Universidad de San Carlos de Guatemala

Facultad de ingeniería

Escuela de ciencias y sistemas

Curso de Organización de Lenguajes y Compiladores 2

PROYECTO: FASE 1

Manual técnico

Grupo 7

201314064 ROMARIO DAVID CASTILLO ECHEVERRIA

201314177 ROBERTO EDUARDO CASEROS REYNOSO

201314448 CARLOS ENRIQUE CANTÉ LÓPEZ

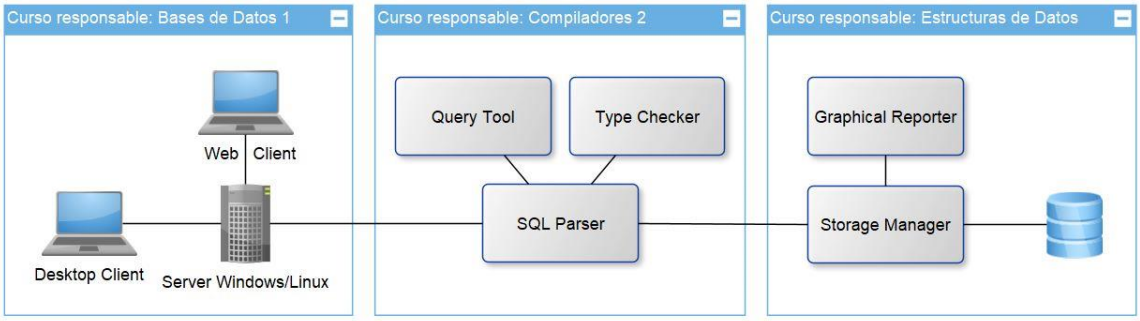
201314556 CARLOS GABRIEL PERALTA CAMBRAN

DESCRIPCION

El programa realiza la validación de sentencias con sintaxis SQL basada en la utilizada por POSTGRESQL, esto lo hace mediante la utilización de la herramienta para la construcción de analizadores PLY.

El programa es parte de de un sistema cuyo funcionamiento es el de un DBMS escrito en el lenguaje Python con todas sus funciones básicas

ESTRUCTURA GENERAL DEL PROYECTO



INTERACCION CON EL ADMINISTRADOR DEL ALAMCENAMIENTO

El programa hace uso de una serie de funciones que son proveidas por el sistema administrador del almacenamiento el cual se encarga de guardar y proveer todos los datos que se manejen por medio del DBMS, dicas funciones se detallan a continuación:

def createDatabase(database: str) -> int:

Crea una base de datos. (CREATE)

Parámetro database: es el nombre de la base de datos, debe cumplir con las reglas de identificadores de SQL.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 base de datos existente

def showDatabases() -> list:

Devuelve una lista de los nombres de las bases de datos. (READ)

Valor de retorno: lista de strings con los nombres de las bases de datos, si ocurrió un error o no hay bases de datos devuelve una lista vacía [].

def alterDatabase(databaseOld, databaseNew) -> int:

Renombra la base de datos databaseOld por databaseNew. (UPDATE)

Parámetro databaseOld: es el nombre actual de la base de datos, debe cumplir con las reglas de identificadores de SQL.

Parámetro databaseNew: es el nuevo nombre que tendrá de la base de datos databaseOld, debe cumplir con las reglas de identificadores de SQL.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 databaseOld no existente, 3 databaseNew existente.

def dropDatabase(database: str) -> int:

Elimina por completo la base de datos indicada en database. (DELETE)

Parámetro database: es el nombre de la base de datos que se desea eliminar, debe cumplir con las reglas de identificadores de SQL.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 base de datos no existente.

Respecto de las funciones CRUD de las tablas están:

def createTable(database: str, table: str, numberColumns: int) -> int:

Crea una tabla en una base de datos especificada recibiendo una lista de índices referentes a la llave primaria y llave foránea. (CREATE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla que se desea crear.

Parámetro numberColumns: es el número de columnas que tendrá cada registro de la tabla.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 base de datos inexistente, 3 tabla existente.

def showTables(database: str) -> list:

Devuelve una lista de los nombres de las tablas de una base de datos. (READ)

Parámetro database: es el nombre de la base de datos a utilizar.

