```
//ver3.1
// ########### Syntax Rules ###########

// Start Symbol
start
  : ( getCollection             // 1
    | setIntermediateAs         // 2
    | saveAs                    // 3
    | spatialJoin               // 4
    | joinOfCollections         // 5
    | filter                    // 6
    | group                     // 7
    | expand                    // 8
    | mergeCollections          // 9
    | intersectCollections      // 10
    | subtractCollections       // 11
    | useDb                     // 12
    | trajectoryMatching        // 13
    | createFuzzyOperator       // 14
    | createJavaScriptFunction  // 15
    )* EOF
  ;

//--------------------------------------------------


collectionReference
  :
    ID  ( AT ID )?  ( AS ID )?
  ;


fieldRef
  :
    ( FIELD_NAME  )+
  ;


value
  :
      INT
    | FLOAT
    | APEX_VALUE
    | QUOTED_VALUE
    | BOOLEAN
  ;


outputFieldSpec
  :
    fieldRef
    COLON ( value
          | fieldRef
          | objectStructure
          )
  ;


parameter
  :
    ID TYPE ID
  ;


generateAction
  :
    GENERATE
      ( objectStructure   ( geometricOption )?
      | geometricOption
      | COUNT LP fieldRef RP
      )
  ;


objectStructure
  :
    LBR
      outputFieldSpec ( COMMA outputFieldSpec )*
    RBR
  ;


geometricOption
  :
    KEEPING GEOMETRY
  | DROPPING GEOMETRY
  | SETTING GEOMETRY
            ( POINT LP fieldRef COMMA fieldRef RP
            | AGGREGATE LP fieldRef RP
            | fieldRef
            | TO_POLYLINE LP fieldRef RP
            )
  ;


caseClause
  :
    CASE
        ( whereCase )+
        others
  ;


others
  :
      KEEP OTHERS | DROP OTHERS
  ;


whereCase
  :
    WHERE
      orCondition
      ( generateAction )?
      ( fuzzyCheck )*
      ( alphaCut )*
      ( keepDropFuzzySets )?
  ;


orCondition
  :
    andCondition  ( OR andCondition )*
  ;


andCondition
  :
    notCondition  ( AND notCondition )*
  ;


notCondition
  :
    ( NOT )?  predicate
  ;
```

```
predicate
  :
      expression ( comparator expression )?
    | withPredicate
    | withoutPredicate
    | ( WITHIN | KNOWN | UNKNOWN ) FUZZY SETS ID ( COMMA ID )*
    | ifFails
    | OVERLAP LP RP
    | INSIDE LP (LEFT|RIGHT) RP
    | HOWMEET LP (LEFT|RIGHT) RP
  ;


withPredicate
  :
    WITH (ID|ARRAY)? fieldRef ( COMMA fieldRef )*
  ;


withoutPredicate
  :
    WITHOUT fieldRef  ( COMMA fieldRef )*
  ;


expression
  :
    (ADD |SUB)? term ( (ADD|SUB) term )*
  ;


term
  :
    factor ( (MUL|DIV) factor )*
  ;


factor
  :
      fieldRef
    | LP orCondition RP
    | INT
    | FLOAT
    | APEX_VALUE
    | QUOTED_VALUE
    | ID  ( LP (functionParams)? RP )?
  ;


functionParams
  :
    expression ( COMMA expression )*
  ;


comparator
  :
    ( EQ | NEQ | LT | GT | LE | GE )
  ;


// token arricchito con il segno
numeric
  :
    ( ADD | SUB )? ( FLOAT | INT )
  ;
```

```
//--- fuzzy part
fuzzyCheck
  :
    CHECK_FOR FUZZY SET ID
      USING orCondition
  ;


alphaCut
  :
    ALPHACUT numeric ON ID
  ;


keepDropFuzzySets
  :
      DROPPING ALL FUZZY SETS
    | KEEPING ALL FUZZY SETS
    | DROPPING FUZZY SETS ID ( COMMA ID )*
    | KEEPING FUZZY SETS ID ( COMMA ID )*
  ;


addFields
  :
    ADD_ST FIELDS
      nonFuzzyFunction AS fieldRef
      ( COMMA nonFuzzyFunction AS fieldRef )*
  ;



fuzzySetReference
  :
       ID ( AS ID )?
    | RIGHT LP ID RP ( AS ID )?
    | LEFT  LP ID RP ( AS ID )?
    | ( INSIDE LP ( LEFT | RIGHT ) RP
      | OVERLAP LP RP
      | HOWMEET LP ( LEFT | RIGHT ) RP
      )
      AS ID
  ;


nonFuzzyFucntion
  :
      DISTANCE LP ID RP ( comparator numeric )?
    | AREA LP ID RP ( comparator numeric )?
