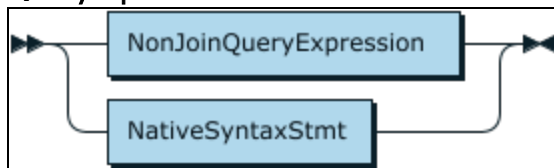
DirectSelectStmt

```
 ::= QueryExpression
    | '(' QueryExpression ')'
```

referenced by:

- CreateTableStmt
- ExecStmt
- NonJoinQueryExpression
- TopLevelSelectStmt

QueryExpression:

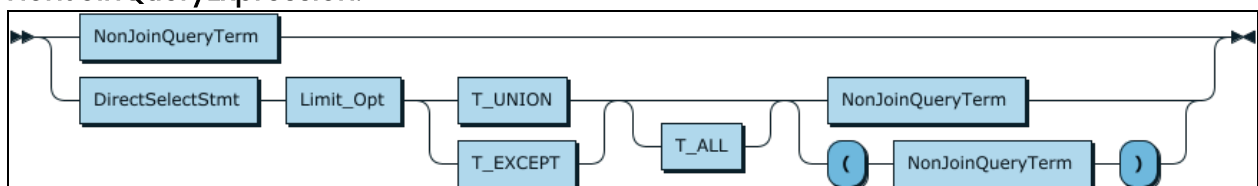


```
QueryExpression
 ::= NonJoinQueryExpression
    | NativeSyntaxStmt
```

referenced by:

- DirectSelectStmt
- InsertList
- SubQuery

NonJoinQueryExpression:



NonJoinQueryExpression

```

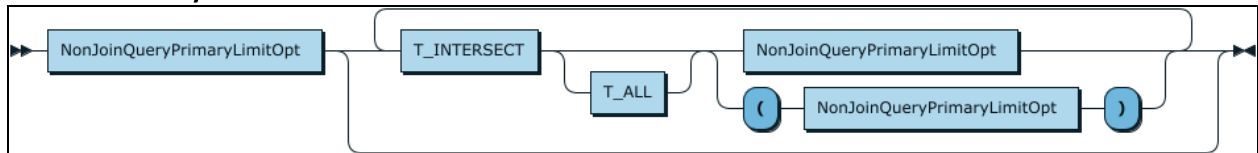
::= NonJoinQueryTerm
    | DirectSelectStmt Limit_Opt ( T_UNION | T_EXCEPT ) T_ALL? ( NonJoinQueryTerm | '('
        NonJoinQueryTerm ')' )

```

referenced by:

- [QueryExpression](#)

NonJoinQueryTerm:



NonJoinQueryTerm

```

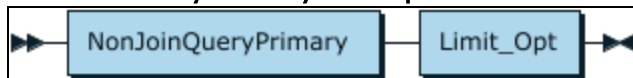
::= NonJoinQueryPrimaryLimitOpt ( T_INTERSECT T_ALL? ( NonJoinQueryPrimaryLimitOpt | '('
        NonJoinQueryPrimaryLimitOpt ')' ) ) *

```

referenced by:

- [NonJoinQueryExpression](#)

NonJoinQueryPrimaryLimitOpt:



NonJoinQueryPrimaryLimitOpt

```

::= NonJoinQueryPrimary Limit_Opt

```

referenced by:

- [NonJoinQueryTerm](#)

NonJoinQueryPrimary:

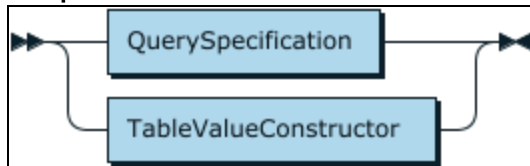


```
NonJoinQueryPrimary
  ::= SimpleTable
```

referenced by:

- NonJoinQueryPrimaryLimitOpt

SimpleTable:

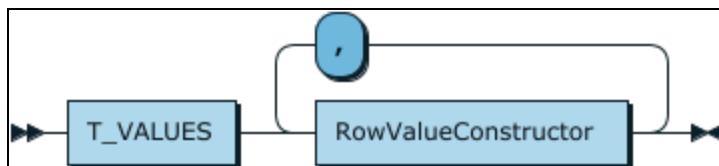


```
SimpleTable
  ::= QuerySpecification
  | TableValueConstructor
```

referenced by:

- NonJoinQueryPrimary

TableValueConstructor:

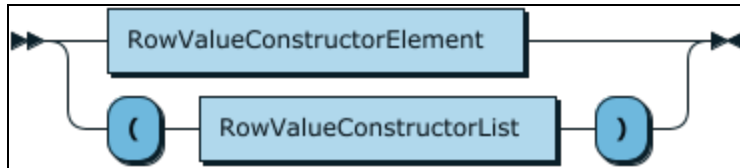


```
TableValueConstructor
  ::= T_VALUES RowValueConstructor (',' RowValueConstructor)*
```

referenced by:

- SimpleTable

RowValueConstructor:



RowValueConstructor

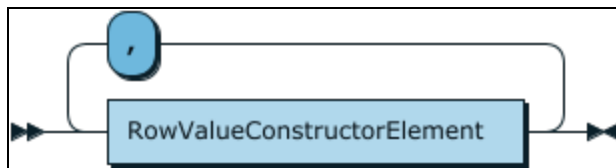
```

::= RowValueConstructorElement
    | '(' RowValueConstructorList ')'
    
```

referenced by:

- BetweenPredicate
- ComparisonPredicate
- InPredicate
- NullPredicate
- QuantifiedComparisonPredicate
- TableValueConstructor

RowValueConstructorList:



RowValueConstructorList

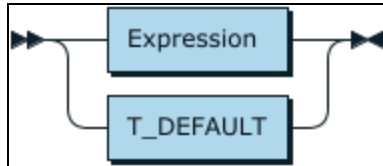
```

::= RowValueConstructorElement (',' RowValueConstructorElement)*
    
```

referenced by:

- MergeWhenNotMatchedClause
- RowValueConstructor

RowValueConstructorElement:



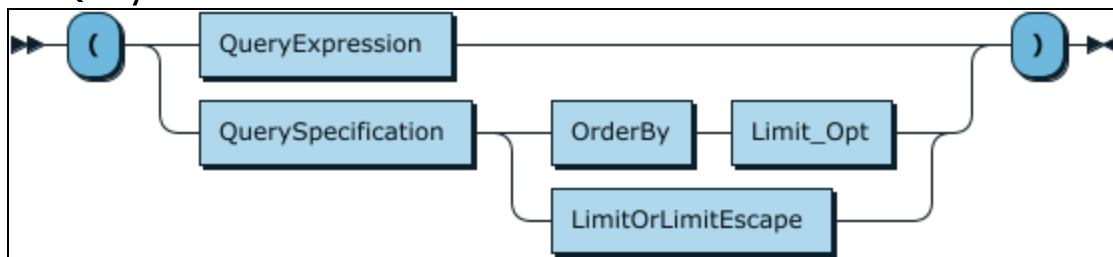
RowValueConstructorElement

::= Expression
 | T_DEFAULT

referenced by:

- RowValueConstructor
- RowValueConstructorList

SubQuery:



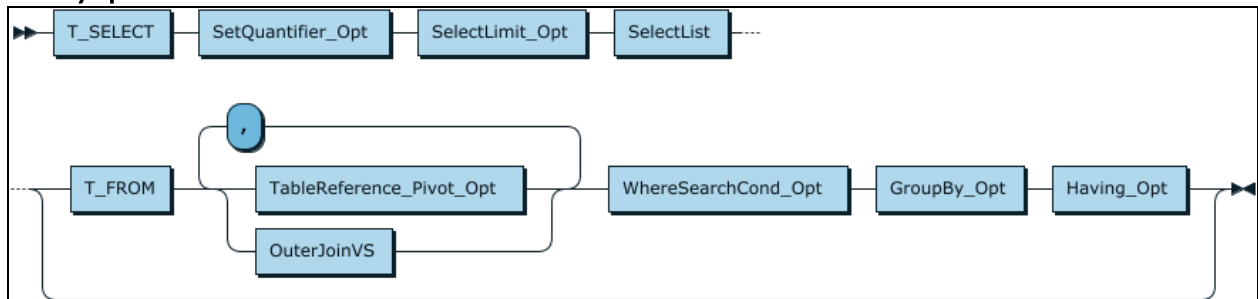
SubQuery ::= '(' (QueryExpression | QuerySpecification (OrderBy Limit_Opt | LimitOrLimitEscape)))'

referenced by:

- ExistsPredicate
- InPredicateValue

- QuantifiedComparisonPredicate
- TablePrimary
- ValueExpressionPrimary

QuerySpecification:



QuerySpecification

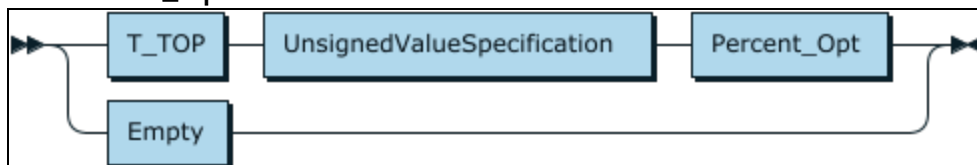
```

::= T_SELECT SetQuantifier_Opt SelectLimit_Opt SelectList ( T_FROM ( TableReference_Pivot_Opt |
    OuterJoinVS ) ( ',' ( TableReference_Pivot_Opt | OuterJoinVS ) ) *
    WhereSearchCond_Opt GroupBy_Opt Having_Opt ) ?
    
```

referenced by:

- CreateViewStmt
- InsertList
- SimpleTable
- SubQuery

SelectLimit_Opt:



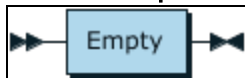
SelectLimit_Opt

```
::= T_TOP UnsignedValueSpecification Percent_Opt
| Empty
```

referenced by:

- [QuerySpecification](#)

Percent_Opt:

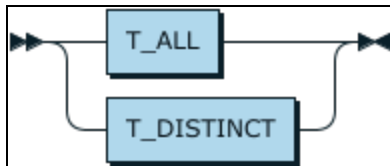


```
Percent_Opt
::= Empty
```

referenced by:

- [SelectLimit_Opt](#)

SetQuantifier:

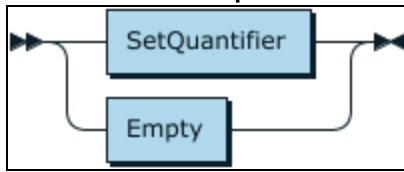


```
SetQuantifier
::= T_ALL
| T_DISTINCT
```

referenced by:

- [ScalarOrAggrFn](#)
- [SetQuantifier_Opt](#)

SetQuantifier_Opt:



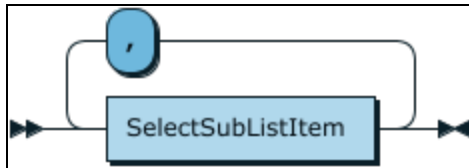
```

SetQuantifier_Opt
  ::= SetQuantifier
  | Empty
  
```

referenced by:

- QuerySpecification
- SetFunction

SelectList:



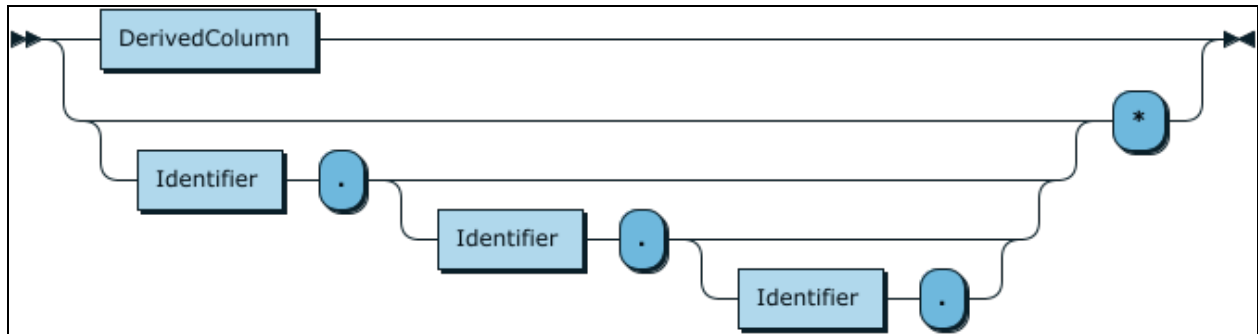
```

SelectList
  ::= SelectSubListItem (',' SelectSubListItem)*
  
```

referenced by:

- QuerySpecification

SelectSubListItem:



SelectSubListItem

::= DerivedColumn

| (Identifier '!' (Identifier '!' (Identifier '!')?)?)? '*'

referenced by:

- SelectList

DerivedColumn:



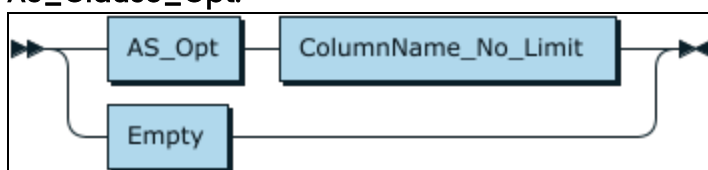
DerivedColumn

::= (Expression | BooleanValueExpression) AS_Clause_Opt

referenced by:

- SelectSubListItem

AS_Clause_Opt:



AS_Clause_Opt

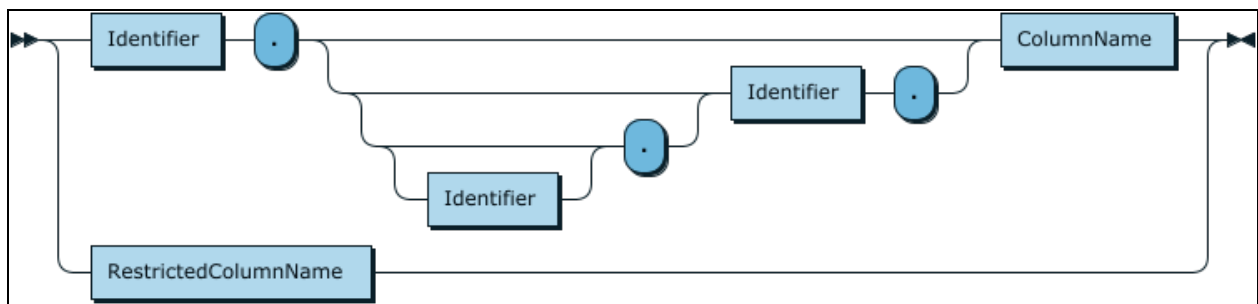
::= AS_Opt ColumnName_No_Limit

| Empty

referenced by:

- [DerivedColumn](#)

ColumnReference:



ColumnReference

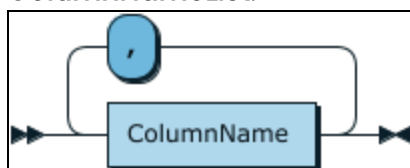
::= Identifier '.' ((Identifier? '.')? Identifier '.')? ColumnName

| RestrictedColumnName

referenced by:

- [ColumnReferencelist](#)
- [ColumnReferences](#)
- [GroupColumnList](#)
- [ValueExpressionPrimary](#)

ColumnNameList:



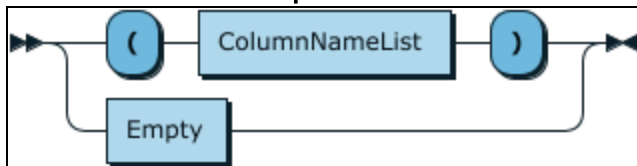
ColumnNameList

$::= \text{ColumnName} (',' \text{ColumnName})^*$

referenced by:

- ColumnNameList_Opt
- CreateViewStmt
- DerivedColumnList_Opt
- InsertList
- PrivilegeColumnList_Opt
- ReturnEscapeStmt
- TableConstraintDefinition

ColumnNameList_Opt:



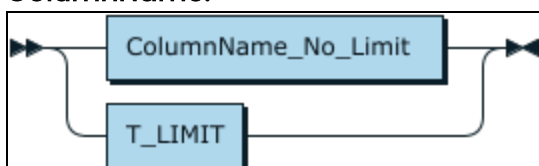
ColumnNameList_Opt

$::= '(' \text{ColumnNameList} ')'$
 $| \text{Empty}$

referenced by:

- CreateTableStmt

ColumnName:



ColumnName

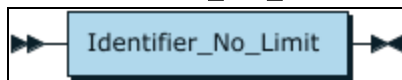
::= ColumnName_No_Limit

| T_LIMIT

referenced by:

- ColumnDefinition
- ColumnNameList
- ColumnReference
- SetClause

ColumnName_No_Limit:



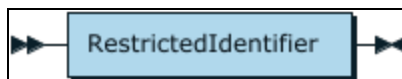
ColumnName_No_Limit

::= Identifier_No_Limit

referenced by:

- AS_Clause_Opt
- ColumnName

RestrictedColumnName:



RestrictedColumnName

::= RestrictedIdentifier

referenced by:

- [ColumnReference](#)

TableReference_Pivot_Opt:

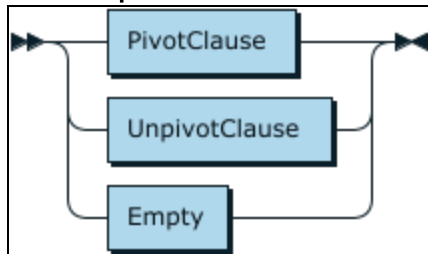


```
TableReference_Pivot_Opt
::= TableReference Pivot_Opt
```

referenced by:

- [JoinedTable](#)
- [QualifiedJoin](#)
- [QuerySpecification](#)

Pivot_Opt:

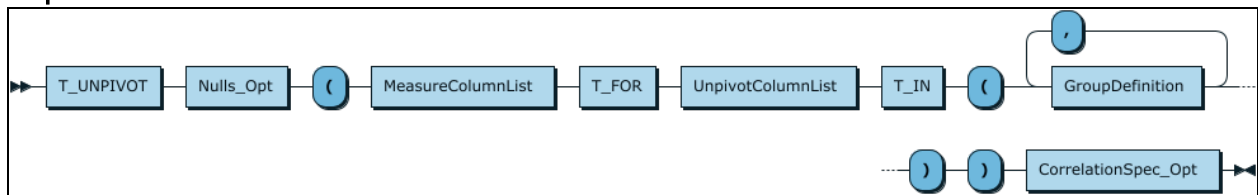


```
Pivot_Opt
::= PivotClause
| UnpivotClause
| Empty
```

referenced by:

- [TableReference_Pivot_Opt](#)

UnpivotClause:



UnpivotClause

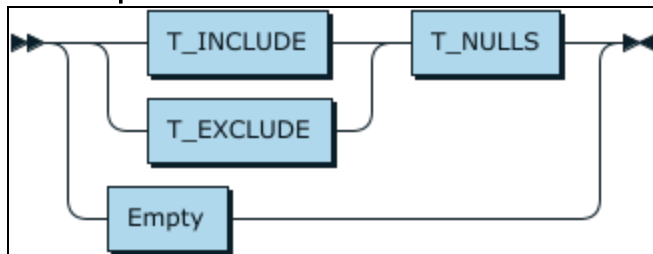
```

::= T_UNPIVOT Nulls_Opt '(' MeasureColumnList T_FOR UnpivotColumnList T_IN '(' GroupDefinition ( ';'
      GroupDefinition ) * ')' ) CorrelationSpec_Opt
  
```

referenced by:

- [Pivot_Opt](#)

Nulls_Opt:



Nulls_Opt

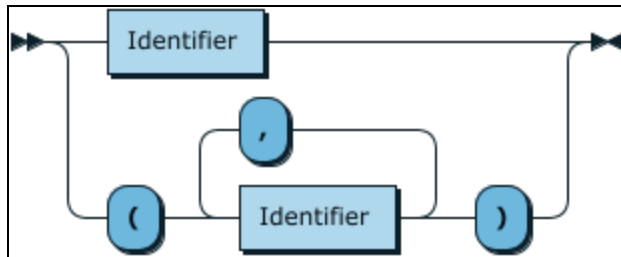
```

::= ( T_INCLUDE | T_EXCLUDE ) T_NULLS
    | Empty
  
```

referenced by:

- [UnpivotClause](#)

MeasureColumnList:



MeasureColumnList

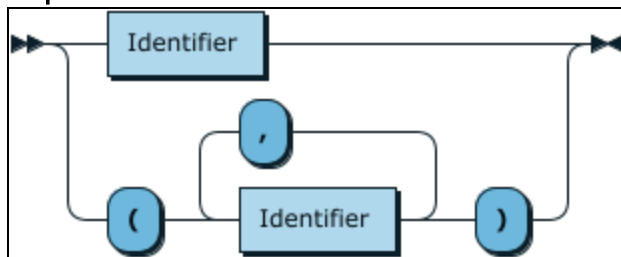
::= Identifier

| '(' Identifier (',' Identifier)* ')'

referenced by:

- UnpivotClause

UnpivotColumnList:



UnpivotColumnList

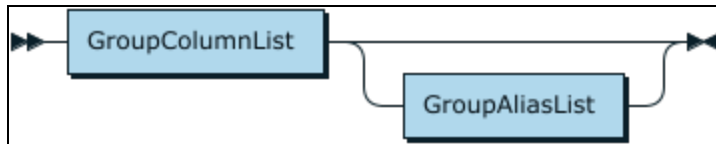
::= Identifier

| '(' Identifier (',' Identifier)* ')'

referenced by:

- UnpivotClause

GroupDefinition:



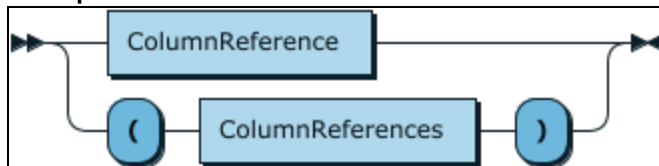
GroupDefinition

::= GroupColumnList GroupAliasList?

referenced by:

- UnpivotClause

GroupColumnList:



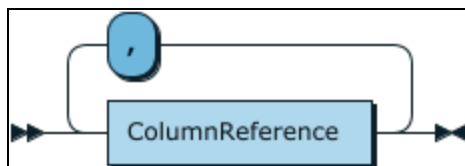
GroupColumnList

::= ColumnReference
| '(' ColumnReferences ')'

referenced by:

- GroupDefinition

ColumnReferences:



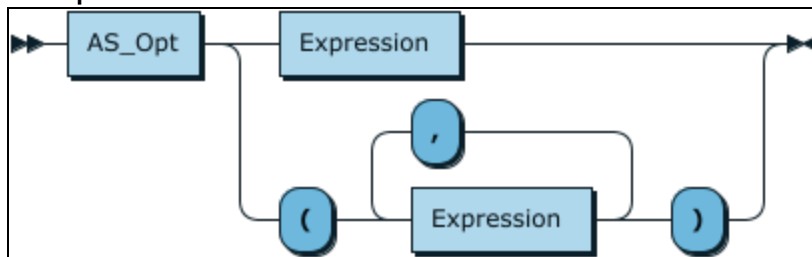
ColumnReferences

```
 ::= ColumnReference (',' ColumnReference)*
```

referenced by:

- ColumnReferenceList
- GroupColumnList

GroupAliasList:



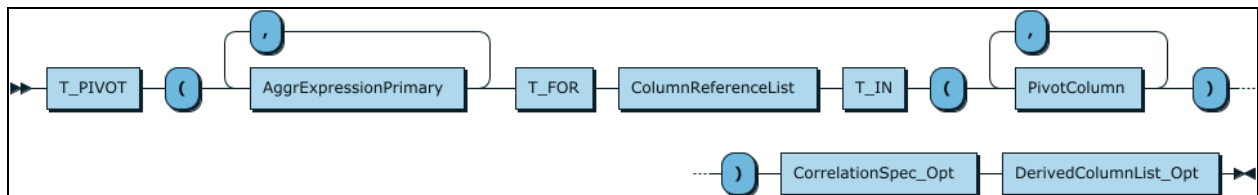
GroupAliasList

```
 ::= AS_Opt ( Expression | '(' Expression (',' Expression)* ')' )
```

referenced by:

- GroupDefinition

PivotClause:



PivotClause

```
 ::= T_PIVOT '(' AggrExpressionPrimary (',' AggrExpressionPrimary)* T_FOR ColumnReferenceList T_IN '
    (' PivotColumn (',' PivotColumn)* ')' ')' CorrelationSpec_Opt DerivedColumnList_
    Opt
```

referenced by:

- [Pivot_Opt](#)

AggrExpressionPrimary:



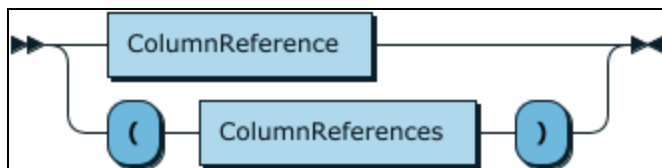
AggrExpressionPrimary

::= Expression CorrelationSpec_Opt

referenced by:

- [PivotClause](#)

ColumnReferenceList:



ColumnReferenceList

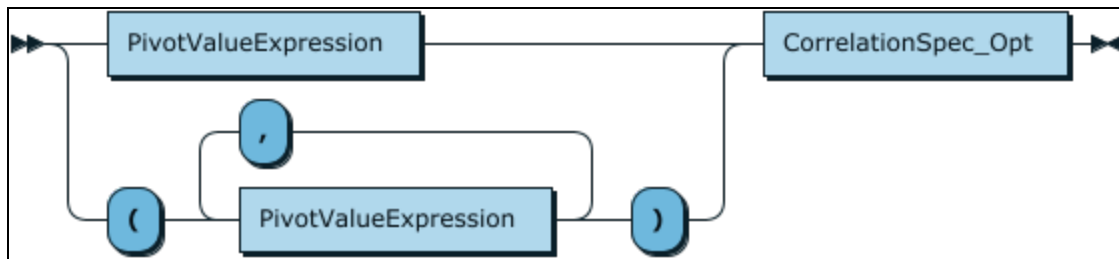
::= ColumnReference

| '(' ColumnReferences ')'

referenced by:

- [PivotClause](#)

PivotColumn:



PivotColumn

$::= (\text{PivotValueExpression} \mid '(\text{PivotValueExpression} (',' \text{PivotValueExpression})^*)') \text{CorrelationSpec_Opt}$

referenced by:

- **PivotClause**

PivotValueExpression:



PivotValueExpression

$::= \text{Expression}$

referenced by:

- **PivotColumn**

TableReference:



TableReference

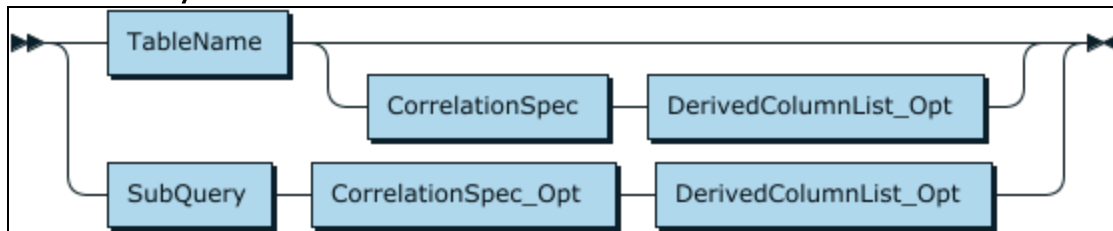
$::= \text{TablePrimary}$

| JoinedTable

referenced by:

- TableReference_Pivot_Opt

TablePrimary:



TablePrimary

```

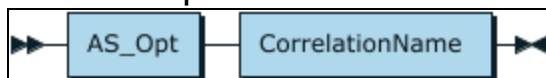
::= TableName ( CorrelationSpec DerivedColumnList_Opt )?
| SubQuery CorrelationSpec_Opt DerivedColumnList_Opt

```

referenced by:

- MergeStmt
- TableReference

CorrelationSpec:



CorrelationSpec

```

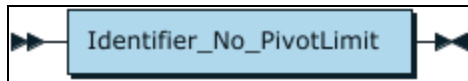
::= AS_Opt CorrelationName

```

referenced by:

- CorrelationSpec_Opt
- TablePrimary

CorrelationName:



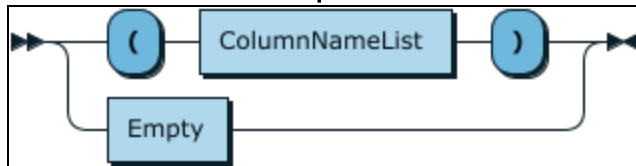
CorrelationName

::= Identifier_No_PivotLimit

referenced by:

- [CorrelationSpec](#)

DerivedColumnList_Opt:



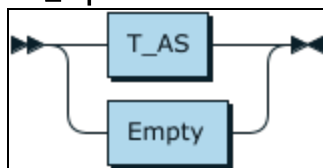
DerivedColumnList_Opt

::= '(' ColumnNameList ')'
| Empty

referenced by:

- [MergeWhenNotMatchedClause](#)
- [PivotClause](#)
- [TablePrimary](#)

AS_Opt:

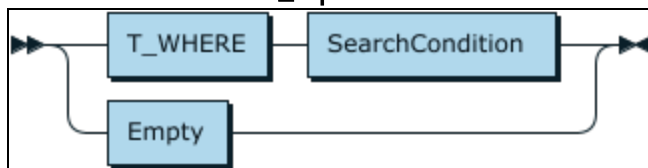


```
AS_Opt ::= T_AS
      | Empty
```

referenced by:

- AS_Clause_Opt
- CorrelationSpec
- GroupAliasList

WhereSearchCond_Opt:

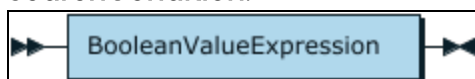


```
WhereSearchCond_Opt
  ::= T_WHERE SearchCondition
     | Empty
```

referenced by:

- DeleteStmtSearched
- QuerySpecification
- UpdateStmtSearched
- UpsertStmtSearched

SearchCondition:



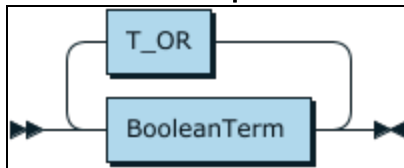
SearchCondition

::= BooleanValueExpression

referenced by:

- BooleanPrimary
- Having_Opt
- MergeStmt
- QualifiedJoin
- SearchedWhenClause
- WhereSearchCond_Opt

BooleanValueExpression:



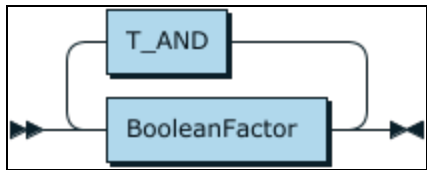
BooleanValueExpression

::= BooleanTerm (T_OR BooleanTerm)*

referenced by:

- CaseAbbreviation
- DerivedColumn
- GroupBy_Opt
- SearchCondition

BooleanTerm:



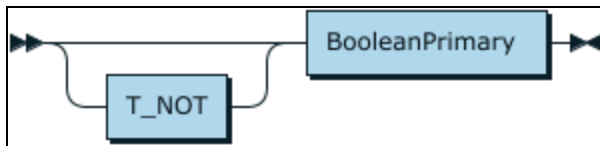
BooleanTerm

$::= \text{BooleanFactor (T_AND BooleanFactor)}^*$

referenced by:

- [BooleanValueExpression](#)
- [Result](#)

BooleanFactor:



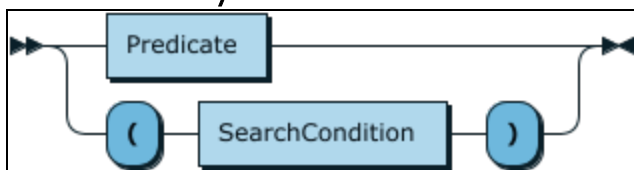
BooleanFactor

$::= \text{T_NOT? BooleanPrimary}$

referenced by:

- [BooleanTerm](#)

BooleanPrimary:



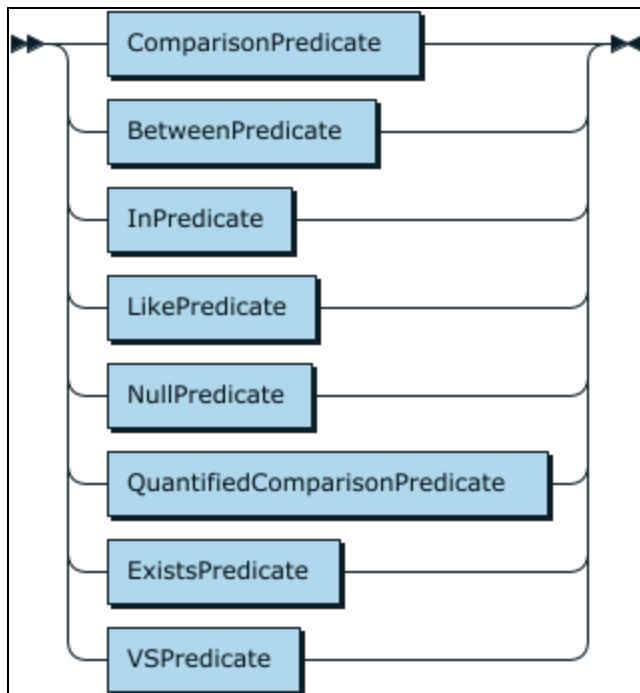
BooleanPrimary

```
 ::= Predicate
    | '(' SearchCondition ')'
```

referenced by:

- BooleanFactor

Predicate:



Predicate

```
 ::= ComparisonPredicate
    | BetweenPredicate
    | InPredicate
    | LikePredicate
    | NullPredicate
    | QuantifiedComparisonPredicate
    | ExistsPredicate
    | VSPredicate
```

referenced by:

- [BooleanPrimary](#)

ComparisonPredicate:



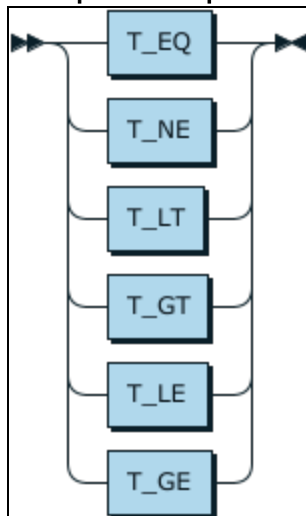
ComparisonPredicate

::= RowValueConstructor ComparisonOp RowValueConstructor

referenced by:

- [Predicate](#)

ComparisonOp:



ComparisonOp

::= T_EQ

| T_NE

| T_LT

| T_GT

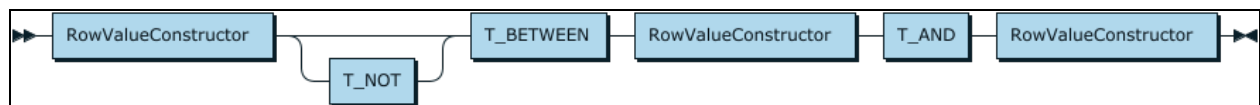
| T_LE

| T_GE

referenced by:

- ComparisonPredicate
- QuantifiedComparisonPredicate

BetweenPredicate:



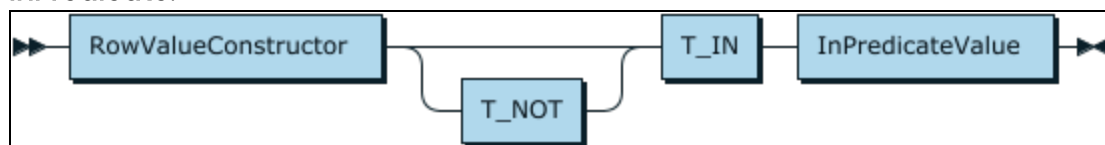
BetweenPredicate

::= RowValueConstructor T_NOT? T_BETWEEN RowValueConstructor T_AND RowValueConstructor

referenced by:

- Predicate

InPredicate:



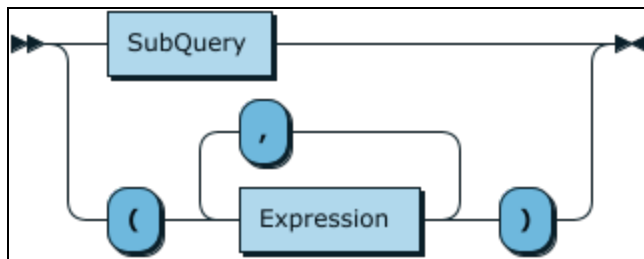
InPredicate

::= RowValueConstructor T_NOT? T_IN InPredicateValue

referenced by:

- Predicate

InPredicateValue:



InPredicateValue

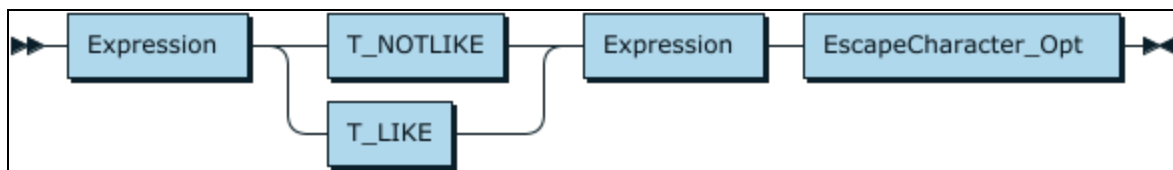
::= SubQuery

| '(' Expression (';' Expression)* ')'

referenced by:

- InPredicate

LikePredicate:



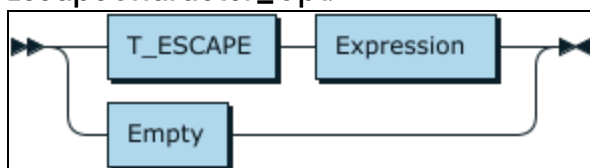
LikePredicate

::= Expression (T_NOTLIKE | T_LIKE) Expression EscapeCharacter_Opt

referenced by:

- Predicate

EscapeCharacter_Opt:



EscapeCharacter_Opt

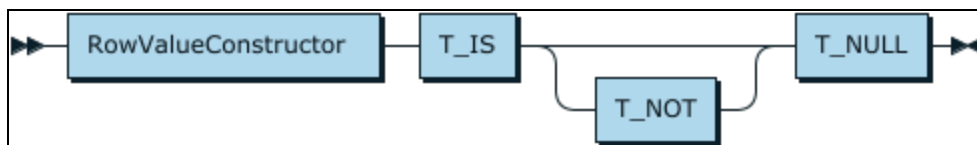
::= T_ESCAPE Expression

| Empty

referenced by:

- LikePredicate

NullPredicate:



NullPredicate

::= RowValueConstructor T_IS T_NOT? T_NULL

referenced by:

- Predicate

QuantifiedComparisonPredicate:



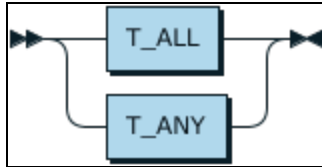
QuantifiedComparisonPredicate

::= RowValueConstructor ComparisonOp Quantifier SubQuery

referenced by:

- Predicate

Quantifier:



Quantifier

::= T_ALL
| T_ANY

referenced by:

- QuantifiedComparisonPredicate

ExistsPredicate:



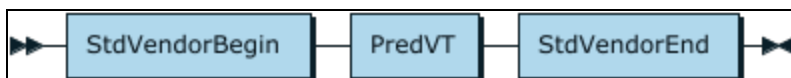
ExistsPredicate

::= T_EXISTS SubQuery

referenced by:

- Predicate

VSPredicate:



VSPredicate

::= StdVendorBegin PredVT StdVendorEnd

referenced by:

- Predicate

StdVendorBegin:



StdVendorBegin

```
::= T_SVBEGIN T_IDENTIFIER '(' T_IDENTIFIER ')' ',' T_IDENTIFIER '(' T_IDENTIFIER ')'
```

referenced by:

- OuterJoinVS
- ProcedureStmt
- VSPredicate
- ValueVS

StdVendorEnd:



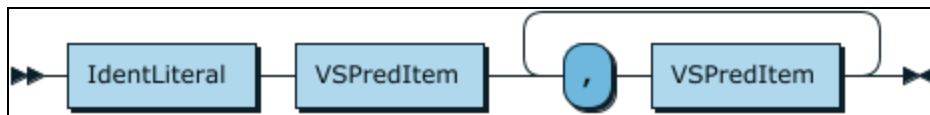
StdVendorEnd

```
::= T_SVEND
```

referenced by:

- OuterJoinVS
- ProcedureStmt
- VSPredicate
- ValueVS

PredVT:

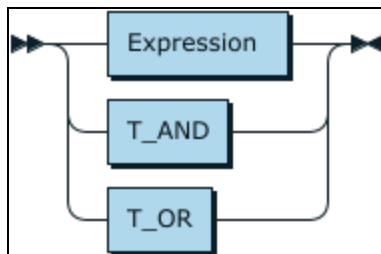


```
PredVT ::= IdentLiteral VSPredItem (',' VSPredItem )+
```

referenced by:

- `VSPredicate`

VSPredItem:

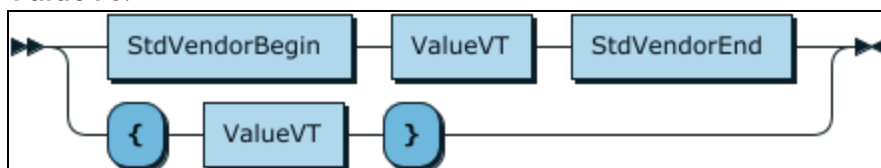


```
VSPredItem  
    ::= Expression  
    | T_AND  
    | T_OR
```

referenced by:

- `PredVT`

ValueVS:

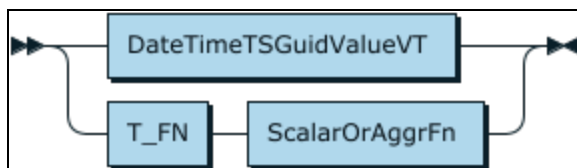


```
ValueVS ::= StdVendorBegin ValueVT StdVendorEnd
        | '{ ValueVT }'
```

referenced by:

- ValueExpressionPrimary

ValueVT:

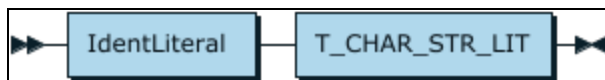


```
ValueVT ::= DateTimeTSGuidValueVT
        | T_FN ScalarOrAggrFn
```

referenced by:

- ValueVS

DateTimeTSGuidValueVT:



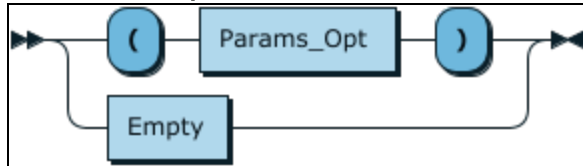
```
DateTimeTSGuidValueVT
    ::= IdentLiteral T_CHAR_STR_LIT
```

referenced by:

- ValueVT

- ValueExpressionPrimary
- ValueVT

ParamList_Opt:



ParamList_Opt

```

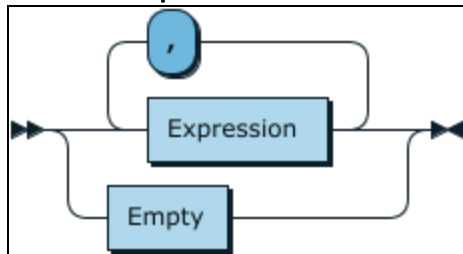
::= '(' Params_Opt ')'
| Empty

```

referenced by:

- ScalarOrAggrFn

Params_Opt:



Params_Opt

```

::= Expression (',' Expression)*
| Empty

```

referenced by:

- ParamList_Opt
- ScalarOrAggrFn

ConvertParams:



ConvertParams

::= Expression ',' DataType

referenced by:

- [ScalarOrAggrFn](#)

CastParams:



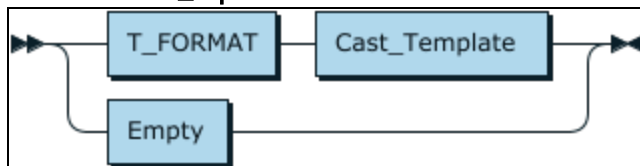
CastParams

::= Expression T_AS DataType CastFormat_Opt

referenced by:

- [ScalarOrAggrFn](#)

CastFormat_Opt:



CastFormat_Opt

::= T_FORMAT Cast_Template
| Empty

referenced by:

- CastParams

Cast_Template:



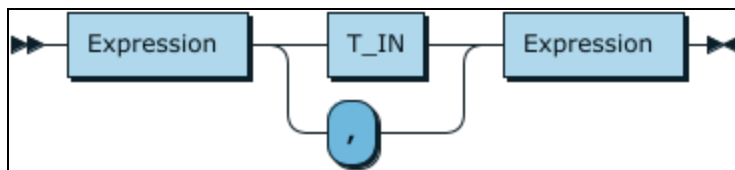
Cast_Template

::= T_CHAR_STR_LIT

referenced by:

- CastFormat_Opt

PositionParams:



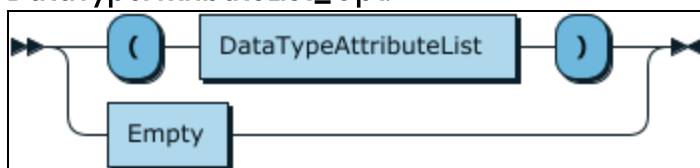
PositionParams

::= Expression (T_IN | ',') Expression

referenced by:

- ScalarOrAggrFn

DataTypeAttributeList_Opt:

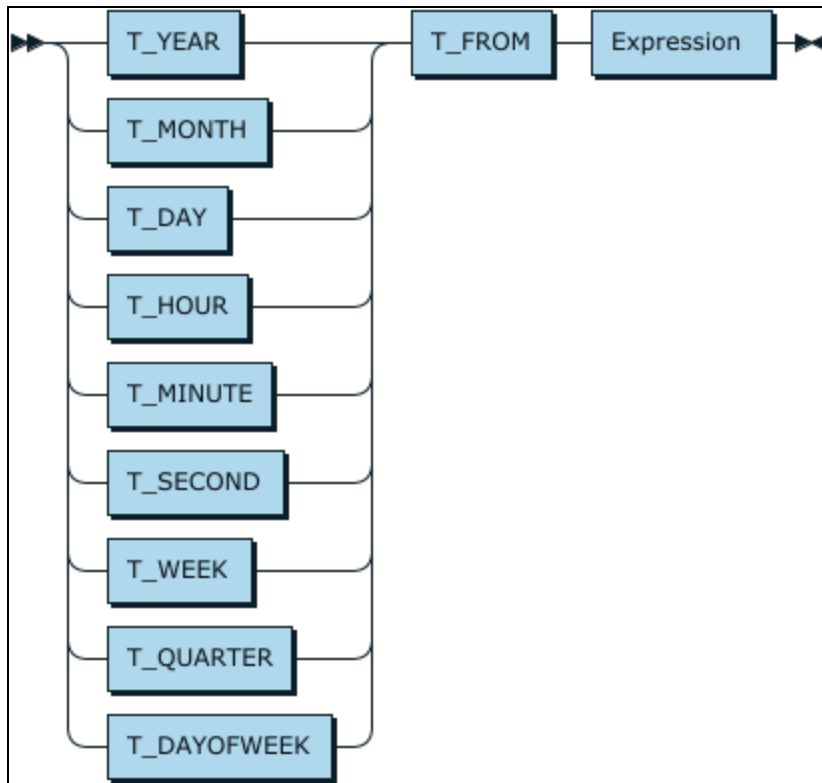


DataTypeAttributeList_Opt

```
 ::= ('( DataTypeAttributeList ')  
      | Empty
```

no references

ExtractParams:



ExtractParams

```
 ::= ( T_YEAR | T_MONTH | T_DAY | T_HOUR | T_MINUTE | T_SECOND | T_WEEK | T_QUARTER | T_  
       DAYOFWEEK ) T_FROM Expression
```

referenced by:

- [ScalarOrAggrFn](#)

OuterJoinVT:



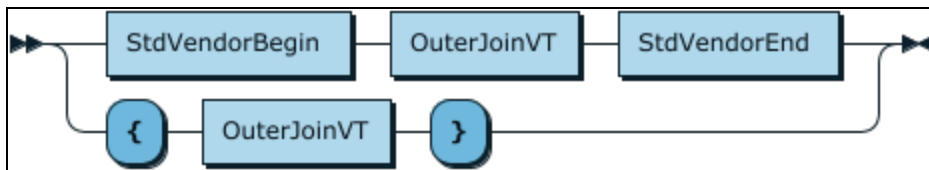
OuterJoinVT

::= IdentLiteral JoinedTable

referenced by:

- OuterJoinVS

OuterJoinVS:



OuterJoinVS

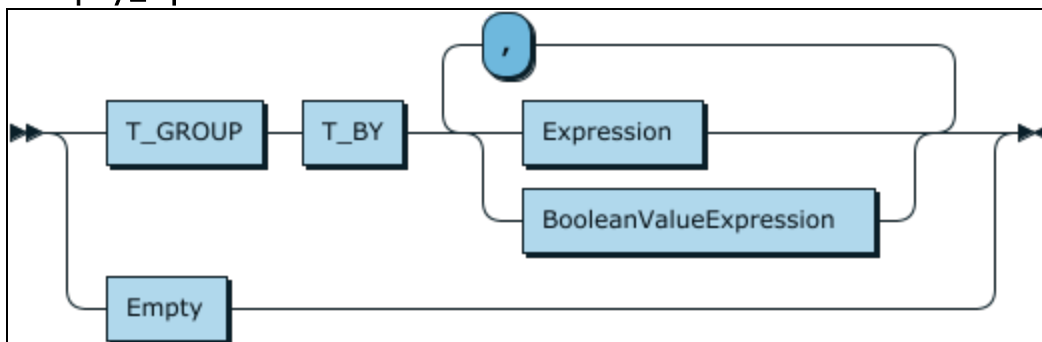
::= StdVendorBegin OuterJoinVT StdVendorEnd

| '{ OuterJoinVT }'

referenced by:

- QuerySpecification

GroupBy_Opt:



GroupBy_Opt

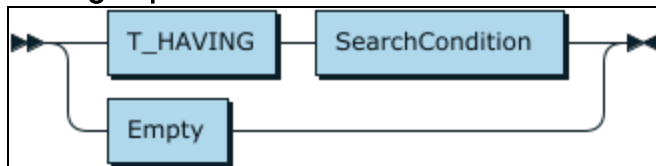
::= T_GROUP T_BY (Expression | BooleanValueExpression) (',' (Expression | BooleanValueExpression)) *

| Empty

referenced by:

- QuerySpecification

Having_Opt:



Having_Opt

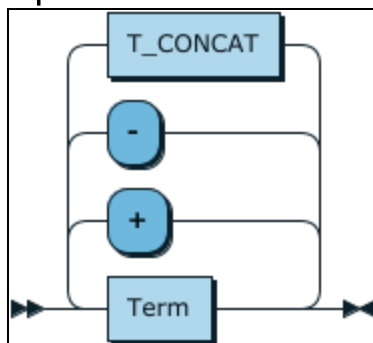
::= T_HAVING SearchCondition

| Empty

referenced by:

- QuerySpecification

Expression:



Expression

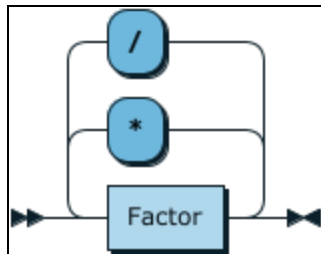
::= Term (('+' | '-' | T_CONCAT) Term)*

referenced by:

- [AggrExpressionPrimary](#)
- [CaseAbbreviation](#)
- [CastParams](#)
- [CoalesceListExpression](#)
- [ConvertParams](#)
- [DerivedColumn](#)
- [EscapeCharacter_Opt](#)
- [ExtractParams](#)
- [GroupAliasList](#)
- [GroupBy_Opt](#)
- [InPredicateValue](#)
- [LikePredicate](#)
- [Params_Opt](#)
- [PassingClauseValue](#)
- [PivotValueExpression](#)
- [PositionParams](#)
- [ProcedureParams_Opt](#)
- [Result](#)
- [RowValueConstructorElement](#)
- [ScalarOrAggrFn](#)
- [SetFunction](#)
- [SimpleCase](#)
- [SimpleWhenClause](#)
- [SortKey](#)
- [UpdateSource](#)

- [VSPredItem](#)
- [ValueExpressionPrimary](#)

Term:

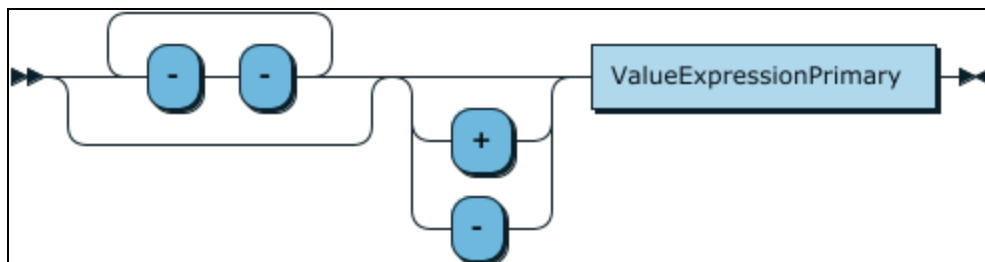


Term ::= **Factor** (**'*'** | **'/'**) **Factor** *

referenced by:

- [Expression](#)

Factor:

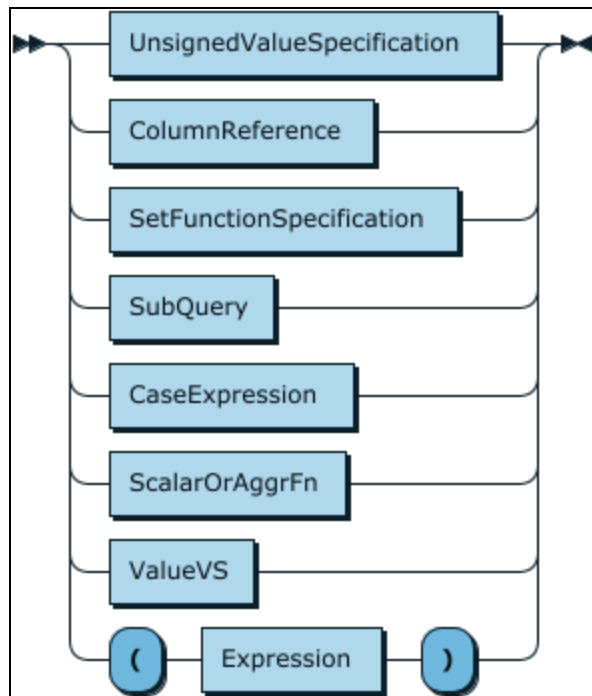


Factor ::= (**'-'**) * (**'+'** | **'-'**) ? **ValueExpressionPrimary**

referenced by:

- [Term](#)

ValueExpressionPrimary:



ValueExpressionPrimary

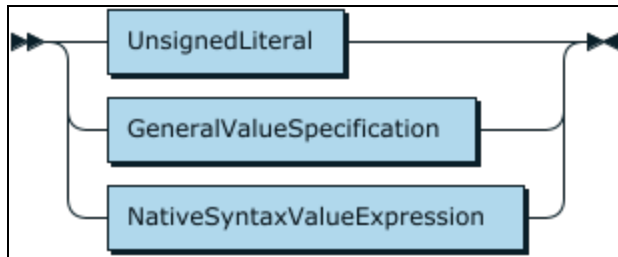
```

::= UnsignedValueSpecification
    | ColumnReference
    | SetFunctionSpecification
    | SubQuery
    | CaseExpression
    | ScalarOrAggrFn
    | ValueVS
    | "(" Expression ")"
  
```

referenced by:

- Factor

UnsignedValueSpecification:



```

UnsignedValueSpecification
  ::= UnsignedLiteral
  | GeneralValueSpecification
  | NativeSyntaxValueExpression
  
```

referenced by:

- Limit
- LimitWithSkip
- SelectLimit_Opt
- ValueExpressionPrimary

ValueSpecification:



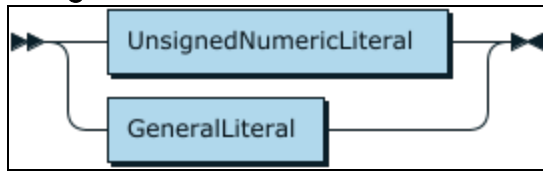
```

ValueSpecification
  ::= Literal
  | GeneralValueSpecification
  
```

referenced by:

- SetStmt

UnsignedLiteral:



UnsignedLiteral

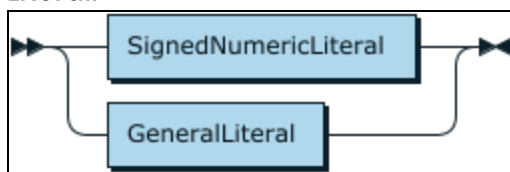
::= UnsignedNumericLiteral

| GeneralLiteral

referenced by:

- UnsignedValueSpecification

Literal:



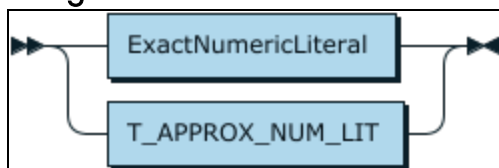
Literal ::= SignedNumericLiteral

| GeneralLiteral

referenced by:

- DataTypeAttributeValue
- ValueSpecification

UnsignedNumericLiteral:



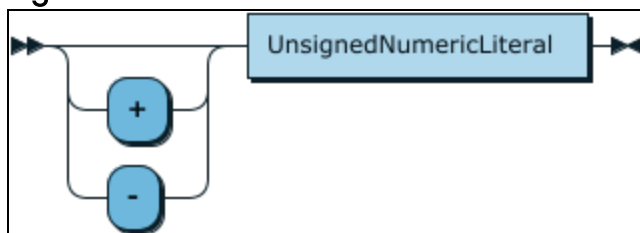
UnsignedNumericLiteral

::= ExactNumericLiteral
| T_APPROX_NUM_LIT

referenced by:

- SignedNumericLiteral
- UnsignedLiteral

SignedNumericLiteral:



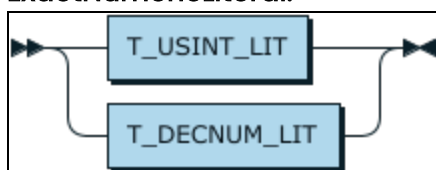
SignedNumericLiteral

::= ('+' | '-')? UnsignedNumericLiteral

referenced by:

- Literal

ExactNumericLiteral:



ExactNumericLiteral

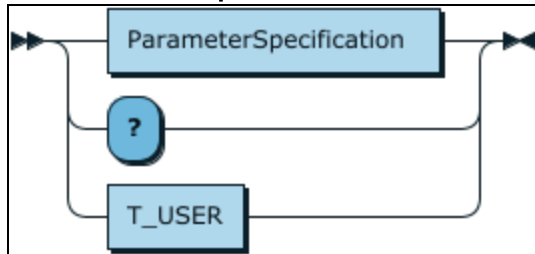
::= T_USINT_LIT

| T_DECNUM_LIT

referenced by:

- UnsignedNumericLiteral

GeneralValueSpecification:



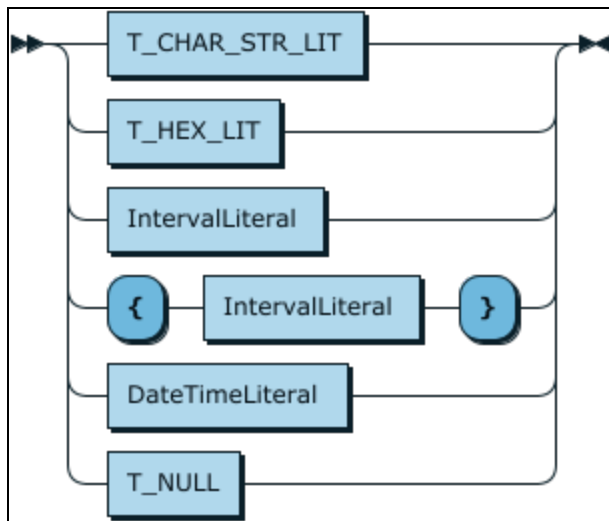
```

GeneralValueSpecification
    ::= ParameterSpecification
    | '?'
    | T_USER
    
```

referenced by:

- UnsignedValueSpecification
- ValueSpecification

GeneralLiteral:



GeneralLiteral

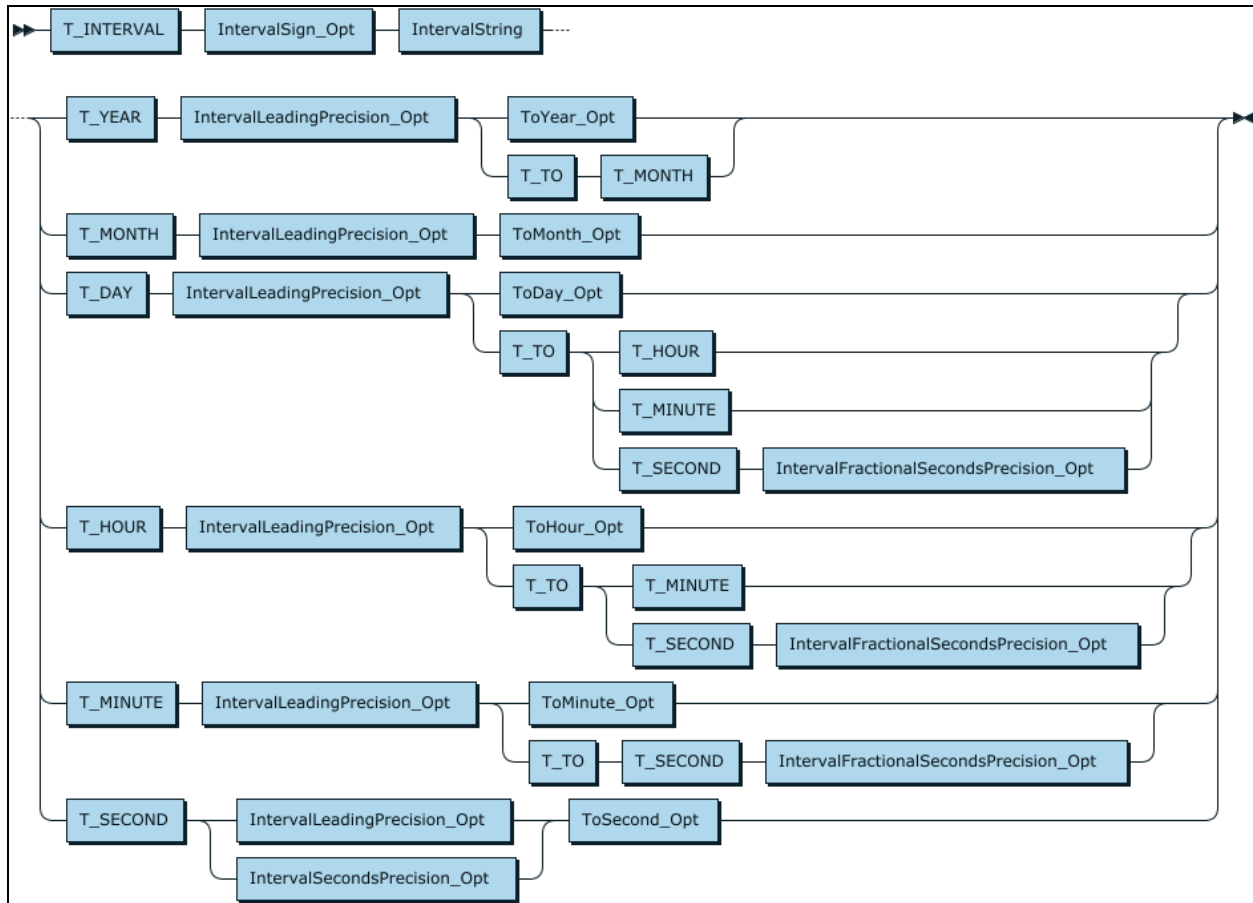
```

::= T_CHAR_STR_LIT
   | T_HEX_LIT
   | IntervalLiteral
   | '{ IntervalLiteral }'
   | DateTimeLiteral
   | T_NULL
    
```

referenced by:

- Literal
- UnsignedLiteral

IntervalLiteral:



IntervalLiteral

```

::= T_INTERVAL IntervalSign_Opt IntervalString ( T_YEAR IntervalLeadingPrecision_Opt ( ToYear_Opt |
    T_TO T_MONTH ) | T_MONTH IntervalLeadingPrecision_Opt ToMonth_Opt | T_
    DAY IntervalLeadingPrecision_Opt ( ToDay_Opt | T_TO ( T_HOUR | T_MINUTE |
    T_SECOND IntervalFractionalSecondsPrecision_Opt ) ) | T_HOUR
    IntervalLeadingPrecision_Opt ( ToHour_Opt | T_TO ( T_MINUTE | T_SECOND
    IntervalFractionalSecondsPrecision_Opt ) ) | T_MINUTE IntervalLeadingPrecision_
    Opt ( ToMinute_Opt | T_TOT_SECOND IntervalFractionalSecondsPrecision_Opt ) |
    T_SECOND ( IntervalLeadingPrecision_Opt | IntervalSecondsPrecision_Opt )
    ToSecond_Opt )

```

referenced by:

- GeneralLiteral

IntervalString:



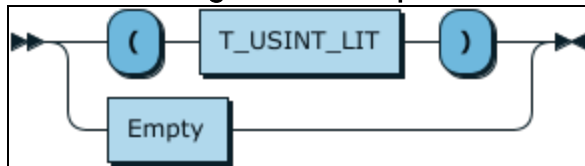
IntervalString

::= T_CHAR_STR_LIT

referenced by:

- IntervalLiteral

IntervalLeadingPrecision_Opt:



IntervalLeadingPrecision_Opt

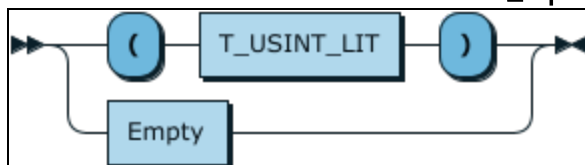
::= '(' T_USINT_LIT ')'

| Empty

referenced by:

- IntervalLiteral
- IntervalType

IntervalFractionalSecondsPrecision_Opt:



IntervalFractionalSecondsPrecision_Opt

```

::= '(' T_USINT_LIT ')'
    | Empty
    
```

referenced by:

- [IntervalLiteral](#)
- [IntervalType](#)

IntervalSecondsPrecision_Opt:



IntervalSecondsPrecision_Opt

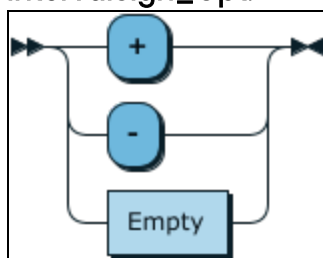
```

::= '(' T_USINT_LIT ',' T_USINT_LIT ')'
    
```

referenced by:

- [IntervalLiteral](#)
- [IntervalType](#)

IntervalSign_Opt:



IntervalSign_Opt

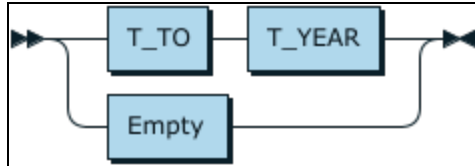
```

::= '+'
    | '-'
    | Empty
    
```

referenced by:

- [IntervalLiteral](#)

ToYear_Opt:

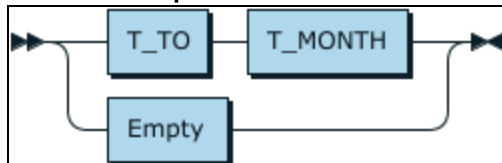


```
ToYear_Opt
  ::= T_TO T_YEAR
  | Empty
```

referenced by:

- [IntervalLiteral](#)
- [IntervalType](#)

ToMonth_Opt:

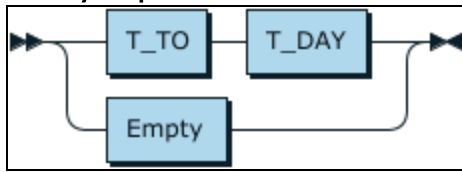


```
ToMonth_Opt
  ::= T_TO T_MONTH
  | Empty
```

referenced by:

- [IntervalLiteral](#)
- [IntervalType](#)

ToDay_Opt:



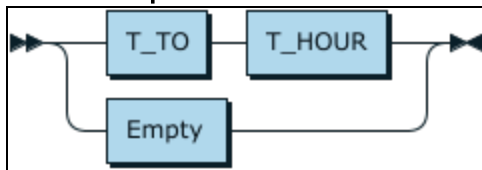
```

ToDay_Opt
    ::= T_TO T_DAY
    | Empty
    
```

referenced by:

- [IntervallLiteral](#)
- [IntervalType](#)

ToHour_Opt:



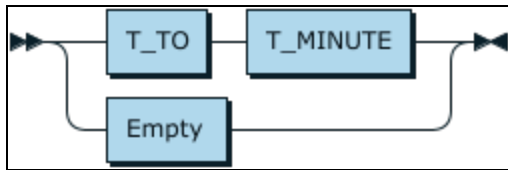
```

ToHour_Opt
    ::= T_TO T_HOUR
    | Empty
    
```

referenced by:

- [IntervallLiteral](#)
- [IntervalType](#)

ToMinute_Opt:

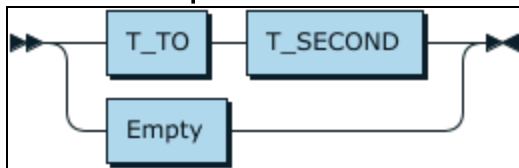


ToMinute_Opt
 ::= T_TO T_MINUTE
 | Empty

referenced by:

- [IntervallLiteral](#)
- [IntervalType](#)

ToSecond_Opt:

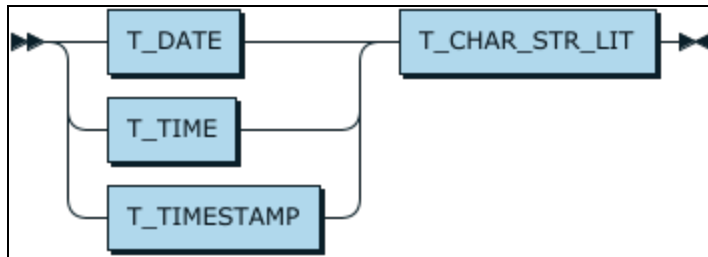


ToSecond_Opt
 ::= T_TO T_SECOND
 | Empty

referenced by:

- [IntervallLiteral](#)
- [IntervalType](#)

DateTimeLiteral:



`DateTimeLiteral`

`::= ( T_DATE | T_TIME | T_TIMESTAMP ) T_CHAR_STR_LIT`

referenced by:

- `GeneralLiteral`

ParameterSpecification:



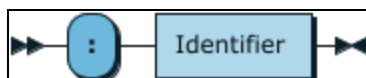
`ParameterSpecification`

`::= ParameterName`

referenced by:

- `GeneralValueSpecification`

ParameterName:



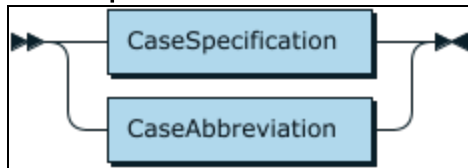
`ParameterName`

`::= ':' Identifier`

referenced by:

- [ParameterSpecification](#)

CaseExpression:



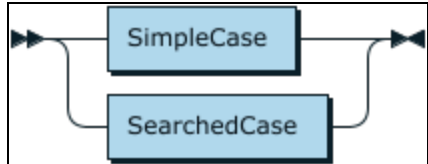
[CaseExpression](#)

[::= CaseSpecification](#)
[| CaseAbbreviation](#)

referenced by:

- [ValueExpressionPrimary](#)

CaseSpecification:



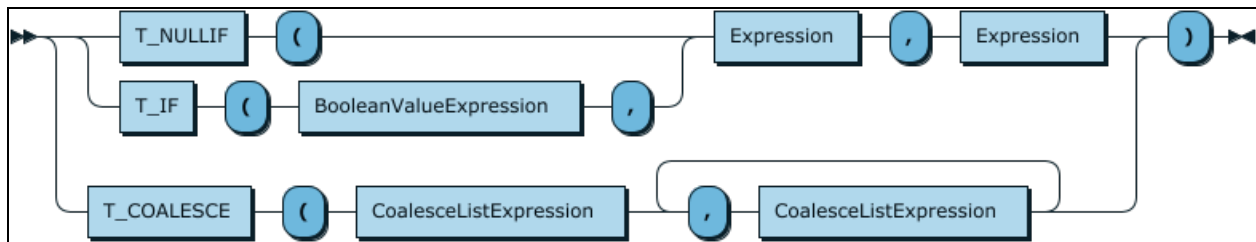
[CaseSpecification](#)

[::= SimpleCase](#)
[| SearchedCase](#)

referenced by:

- [CaseExpression](#)

CaseAbbreviation:



CaseAbbreviation

```

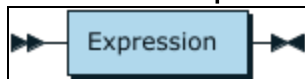
::= ( ( T_NULLIF '(' | T_IF '(' BooleanValueExpression ',' Expression ',' Expression | T_COALESCE '('
      CoalesceListExpression ',' CoalesceListExpression )+ ) ')'

```

referenced by:

- CaseExpression

CoalesceListExpression:



CoalesceListExpression

```

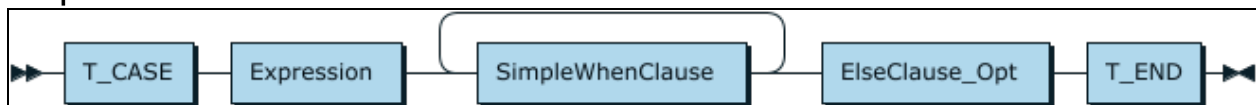
::= Expression

```

referenced by:

- CaseAbbreviation

SimpleCase:



SimpleCase

```

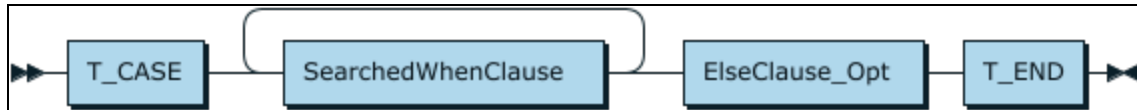
::= T_CASE Expression SimpleWhenClause+ ElseClause_Opt T_END

```

referenced by:

- [CaseSpecification](#)

SearchedCase:



SearchedCase

::= T_CASE SearchedWhenClause+ ElseClause_Opt T_END

referenced by:

- [CaseSpecification](#)

SimpleWhenClause:



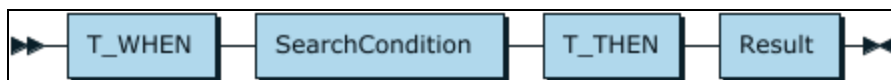
SimpleWhenClause

::= T_WHEN Expression T_THEN Result

referenced by:

- [SimpleCase](#)

SearchedWhenClause:



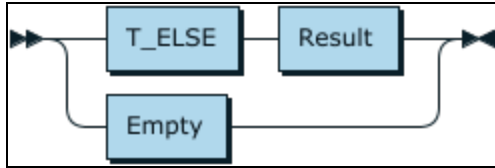
SearchedWhenClause

::= T_WHEN SearchCondition T_THEN Result

referenced by:

- [SearchedCase](#)

ElseClause_Opt:



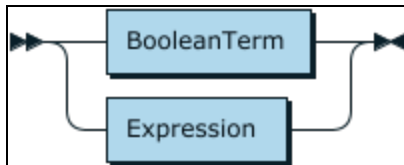
```

ElseClause_Opt
  ::= T_ELSE Result
  | Empty
  
```

referenced by:

- [SearchedCase](#)
- [SimpleCase](#)

Result:



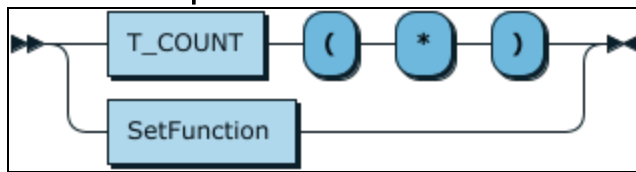
```

Result ::= BooleanTerm
  | Expression
  
```

referenced by:

- [ElseClause_Opt](#)
- [SearchedWhenClause](#)
- [SimpleWhenClause](#)

SetFunctionSpecification:



SetFunctionSpecification

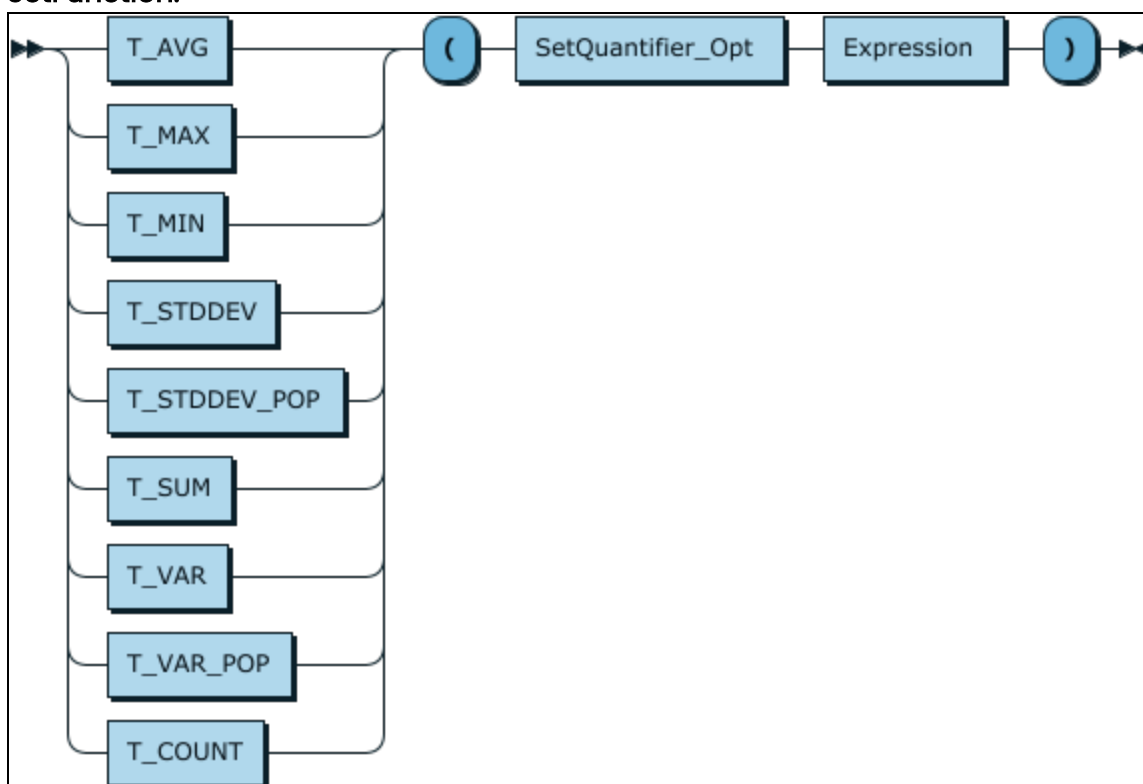
::= T_COUNT '(' '*' ')'

| SetFunction

referenced by:

- ValueExpressionPrimary

SetFunction:



SetFunction

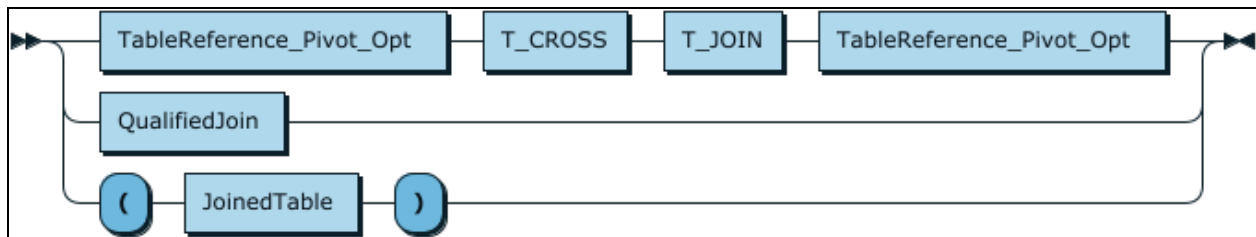
```

::= ( T_AVG | T_MAX | T_MIN | T_STDDEV | T_STDDEV_POP | T_SUM | T_VAR | T_VAR_POP | T_
      COUNT ) '(' SetQuantifier_Opt Expression ')'
    
```

referenced by:

- SetFunctionSpecification

JoinedTable:



JoinedTable

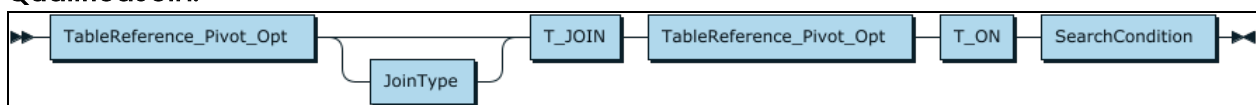
```

::= TableReference_Pivot_Opt T_CROSS T_JOIN TableReference_Pivot_Opt
    | QualifiedJoin
    | '(' JoinedTable ')'
    
```

referenced by:

- JoinedTable
- OuterJoinVT
- TableReference

QualifiedJoin:



QualifiedJoin

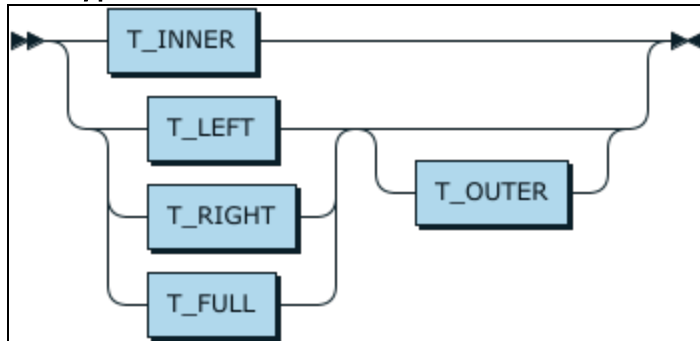
```

::= TableReference_Pivot_Opt JoinType? T_JOIN TableReference_Pivot_Opt T_ON SearchCondition
    
```

referenced by:

- [JoinedTable](#)

JoinType:

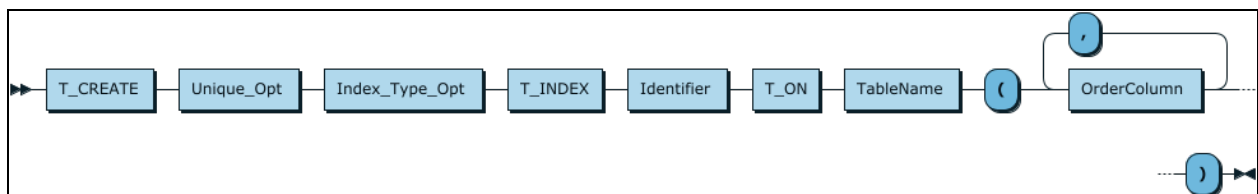


```
JoinType ::= T_INNER
          | ( T_LEFT | T_RIGHT | T_FULL ) T_OUTER?
```

referenced by:

- [QualifiedJoin](#)

CreateIndexStmt:

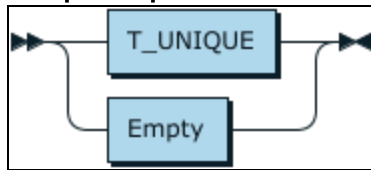


```
CreateIndexStmt
::= T_CREATE Unique_Opt Index_Type_Opt T_INDEX Identifier T_ON TableName '(' OrderColumn (','
    OrderColumn)* ')'
```

referenced by:

- [ExecStmt](#)

Unique_Opt:



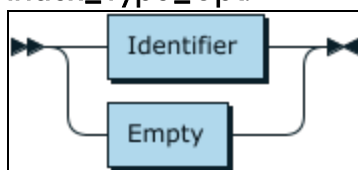
```

Unique_Opt
  ::= T_UNIQUE
  | Empty
  
```

referenced by:

- CreateIndexStmt

Index_Type_Opt:



```

Index_Type_Opt
  ::= Identifier
  | Empty
  
```

referenced by:

- CreateIndexStmt

OrderColumn:



```

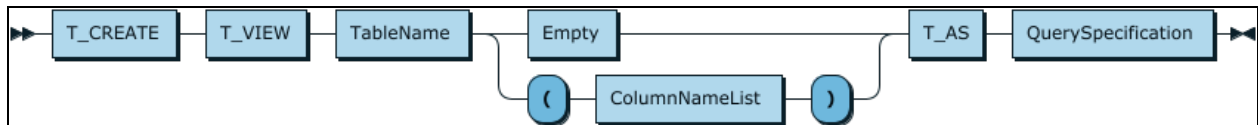
OrderColumn
  
```

::= Identifier OrderingSpecification_Opt

referenced by:

- CreateIndexStmt

CreateViewStmt:



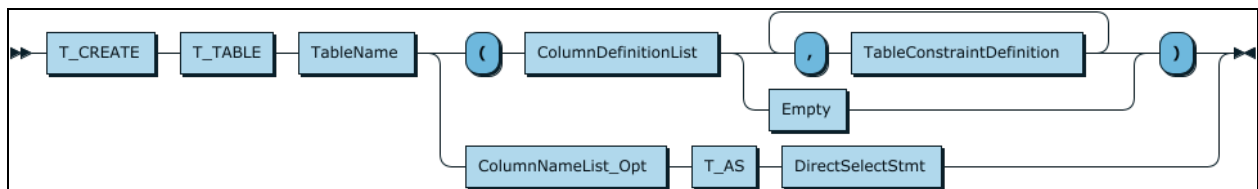
CreateViewStmt

::= T_CREATE T_VIEW TableName (Empty | '(' ColumnNameList ')') T_AS QuerySpecification

referenced by:

- ExecStmt

CreateTableStmt:



CreateTableStmt

::= T_CREATE T_TABLE TableName ('(' ColumnDefinitionList ((',' TableConstraintDefinition)+ | Empty)
' ' | ColumnNameList_Opt T_AS DirectSelectStmt)

referenced by:

- ExecStmt

ColumnDefinitionList:



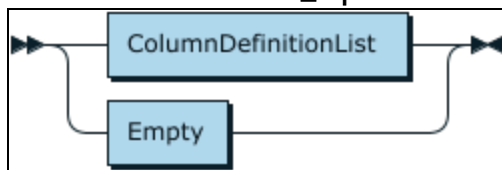
ColumnDefinitionList

::= ColumnDefinition (',' ColumnDefinition)*

referenced by:

- ColumnDefinitionList_Opt
- CreateTableStmt

ColumnDefinitionList_Opt:



ColumnDefinitionList_Opt

::= ColumnDefinitionList
| Empty

referenced by:

- ColumnsClause

ColumnDefinition:



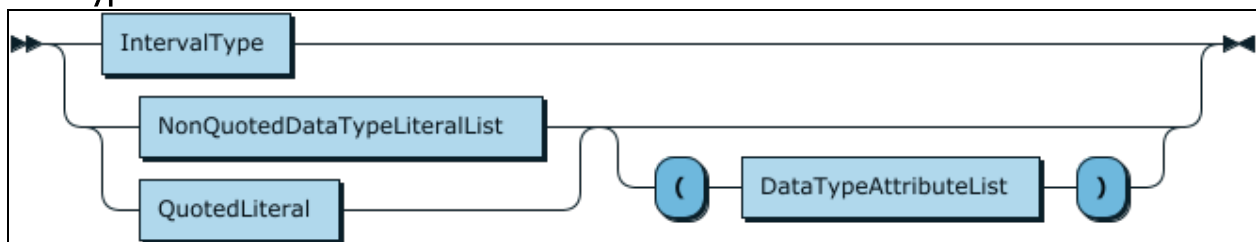
ColumnDefinition

```
 ::= ColumnName DataType ColumnConstraintDefinition_Opt
```

referenced by:

- AddColumnDefinition
- ColumnDefinitionList

DataType:



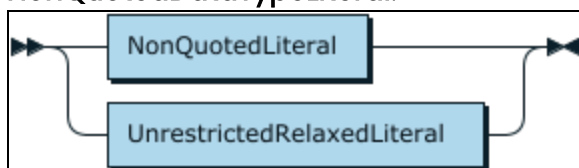
```

DataType ::= IntervalType
          | ( NonQuotedDataTypeLiteralList | QuotedLiteral ) ( '[' DataTypeAttributeList ']' )?
    
```

referenced by:

- CastParams
- ColumnDefinition
- ConvertParams
- DataType_Opt
- ReturningClause

NonQuotedDataTypeLiteral:

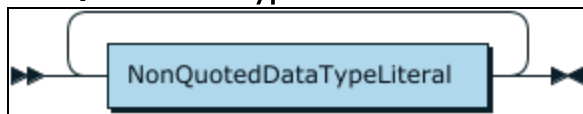


```
NonQuotedDataTypeLiteral
::= NonQuotedLiteral
   | UnrestrictedRelaxedLiteral
```

referenced by:

- NonQuotedDataTypeLiteralList

NonQuotedDataTypeLiteralList:

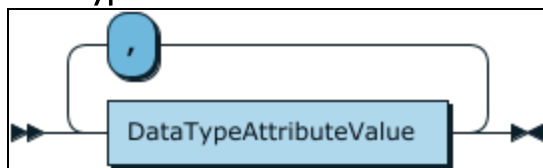


```
NonQuotedDataTypeLiteralList
::= NonQuotedDataTypeLiteral+
```

referenced by:

- DataType

DataTypeAttributeList:

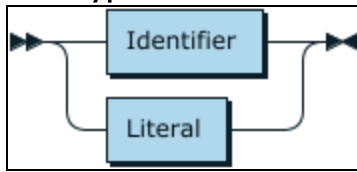


```
DataTypeAttributeList
::= DataTypeAttributeValue (',' DataTypeAttributeValue)*
```

referenced by:

- DataType
- DataTypeAttributeList_Opt

DataTypeAttributeValue:



DataTypeAttributeValue

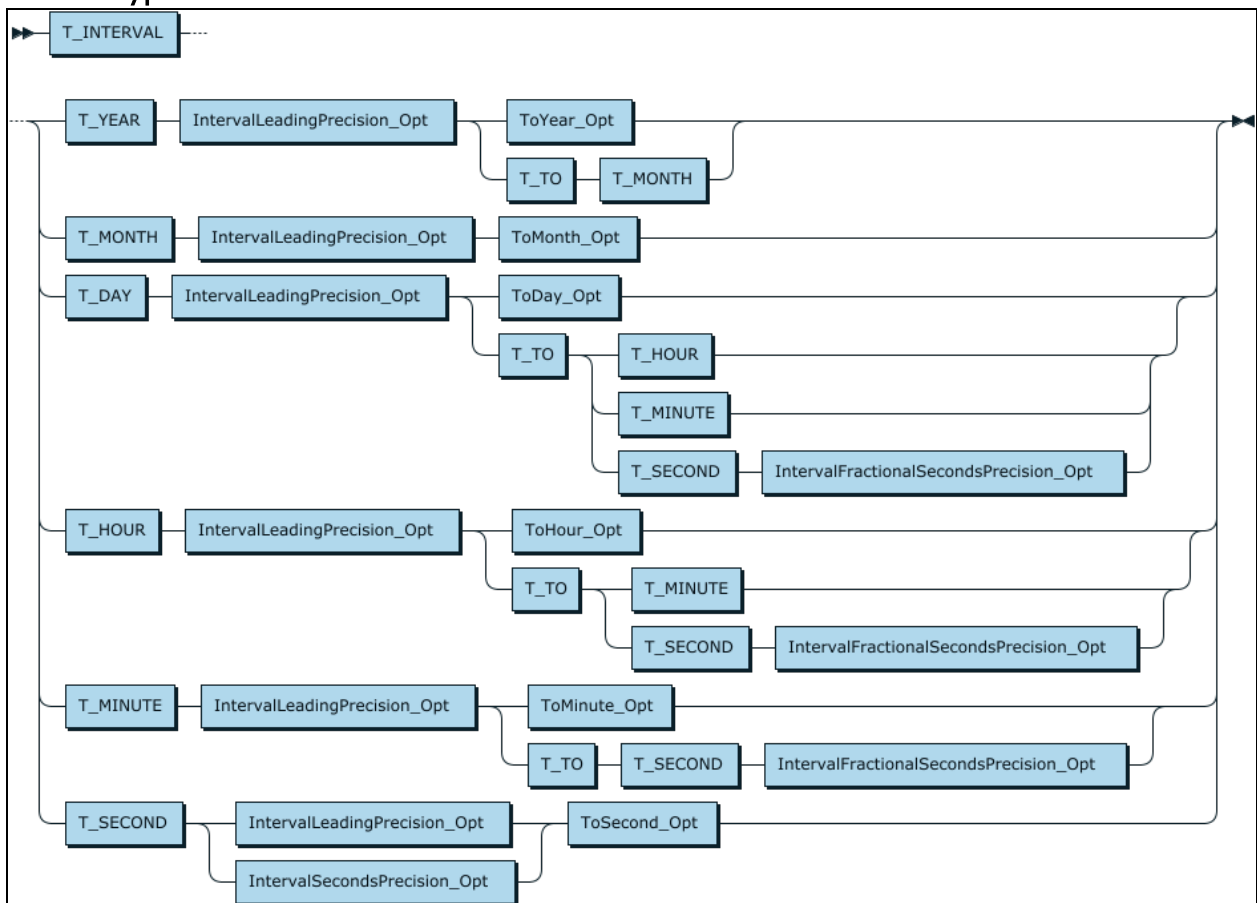
```

::= Identifier
| Literal
  
```

referenced by:

- [DataTypeAttributeList](#)

IntervalType:



IntervalType

```

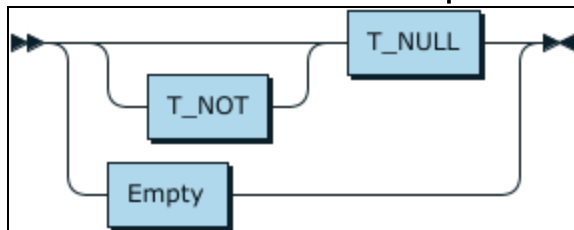
::= T_INTERVAL ( T_YEAR IntervalLeadingPrecision_Opt ( ToYear_Opt | T_TO T_MONTH ) | T_MONTH
    IntervalLeadingPrecision_Opt ToMonth_Opt | T_DAY IntervalLeadingPrecision_
    Opt ( ToDay_Opt | T_TO ( T_HOUR | T_MINUTE | T_SECOND
    IntervalFractionalSecondsPrecision_Opt ) ) | T_HOUR IntervalLeadingPrecision_
    Opt ( ToHour_Opt | T_TO ( T_MINUTE | T_
    SECONDIntervalFractionalSecondsPrecision_Opt ) ) | T_MINUTE
    IntervalLeadingPrecision_Opt ( ToMinute_Opt | T_TO T_
    SECONDIntervalFractionalSecondsPrecision_Opt ) | T_SECOND (
    IntervalLeadingPrecision_Opt | IntervalSecondsPrecision_Opt ) ToSecond_Opt )

```

referenced by:

- [DataType](#)

ColumnConstraintDefinition_Opt:



ColumnConstraintDefinition_Opt

```

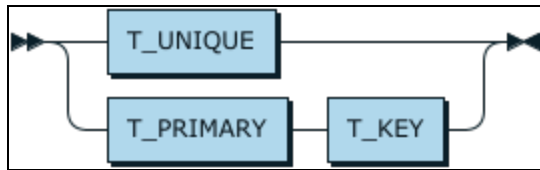
::= T_NOT? T_NULL
    | Empty

```

referenced by:

- [ColumnDefinition](#)

UniqueSpecification:



UniqueSpecification

::= T_UNIQUE

| T_PRIMARY T_KEY

referenced by:

- TableConstraintDefinition

TableConstraintDefinition:



TableConstraintDefinition

::= UniqueSpecification '(' ColumnNameList ')'

referenced by:

- CreateTableStmt

DropTableStmt:



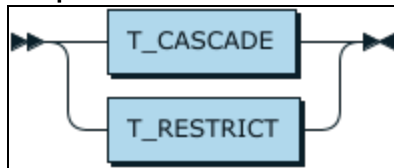
DropTableStmt

::= T_DROP T_TABLE TableName

referenced by:

- ExecStmt

DropBehavior:



DropBehavior

```

::= T_CASCADE
| T_RESTRICT
  
```

referenced by:

- DropViewStmt
- RevokeStmt

DropViewStmt:



DropViewStmt

```

::= T_DROP T_VIEW TableName DropBehavior
  
```

referenced by:

- ExecStmt

DropIndexStmt:



DropIndexStmt

::= T_DROP T_INDEX Identifier T_ON TableName

referenced by:

- ExecStmt

AlterTableStmt:



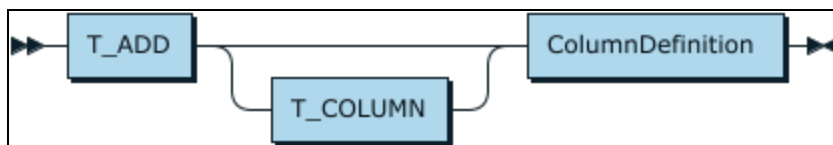
AlterTableStmt

::= T_ALTER T_TABLE TableName Add ColumnDefinition

referenced by:

- ExecStmt

AddColumnDefinition:



AddColumnDefinition

::= T_ADD T_COLUMN? ColumnDefinition

referenced by:

- AlterTableStmt

DeleteStmtSearched:



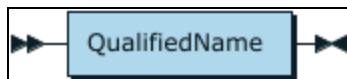
DeleteStmtSearched

::= T_DELETE T_FROM TableName WhereSearchCond_Opt

referenced by:

- ExecStmt

TableName:



TableName

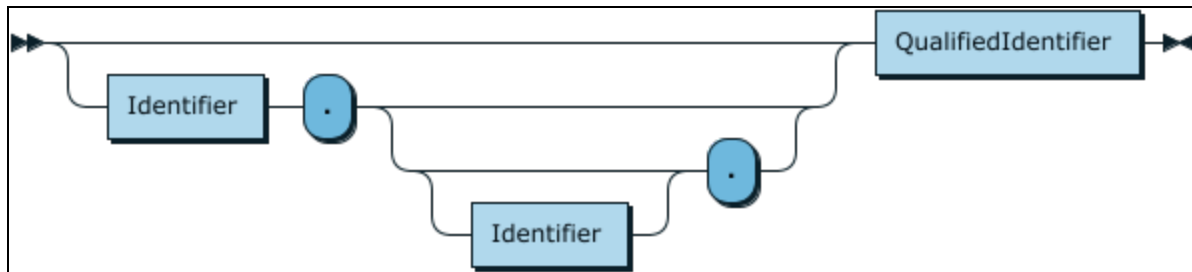
::= QualifiedName

referenced by:

- AlterTableStmt
- CreateIndexStmt
- CreateTableStmt
- CreateViewStmt
- DeleteStmtSearched
- DropIndexStmt
- .DropTableStmt
- DropViewStmt
- InsertStmt
- MergeStmt
- ObjectName

- TablePrimary
- UpdateStmtSearched
- UpsertStmtSearched

QualifiedName:



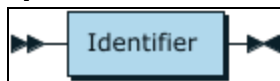
QualifiedName

$::= (\text{Identifier} '!' (\text{Identifier?} '!')?)? \text{QualifiedIdentifier}$

referenced by:

- TableName

QualifiedIdentifier:



QualifiedIdentifier

$::= \text{Identifier}$

referenced by:

- QualifiedName

InsertStmt:



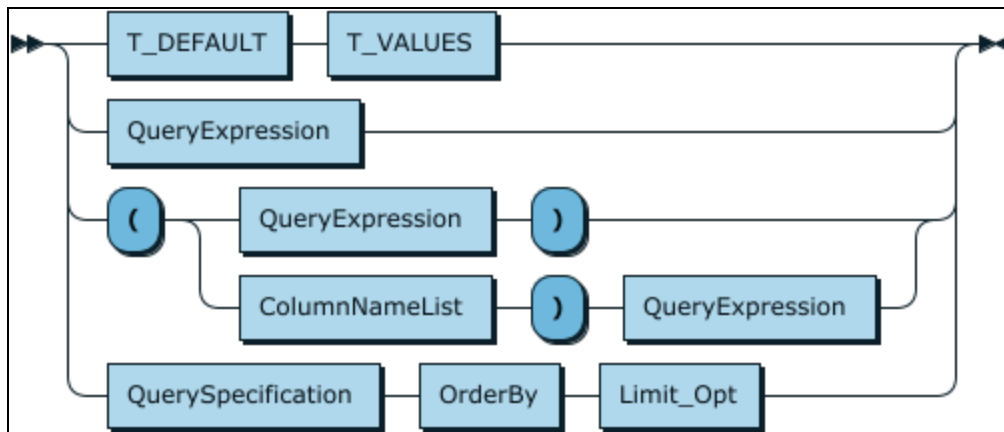
InsertStmt

```
::= T_INSERT T_INT0 TableName InsertList
```

referenced by:

- ExecStmt

InsertList:



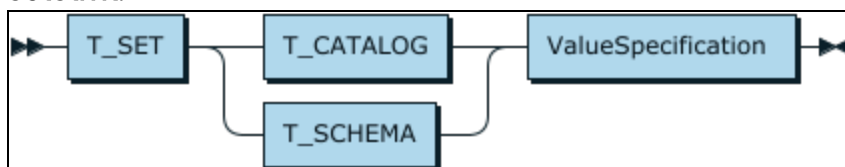
InsertList

```
::= T_DEFAULT T_VALUES
| QueryExpression
| '(' ( QueryExpression ) | ColumnNameList ')' QueryExpression
| QuerySpecification OrderBy Limit_Opt
```

referenced by:

- InsertStmt

SetStmt:

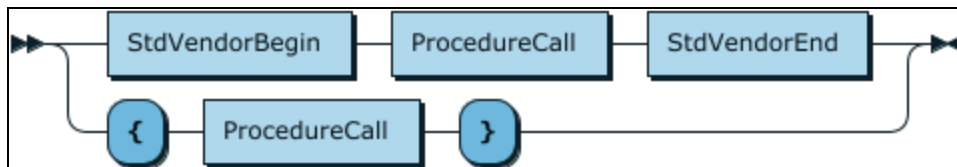


SetStmt ::= T_SET (T_CATALOG | T_SCHEMA) ValueSpecification

referenced by:

- ExecStmt

ProcedureStmt:



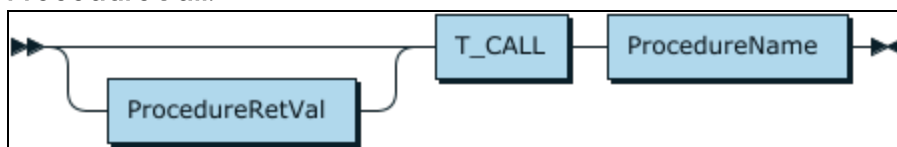
ProcedureStmt

::= StdVendorBegin ProcedureCall StdVendorEnd
| '{ ProcedureCall }'

referenced by:

- ExecStmt

ProcedureCall:



ProcedureCall

::= ProcedureRetVal? T_CALL ProcedureName

referenced by:

- ProcedureStmt

ProcedureName:



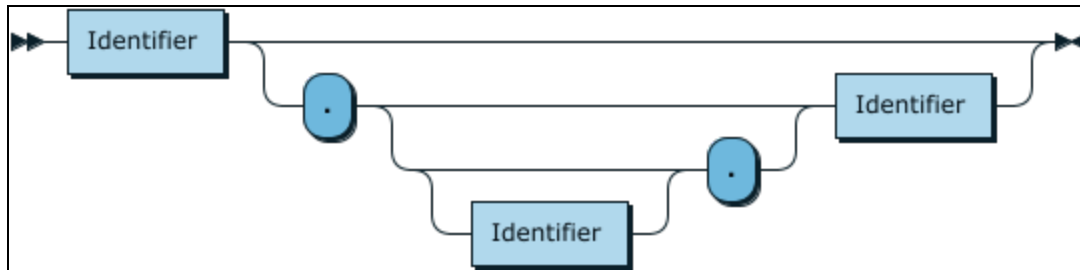
ProcedureName

::= QualifiedProcedureName ProcedureParams_Opt

referenced by:

- ProcedureCall

QualifiedProcedureName:



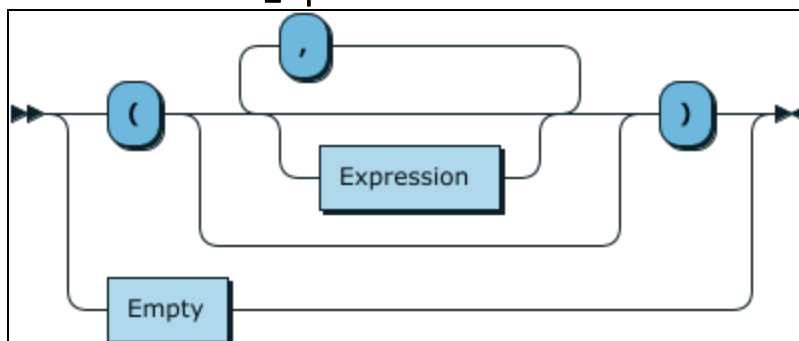
QualifiedProcedureName

::= Identifier ('(' Identifier? ')')? Identifier)?

referenced by:

- ProcedureName

ProcedureParams_Opt:



ProcedureParams_Opt

```
::= '(' ( Expression? ( ';' Expression? )* )? )? ')'
| Empty
```

referenced by:

- ProcedureName

ProcedureRetVal:



ProcedureRetVal

```
::= '?' T_EQ
```

referenced by:

- ProcedureCall

ReturnEscapeStmt:



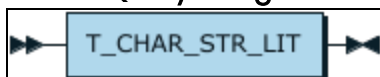
ReturnEscapeStmt

```
::= '{' T_RETURN ColumnNameList T_FROM NestedQueryString '}'
```

referenced by:

- ExecStmt

NestedQueryString:



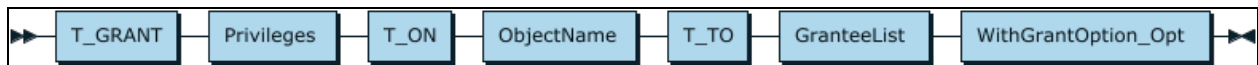
NestedQueryString

::= T_CHAR_STR_LIT

referenced by:

- ReturnEscapeStmt

GrantStmt:



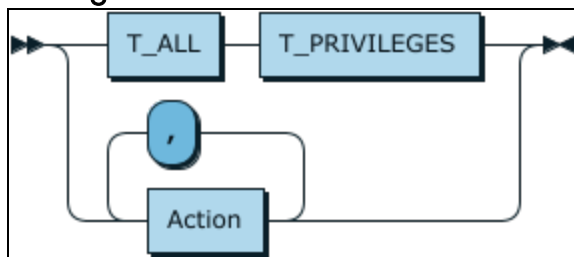
GrantStmt

::= T_GRANT Privileges T_ON ObjectName T_TO GranteeList WithGrantOption_Opt

referenced by:

- ExecStmt

Privileges:



Privileges

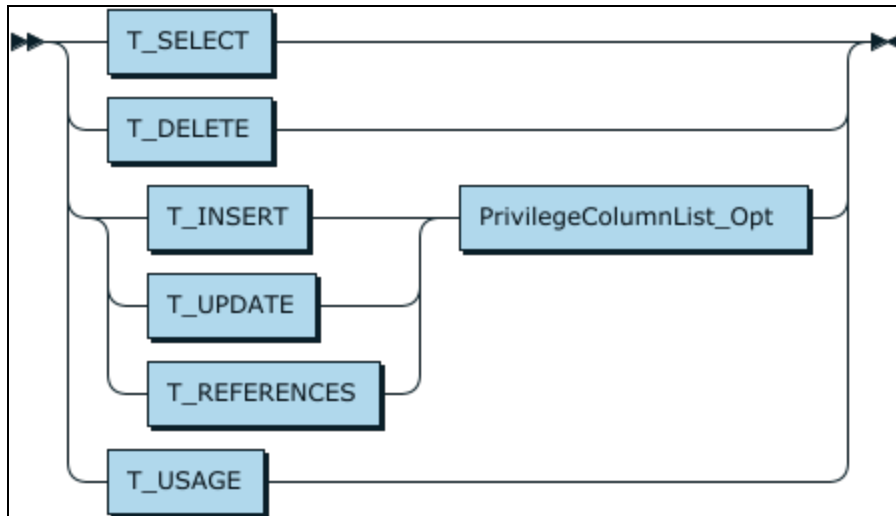
::= T_ALL T_PRIVILEGES

| Action (',' Action)*

referenced by:

- GrantStmt
- RevokeStmt

Action:



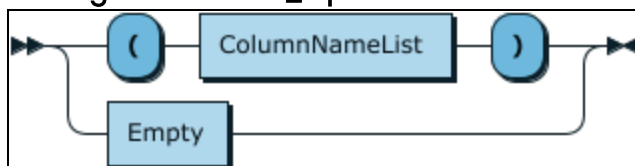
```

Action ::= T_SELECT
        | T_DELETE
        | ( T_INSERT | T_UPDATE | T_REFERENCES ) PrivilegeColumnList_Opt
        | T_USAGE
  
```

referenced by:

- Privileges

PrivilegeColumnList_Opt:



```

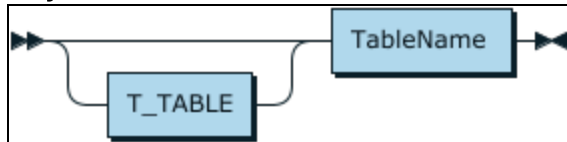
PrivilegeColumnList_Opt
    ::= '(' ColumnNameList ')'
  
```

| Empty

referenced by:

- Action

ObjectName:



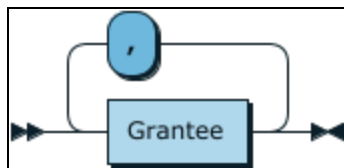
ObjectName

::= T_TABLE? TableName

referenced by:

- GrantStmt
- RevokeStmt

GranteeList:



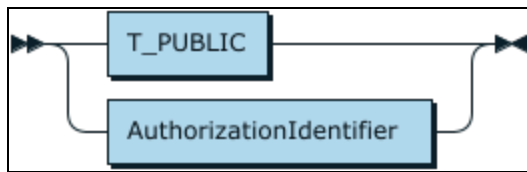
GranteeList

::= Grantee (',' Grantee)*

referenced by:

- GrantStmt
- RevokeStmt

Grantee:

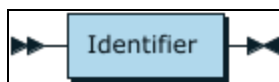


```
Grantee ::= T_PUBLIC
        | AuthorizationIdentifier
```

referenced by:

- GranteeList

AuthorizationIdentifier:



```
AuthorizationIdentifier
    ::= Identifier
```

referenced by:

- Grantee

WithGrantOption_Opt:

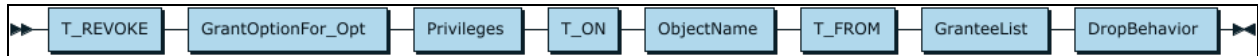


```
WithGrantOption_Opt
    ::= T_WITH T_GRANT T_OPTION
    | Empty
```

referenced by:

- [GrantStmt](#)

RevokeStmt:



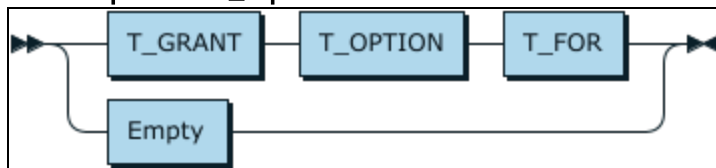
[RevokeStmt](#)

::= T_REVOKE GrantOptionFor_Opt Privileges T_ON ObjectName T_FROM GranteeList DropBehavior

referenced by:

- [ExecStmt](#)

GrantOptionFor_Opt:



[GrantOptionFor_Opt](#)

::= T_GRANT T_OPTION T_FOR

| Empty

referenced by:

- [RevokeStmt](#)

UpdateStmtSearched:



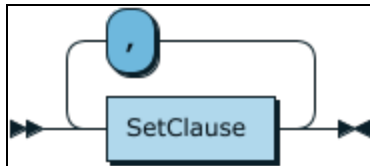
[UpdateStmtSearched](#)

```
::= T_UPDATE TableName T_SET SetClauseList WhereSearchCond_Opt
```

referenced by:

- ExecStmt

SetClauseList:



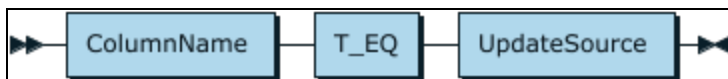
SetClauseList

```
::= SetClause (',' SetClause)*
```

referenced by:

- MergeWhenMatchedClause
- UpdateStmtSearched
- UpsertStmtSearched

SetClause:



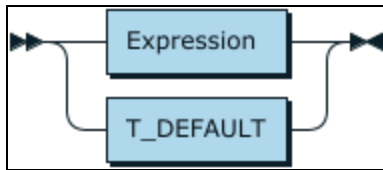
SetClause

```
::= ColumnName T_EQ UpdateSource
```

referenced by:

- SetClauseList

UpdateSource:



UpdateSource

::= Expression
| T_DEFAULT

referenced by:

- SetClause

UpsertStmtSearched:



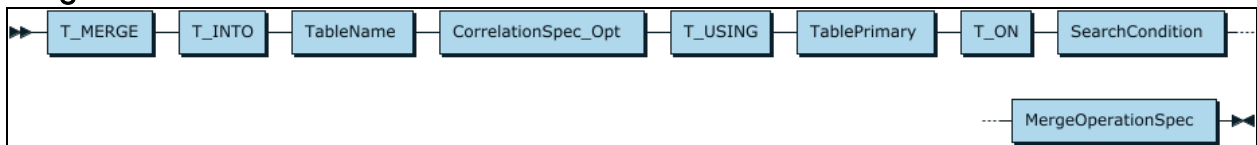
UpsertStmtSearched

::= T_UPSERT TableName T_SET SetClauseList WhereSearchCond_Opt

referenced by:

- ExecStmt

MergeStmt:



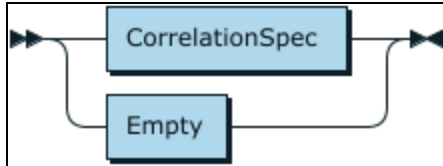
MergeStmt

::= T_MERGE T_INTRO TableNameCorrelationSpec_Opt T_USING TablePrimary T_ON SearchCondition
MergeOperationSpec

referenced by:

- ExecStmt

CorrelationSpec_Opt:



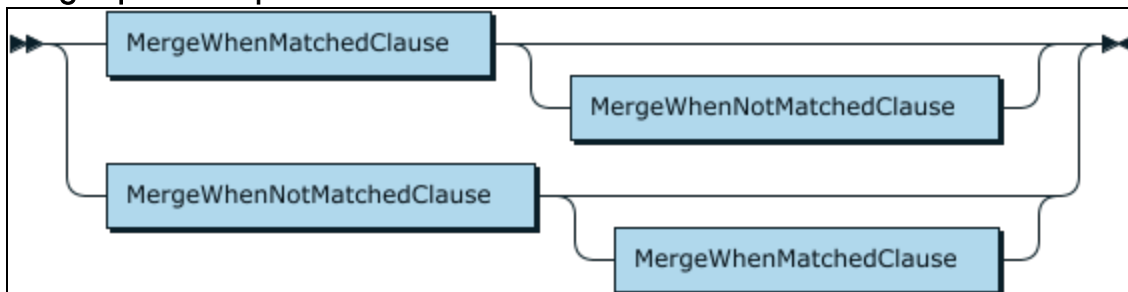
```

CorrelationSpec_Opt
  ::= CorrelationSpec
  | Empty
  
```

referenced by:

- AggrExpressionPrimary
- MergeStmt
- PivotClause
- PivotColumn
- TablePrimary
- UnpivotClause

MergeOperationSpec:



```

MergeOperationSpec
  
```

```

::= MergeWhenMatchedClause MergeWhenNotMatchedClause?
    | MergeWhenNotMatchedClause MergeWhenMatchedClause?
    
```

referenced by:

- [MergeStmt](#)

MergeWhenMatchedClause:



MergeWhenMatchedClause

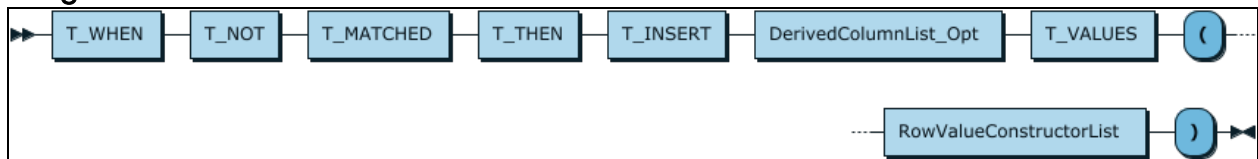
```

::= T_WHEN T_MATCHED T_THEN T_UPDATE T_SET SetClauseList
    
```

referenced by:

- [MergeOperationSpec](#)

MergeWhenNotMatchedClause:



MergeWhenNotMatchedClause

```

::= T_WHEN T_NOT T_MATCHED T_THEN T_INSERT DerivedColumnList_Opt T_VALUES '('
    RowValueConstructorList ')'
    
```

referenced by:

- [MergeOperationSpec](#)

NativeSyntaxStmt:



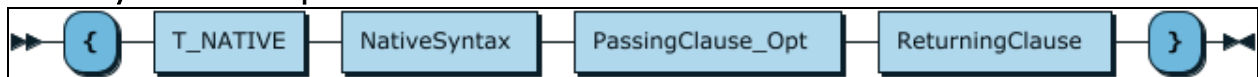
NativeSyntaxStmt

::= '{ T_NATIVE NativeSyntax PassingClause_Opt ColumnsClause }'

referenced by:

- [QueryExpression](#)

NativeSyntaxValueExpression:



NativeSyntaxValueExpression

::= '{ T_NATIVE NativeSyntax PassingClause_Opt ReturningClause }'

referenced by:

- [UnsignedValueSpecification](#)

NativeSyntax:



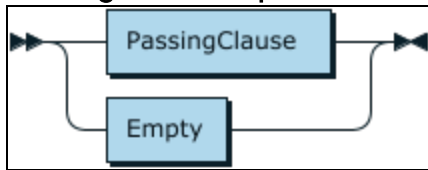
NativeSyntax

::= T_CHAR_STR_LIT

referenced by:

- [NativeSyntaxStmt](#)
- [NativeSyntaxValueExpression](#)

PassingClause_Opt:



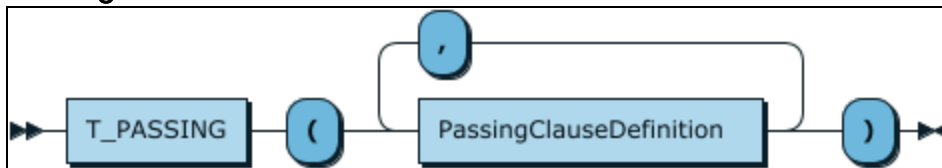
```

PassingClause_Opt
  ::= PassingClause
  | Empty
  
```

referenced by:

- NativeSyntaxStmt
- NativeSyntaxValueExpression

PassingClause:



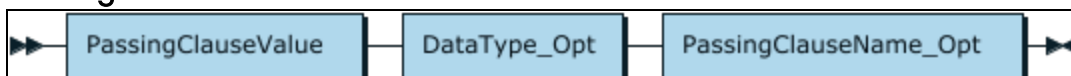
```

PassingClause
  ::= T_PASSING '(' PassingClauseDefinition (',' PassingClauseDefinition)* ')'
  
```

referenced by:

- PassingClause_Opt

PassingClauseDefinition:



```

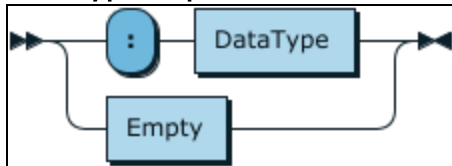
PassingClauseDefinition
  
```

```
 ::= PassingClauseValue DataType_Opt PassingClauseName_Opt
```

referenced by:

- [PassingClause](#)

DataType_Opt:



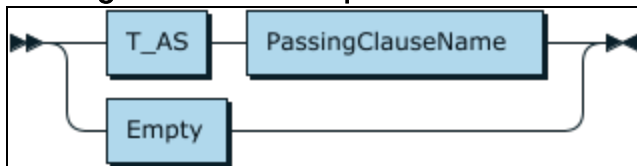
DataType_Opt

```
 ::= ':' DataType
    | Empty
```

referenced by:

- [PassingClauseDefinition](#)

PassingClauseName_Opt:



PassingClauseName_Opt

```
 ::= T_AS PassingClauseName
    | Empty
```

referenced by:

- [PassingClauseDefinition](#)

PassingClauseName:



PassingClauseName

::= RestrictedIdentifier

referenced by:

- PassingClauseName_Opt

PassingClauseValue:



PassingClauseValue

::= Expression

referenced by:

- PassingClauseDefinition

ReturningClause:



ReturningClause

::= T_RETURNING DataType

referenced by:

- NativeSyntaxValueExpression

ColumnsClause:



ColumnsClause

```
::= T_COLUMNS '(' ColumnDefinitionList_Opt ')'
```

referenced by:

- [NativeSyntaxStmt](#)

Appendix 2: Token Definitions

This section defines the tokens for the C++ SQL BNF Grammar described in Appendix 1.

The following token matches the pattern:

T_APPROX_NUM_LIT

```
((DecNumLit) | {USInt}) ("E" | "e") ("+" | "-") ? {USInt}
```

T_CALL

```
(c | C) (a | A) (l | L) (l | L)
```

T_CHAR_STR_LIT

```
({Chars} | "'" | "\"" ) *
```

T_DECNUM_LIT

```
(( {USInt} "." {USInt} ) | ( {USInt} "." ) | ( "." {USInt} ) )
```

T_FN

```
(f | F) (n | N)
```

T_HEX_LIT

```
0 (x | X) [a-fA-F0-9] *
```

T_IDENTIFIER

```
((letter) | "_") ((letter) | {digit} | "_") *
```

If an identifier matches one of the following strings (case-insensitive), it's mapped to the associated token in the following table:

Identifier	Token
"ADD"	T_ADD
"AND"	T_AND
"ANY"	T_ANY
"ALL"	T_ALL

Identifier	Token
"ALTER"	T_ALTER
"AS"	T_AS
"ASC"	T_ASC
"AVG"	T_AVG
"BETWEEN"	T_BETWEEN
"BY"	T_BY
"CASCADE"	T_CASCADE
"CAST"	T_CAST
"CATALOG"	T_CATALOG
"CHECK"	T_CHECK
"COALESCE"	T_COALESCE
"COLUMN"	T_COLUMN
"COLUMNS"	T_COLUMNS
"CONVERT"	T_CONVERT
"COUNT"	T_COUNT
"CREATE"	T_CREATE
"CROSS"	T_CROSS
"CURRENT_DATE"	T_CURRENT_DATE

Identifier	Token
"CURRENT_TIME"	T_CURRENT_TIME
"CURRENT_TIMESTAMP"	T_CURRENT_TIMESTAMP
"DATE"	T_DATE
"DAY"	T_DAY
"DAYOFWEEK"	T_DAYOFWEEK
"DEFAULT"	T_DEFAULT
"DELETE"	T_DELETE
"DESC"	T_DESC
"DISTINCT"	T_DISTINCT
"DROP"	T_DROP
"ELSE"	T_ELSE
"END"	T_END
"ESCAPE"	T_ESCAPE
"EXCEPT"	T_EXCEPT
"EXCLUDE"	T_EXCLUDE
"EXISTS"	T_EXISTS
"EXTRACT"	T_EXTRACT
"FOR"	T_FOR

Identifier	Token
"FORMAT"	T_FORMAT
"FROM"	T_FROM
"GRANT"	T_GRANT
"GROUP"	T_GROUP
"HAVING"	T_HAVING
"HOUR"	T_HOUR
"IN"	T_IN
"INCLUDE"	T_INCLUDE
"INDEX"	T_INDEX
"INNER"	T_INNER
"INSERT"	T_INSERT
"INTERSECT"	T_INTERSECT
"INTERVAL"	T_INTERVAL
"INTO"	T_INTO
"IS"	T_IS
"JOIN"	T_JOIN
"KEY"	T_KEY
"LEFT"	T_LEFT

Identifier	Token
"LIMIT"	T_LIMIT
"MATCHED"	T_MATCHED
"MAX"	T_MAX
"MERGE"	T_MERGE
"MIN"	T_MIN
"MINUTE"	T_MINUTE
"MONTH"	T_MONTH
"NATIVE"	T_NATIVE
"NULL"	T_NULL
"NULLIF"	T_NULLIF
"NULLS"	T_NULLS
"ON"	T_ON
"OPTION"	T_OPTION
"OR"	T_OR
"ORDER"	T_ORDER
"OUTER"	T_OUTER
"PASSING"	T_PASSING
"PERCENT"	T_PERCENT

Identifier	Token
"PIVOT"	T_PIVOT
"POSITION"	T_POSITION
"PRIMARY"	T_PRIMARY
"PRIVILEGES"	T_PRIVILEGES
"PUBLIC"	T_PUBLIC
"QUARTER"	T_QUARTER
"REFERENCES"	T_REFERENCES
"RESTRICT"	T_RESTRICT
"RETURN"	T_RETURN
"RETURNING"	T_RETURNING
"REVOKE"	T_REVOKE
"RIGHT"	T_RIGHT
"SCHEMA"	T_SCHEMA
"SECOND"	T_SECOND
"SELECT"	T_SELECT
"SET"	T_SET
"STDDEV"	T_STDDEV
"STDDEV_POP"	T_STDDEV_POP

Identifier	Token
"SUM"	T_SUM
"SVEBEGIN"	T_SVBEGIN
"SVEND"	T_SVEND
"TABLE"	T_TABLE
"THEN"	T_THEN
"TIME"	T_TIME
"TIMESTAMP"	T_TIMESTAMP
"TIMESTAMPADD"	T_TIMESTAMPADD
"TIMESTAMPDIFF"	T_TIMESTAMPDIFF
"TO"	T_TO
"TOP"	T_TOP
"UNION"	T_UNION
"UNIQUE"	T_UNIQUE
"UNPIVOT"	T_UNPIVOT
"UPDATE"	T_UPDATE
"UPSERT"	T_UPSERT
"USAGE"	T_USAGE
"USER"	T_USER

Identifier	Token
"USING"	T_USING
"VALUES"	T_VALUES
"VAR"	T_VAR
"VAR_POP"	T_VAR_POP
"VIEW"	T_VIEW
"WEEK"	T_WEEK
"WHEN"	T_WHEN
"WHERE"	T_WHERE
"WITH"	T_WITH
"YEAR"	T_YEAR
{" "}	T_CONCAT
"="	T_EQ
">="	T_GE
">"	T_GT
"<"	T_LT
"<="	T_LE
"<>" "!="	T_NE

T_LIKE

(l|L) (i|I) (k|K) (e|E)

T_NOT

```
(n|N) (o|O) (t|T)
```

T_NOTLIKE

```
{Not}{white}+{Like}
```

T_TSIDATATYPE

```
(S|s) (Q|q) (L|l) "_" (T|t) (S|s) (I|i) "_" ({letter}|"_")*
```

T_USINT_LIT

```
{digit}+
```

Reference

1. BNF for SQL-92: <http://savage.net.au/SQL/sql-92.bnf.html#subquery>
2. Credit to Syntax diagrams for BNF generate tool: <http://bottlecaps.de/rr/ui>

Contact Us

For more information or help using this product, please contact our Technical Support staff. We welcome your questions, comments, and feature requests.

Note:

To help us assist you, prior to contacting Technical Support please prepare a detailed summary of the Simba C++ SQL Engine version and development platform that you are using.

You can contact Technical Support at www.insightsoftware.com.

Third-Party Trademarks

Simba, the Simba logo, SimbaEngine, Simba SDK, and Simba Technologies are registered trademarks of Simba Technologies Inc. in Canada, United States and/or other countries. All other trademarks and/or servicemarks are the property of their respective owners.

All other trademarks are trademarks of their respective owners.