Valor de retorno: si existen la base de datos y las tablas devuelve una lista de nombres de tablas; si existe la base de datos, pero no existen tablas devuelve una lista vacía; y si no existe la base de datos devuelve None.

def extractTable(database: str, table: str) -> list:

Extrae y devuelve una lista con elementos que corresponden a cada registro de la tabla. (READ)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Valor de retorno: si existe la base de datos, la tabla y los registros devuelve una lista con los registros, si existen las base de datos, la tablas pero no registros devuelve una lista vacía, y si ocurre un error o si no existe la base de datos o la tabla devuelve None.

def extractRangeTable(database: str, table: str, columnNumber: int, lower: any, upper: any) -> list:

Extrae y devuelve una lista con los elementos que corresponden a un rango de registros de la tabla. (READ)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Parámetro columnNumber: es el número de índice de columna a restringir o verificar con los valores upper y lower. Parámetro lower: es el límite inferior (inclusive) del rango a extraer de la columna indicada de la tabla.

Parámetro upper: es el límite superior (inclusive) del rango a extraer de la columna indicada de la tabla.

Valor de retorno: si existe la base de datos, la tabla y los registros devuelve una lista con los registros(lista), si existen las base de datos, la tablas pero no registros devuelve una lista vacía, y si no existe la base de datos o la tabla o cualquier error devuelve None.

def alterAddFK(database: str, table: str, references: dict) -> int:

Asocia la integridad referencial entre llaves foráneas y llaves primarias, para efectos de la fase 1 se ignora esta petición. Debido a que será parte de la fase 2 en la construcción de índices secundarios. (UPDATE PENDIENTE)

def alterAddIndex(database: str, table: str, references: dict) -> int:

Asocia un índice, para efectos de la fase 1 se ignora esta petición. Debido a que será parte de la fase 2 en la construcción de índices secundarios. (UPDATE PENDIENTE)

def alterTable(database: str, tableOld: str, tableNew: str) -> int:

Renombra el nombre de la tabla de una base de datos especificada. (UPDATE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro tableOld: es el nombre de la tabla a renombrar.

Parámetro tableNew: es el nuevo nombre con que renombrará la tableOld.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 tableOld no existente, 4 tableNew existente.

def alterAddColumn(database: str, table: str, default: any) -> int:

Agrega una columna al final de cada registro de la tabla y base de datos especificada. (UPDATE) Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a modificar.

Parámetro default: es el valor que se establecerá en a la nueva columna para los registros existentes.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente.

def alterDropColumn(database: str, table: str, columnNumber: int) -> int:

Eliminar una n-ésima columna de cada registro de la tabla excepto si son llaves primarias. (DELETE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a modificar.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente, 4 llave no puede eliminarse o tabla quedarse sin columnas, 5 columna fuera de límites.

def dropTable(database: str, table: str) -> int:

Elimina por completo una tabla de una base de datos especificada. (DELETE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a eliminar.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente.

Respecto de las funciones CRUD de las tuplas están:

def insert(database: str, table: str, register: list) -> int:

Inserta un registro en la estructura de datos asociada a la tabla y la base de datos. (CREATE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Parámetro register: es una lista de elementos que representan un registro.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente, 4 llave primaria duplicada, 5 columnas fuera de límites.

def loadCSV(file: str, database: str, table: str) -> list:

Carga un archivo CSV de una ruta especificada indicando la base de datos y tabla donde será almacenado. La base de datos y la tabla deben existir, y coincidir con el número de columnas. Si hay llaves primarias duplicadas se ignoran. No se utilizan títulos de columnas y la separación es por comas. (CREATE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Valor de retorno: lista con los valores enteros que devuelve el insert por cada fila del CSV, si ocurrió un error o el archivo CSV no tiene filas devuelve una lista vacía [].

def extractRow(database: str, table: str, columns: list) -> list:

Extrae y devuelve un registro especificado por su llave primaria. (READ)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Parámeto columns: es la llave primaria, si es simple [llave], si es compuesta [llaveatr1, llaveatr2...]. (si no hay pk se debe enviar la hiddenPK)

Valor de retorno: lista con los valores del registro, si ocurrió un error o no hay registro que mostrar devuelve una lista vacía [].

def update(database: str, table: str, register: dict, columns: list) -> int:

Inserta un registro en la estructura de datos asociada a la tabla y la base de datos. (UPDATE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Parámetro register: es una lista de elementos llave:valor que representa los elementos a actualizar del registro. La llave el número de coluna y el valor el contenido del campo.