    | ORIENTATION LP ( LEFT | RIGHT ) ( COMMA  ID COLON numeric )? RP
    | INCLUDED LP ( LEFT | RIGHT) RP
    | MEET
    | INTERSECT
  ;


setKeepDropFuzzySets
  :
      KEEP (ALL | LEFT | RIGHT) SOURCE FUZZY SETS
    | DROP SOURCE FUZZY SETS
  ;


addNewFunzzySets ]
  :
    ADD_ST NEW FUZZY SETS
```

```
      fuzzySetReference
        ( COMMA fsr=fuzzySetReference )*
  ;


ifFails
  :
    IFFAILS LP orIffCondition COMMA numeric RP
  ;


orIffCondition
  :
    andIffCondition ( OR andIffCondition  )*
  ;


andIffCondition
  :
    notIffCondition ( AND notIffCondition )*
  ;


notIffCondition
  :
    (NOT)? predicate
  ;




// -------------- Basic operators ----------------


getCollection
  :
    GET COLLECTION
      ID ( AT ID )?
    SC
  ;


setIntermediateAs
  :
    SET INTERMEDIATE AS
      ID
    SC
  ;


saveAs
  :
    SAVE AS
      ID AT ID
    SC
  ;


spatialJoin
  :
    SPATIAL JOIN OF COLLECTIONS
      collectionReference COMMA collectionReference
      ( ON nonFuzzyFucntion )?
      SET GEOMETRY ( INTERSECTION | RIGHT | LEFT | ALL )
      ( addFields )?
      ( setKeepDropFuzzySets )?
      ( addNewFunzzySets  )?
      ( caseClause )?
    SC
  ;


joinOfCollections
  :
    JOIN OF COLLECTIONS
      collectionReference COMMA collectionReference
      ( addFields )?
      ( setKeepDropFuzzySets )?
      ( addNewFunzzySets  )?
      ( caseClause )?
    SC
  ;


filter
  :
    FILTER
        caseClause
    SC
  ;


group
  :
    GROUP
      ( groupPartition )+
      others
    SC
  ;


groupPartition
 :
    PARTITION orCondition
      BY fieldRef( COMMA fieldRef )*
      INTO fieldRef ( DROP GROUPING FIELDS )?
      ( ORDER BY  fieldRef ( VERSUS )? ( COMMA fieldRef ( VERSUS )? )*  )?
      ( generateAction )?
  ;


expand
  :
    EXPAND
      ( unpack )+
      others
    SC
  ;


unpack
  :
    UNPACK  orCondition
      ARRAY fieldRef
      TO ID
      ( generateAction )?
  ;


mergeCollections
  :
    ( ALL )? MERGE COLLECTIONS
        collectionReference ( COMMA collectionReference )+
    SC
  ;
```

```
intersectCollections
  :
    INTERSECT COLLECTIONS
      collectionReference COMMA collectionReference
    SC
  ;


subtractCollections
  :
    SUBTRACT COLLECTIONS
      collectionReference COMMA collectionReference
    SC
  ;


useDb
  :
    USE
      DB (ID | APEX_VALUE) (AS (ID | APEX_VALUE) )?
              ( COMMA DB (ID | APEX_VALUE) ( AS (ID | APEX_VALUE) )? )*
      ON
        ( DEFAULT SERVER
        | SERVER (ID | APEX_VALUE) ( (ID | APEX_VALUE) )?
        )
    SC
  ;


trajectoryMatching
  :
    TRAJECTORY MATCHING
      collectionReference COMMA collectionReference
      ( trajectoryPartition )+
      others
    SC
  ;


trajectoryPartition
  :
    PARTITION
      orCondition
      ( partitionMatching )+
  ;


partitionMatching
  :
    MATCHING fieldRef
      WRT fieldRef
      THRESHOLD LP  ID  RP numeric
      ( WHERE orCondition )?
      INTO fieldRef
      ( ADDING fieldRef TO INPUT )?
      ( MIN SIMILARITY numeric )?
  ;


createFuzzyOperator
  :
    CREATE_FO ID
      PARAMETERS parameter ( COMMA parameter )*
      PRECONDITION orCondition
      EVALUATE expression
      RANGE LP numeric COMMA numeric RP
      POLYLINE LP Numeric COMMA numeric RP ( COMMA LP numeric COMMA numeric
      RP )*
    SC
  ;


createJavaScriptFunction
  :
    CREATE_JF ID
      PARAMETERS parameter ( COMMA parameter )*
      PRECONDITION orCondition
      BODY [...] END_BODY
    SC
  ;



// ************************************
// ***
// ***            SCANNER
// ***
// ************************************

fragment LETTER : 'A'..'Z'|'a'..'z';
fragment DIGIT0 : '1'..'9';
fragment DIGIT  : '0'..'9';
fragment WS     : ( ' ' | '\t' | '\r' | '\n' )+  ;

// boolean Operator
AND : 'AND';
OR  : 'OR';
NOT : 'NOT';

// keywords

ADDING        : 'ADDING';
ADD_ST        : 'ADD';
AGGREGATE     : 'AGGREGATE';
ALL           : 'ALL';
ALPHACUT      : 'ALPHA-CUT';
AREA          : 'AREA';
ARRAY         : 'ARRAY';
AS            : 'AS';
BODY          : 'BODY';
BOOLEAN       : 'TRUE' | 'FALSE';
BY            : 'BY';
CASE          : 'CASE';
COLLECTION    : 'COLLECTION';
COLLECTIONS   : 'COLLECTIONS';
CREATE_FO     : 'CREATE' WS 'FUZZY' WS 'OPERATOR';
CREATE_JF     : 'CREATE' WS 'JAVASCRIPT' WS 'FUNCTION';
COUNT         : 'COUNT';
DB            : 'DB';
DEFAULT       : 'DEFAULT';
DIRECTION     : 'DIRECTION';
DISTANCE      : 'DISTANCE';
DROP          : 'DROP';
DROPPING      : 'DROPPING';
END_BODY      : 'END' WS 'BODY';
EXPAND        : 'EXPAND';
EVALUATE      : 'EVALUATE';
FIELDS        : 'FIELDS';
FILTER        : 'FILTER';
FUZZY         : 'FUZZY';
GENERATE      : 'GENERATE';
GEOMETRY      : 'GEOMETRY';
GET           : 'GET';
GROUP         : 'GROUP';
```

```
GROUPING      : 'GROUPING';
HOWMEET       : 'HOW-MEET';
IFFAILS       : 'IF-FAILS';
INCLUDED      : 'INCLUDED';
INPUT         : 'INPUT';
INSIDE        : 'INSIDE';
INTERMEDIATE  : 'INTERMEDIATE';
INTERSECT     : 'INTERSECT';
INTERSECTION  : 'INTERSECTION';
INTO          : 'INTO';
JOIN          : 'JOIN';
KEEP          : 'KEEP';
KEEPING       : 'KEEPING';
KNOWN         : 'KNOWN';
LEFT          : 'LEFT';
MATCHING      : 'MATCHING';
MEET          : 'MEET';
MERGE         : 'MERGE';
MIN           : 'MIN';
OF            : 'OF';
ON            : 'ON';
ORIENTATION   : 'ORIENTATION';
OTHERS        : 'OTHERS';
ORDER         : 'ORDER' | 'SORTED';
OVERLAP       : 'OVERLAP';
PARAMETERS    : 'PARAMETERS';
PARTITION     : 'PARTITION';
PERIMETER     : 'PERIMETER';
POINT         : 'POINT';
POLYLINE      : 'POLYLINE';
PRECONDITION  : 'PRECONDITION';
RANGE         : 'RANGE';
RIGHT         : 'RIGHT';
SAVE          : 'SAVE';
SERVER        : 'SERVER';
SET           : 'SET';
SETS          : 'SETS';
SETTING       : 'SETTING';
SIMILARITY    : 'SIMILARITY';
SOURCE        : 'SOURCE' ;
SPATIAL       : 'SPATIAL';
SUBTRACT      : 'SUBTRACT';
TO            : 'TO';
TO_POLYLINE   : 'TO_POLYLINE';
TRAJECTORY    : 'TRAJECTORY';
THRESHOLD     : 'THRESHOLD';
TYPE          : 'TYPE';
UNKNOWN       : 'UNKNOWN';
UNPACK        : 'UNPACK';
USE           : 'USE';
USING         : 'USING';
VERSUS        : 'DESC' | 'ASC';
WHERE         : 'WHERE';
WITH          : 'WITH';
WITHIN        : 'WITHIN';
WITHOUT       : 'WITHOUT';
WRT           : 'WRT';

INT: '0' | DIGIT0 DIGIT* ;
FLOAT: DIGIT0 DIGIT* DOT DIGIT+ | '0' DOT DIGIT+;

ID: LETTER (LETTER | DIGIT | '_')*;
ID2: (LETTER | DIGIT | '_')+;

FIELD_NAME: ( DOT (LETTER | DIGIT | '_')+ )
            | DOT '"' (~('"') )* '"'
            | DOT '~geometry'
            | '~geometry';
```

```
// puntuaction
AT    : '@';
EQ    : '=';
NEQ   : '!=';
LE    : '<=';
GE    : '>=';
LT    : '<';
GT    : '>';
DOT   : '.';
ADD   : '+';
SUB   : '-';
MUL   : '*';
DIV   : '\\';
COMMA : ',';
COLON : ':';
SC    : ';';
LP    : '(';
RP    :  ')';
LB    : '[';
RB    : ']';
LBR   : '{';
RBR   : '}';
APEX  : '\'';
QUOTE : '"';
SLASH : '/';
TILDE : '~';
XXX   : '###TEST***';

WHITE_SPACES  : WS ;
APEX_VALUE    : '\'' (~('\'') )* '\'' ;
QUOTED_VALUE  : '"' (~('"') )* '"' ;
SCAN_ERROR    : . ;
```