Parámetro columns: es la llave primaria, si es simple [llave], si es compuesta [llaveatr1, llaveatr2...]. (si no hay pk se debe enviar la hiddenPK) Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente, 4 llave primaria no existe.

def delete(database: str, table: str, columns: list) -> int:

Elimina un registro de una tabla y base de datos especificados por la llave primaria. (DELETE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Parámetro columns: es la llave primaria, si es simple [llave], si es compuesta [llaveatr1, llaveatr2...]. (si no hay pk se debe enviar la hiddenPK)

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente, 4 llave primaria no existe.

def truncate(database: str, table: str) -> int:

Elimina todos los registros de una tabla y base de datos. (DELETE)

Parámetro database: es el nombre de la base de datos a utilizar.

Parámetro table: es el nombre de la tabla a utilizar.

Valor de retorno: 0 operación exitosa, 1 error en la operación, 2 database no existente, 3 table no existente.

GRAMATICA UTILIZADA

Debido a que la herramienta de PLY es un analizador que se ejecuta de manera ascendente se decidió utilizar una gramática ascendente ya que esto mejora el funcionamiento de la herramienta además de reducir la complejidad del código utilizado.

La gramática se utilizada se describe a continuación:

###Palabras Reservadas

- SMALLINT
- INTEGER
- BIGINIT
- DECIMAL
- NUMERIC
- REAL
- DOUBLE
- PRECISION
- MONEY
- VARCHAR
- CHARACTER
- TEXT
- TIMESTAMP
- WITHOUT
- TIME
- ZONE
- WITH
- DATE
- INTERVAL
- YEAR
- MONTH
- DAY
- HOUR
- MINUTE
- SECOND
- TO
- BOOLEAN
- CREATE
- TYPE
- AS
- ENUM
- BETWEEN
- IN
- LIKE
- ILIKE
- SIMILAR
- ISNULL
- NOTNULL
- NOT
- NULL
- AND
- OR
- REPLACE
- DATABASE
- IF
- EXISTS
- OWNER
- MODE
- SHOW
- DATABASES
- ALTER
- RENAME
- DROP
- TABLE
- CONSTRAINT
- UNIQUE
- CHECK
- PRIMARY
- KEY
- REFERENCES
- FOREIGN
- ADD
- SET
- DELETE
- FROM
- WHERE
- INHERITS
- INSERT
- INTO
- UPDATE
- VALUES
- SELECT
- DISTINCT
- GROUP
- BY
- HAVING
- SUM
- COUNT
- AVG
- MAX
- MIN
- ABS
- CBRT
- CEIL
- CEILING
- DEGREES
- DIV
- EXP
- FACTORIAL
- FLOOR
- GCD
- LCM
- LN
- LOG
- LOG10
- MIN_SCALE
- MOD
- PI
- POWER

- RADIANS
- ROUND
- SCALE
- SIGN
- SQRT
- TRIM_SCALE
- WIDTH_BUCKET
- RANDOM
- SETSEED
- ACOS
- ACOSD
- ASIN
- ASIND
- ATAN
- ATAND
- ATAN2
- ATAN2D
- COS
- COSD
- COT
- COTD
- SIN
- SIND
- TAN
- TAND
- SINH
- COSH
- TANH
- ASINH
- ACOSH
- ATANH
- LENGTH
- SUBSTRING
- TRIM
- GET_BYTE
- MD5
- SET_BYTE
- SHA256
- SUBSTR
- CONVERT
- ENCODE
- DECODE
- EXTRACT
- CENTURY
- DECADE
- DOW
- DOY
- EPOCH
- ISODOWN
- ISOYEAR
- MICROSECONDS
- MILENNIUM
- MILLISECONDS
- QUARTER
- TIMEZONE
- TIMEZONE_HOUR
- TIMEZONE_MINUTE
- WEEK
- AT
- CURRENT_DATE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- LOCALTIME
- LOCALTIMESTAMP
- PG_SLEEP
- PG_SLEEP_FOR
- PG_SLEEP_UNTIL
- INNER
- LEFT
- RIGHT
- FULL
- OUTER
- JOIN
- ALL
- ANY
- SOME
- ORDER
- ASC
- DESC
- CASE
- WHEN
- THEN
- ELSE
- END
- GREATEST
- LEAST
- LIMIT
- UNION
- INTERSECT
- EXCEPT
- IS
- DEFAULT
- TRUE
- FALSE
- COLUMN
- CURRENT_USER
- SESSION_USER
- DATE_PART
- NOW
- TRUNC
- OFFSET
- NULLS
- FIRST
- LAST
- CHAR

TOKENS

- PTCOMA
- COMA
- PUNTO
- TYPECAST
- MAS
- MENOS
- POTENCIA
- MULTIPLICACION
- DIVISION
- MODULO
- MENOR_QUE
- MENOR_IGUAL
- MAYOR_QUE
- MAYOR_IGUAL
- IGUAL
- DISTINTO
- LLAVEIZQ
- LLAVEDER
- PARIZQUIERDO
- PARDERECHO
- DECIMAL_
- ENTERO
- CADENA
- ID
- ESPACIO

#GRAMATICA

```
<init>          ::=   <instrucciones>

<instrucciones>    ::=   <instrucciones> <instruccion>
                |   <intruccion>


<instruccion>       ::=   <insert_table>
                |  <delete_table>
                |  <update_table>
                |    <crear_instr>
                |    <alter_instr>
                |    <drop_instr>
                |    <inst_select> PTCOMA
```

<crear_instr>	::= CREATE TABLE ID PARIZQUIERDO <columnas> PARDERECHO <herencia> PTCOMA
	| CREATE <opReplace> DATABASE <opExists> ID <opDatabase> PTCOMA
	| CREATE TYPE ID AS ENUM PARIZQUIERDO ID PARDERECHO PTCOMA

<insert_table>	::= INSERT INTO ID VALUES <lista_valores> PTCOMA
	| INSERT INTO ID PARIZQUIERDO <lista_columnas> PARDERECHO VALUES <lista_valores> PTCOMA
	| INSERT INTO ID DEFAULT VALUES PTCOMA
	| INSERT INTO ID PARIZQUIERDO <lista_columnas> PARDERECHO DEFAULT VALUES PTCOMA

<lista_columnas>	::= <lista_columnas> COMA ID
	| ID

<lista_valores>	::= <lista_valores> COMA <tupla>
	| <tupla>

<tupla>	::= PARIZQUIERDO <lista_expresiones> PARDERECHO

<lista_expresiones>	::= <lista_expresiones> COMA <expresion>
	| <expresion>

<expresion>	::= CADENA
	| ENTERO
	| DECIMAL_

<delete_table>	::= DELETE FROM ID PTCOMA
	| DELETE FROM ID WHERE <exp_operacion> PTCOMA

<exp_operacion>	::= <exp_logica>

<exp_logica>	::= <exp_logica> OR <exp_logica>
	| <exp_logica> AND <exp_logica>
	| NOT <exp_logica>
	| <exp_relacional>

<exp_relaciona>	::= <exp_relacional> MENOR_QUE <exp_relacional>
	| <exp_relacional> MENOR_IGUAL <exp_relacional>
	| <exp_relacional> MAYOR_QUE <exp_relacional>
	| <exp_relacional> MAYOR_IGUAL <exp_relacional>
	| <exp_relacional> DISTINTO <exp_relacional>
	| <exp_relacional> IGUAL <exp_relacional>
	| <exp_aritmetica>

<exp_aritmetica> ::= <exp_aritmetica> MAS <exp_aritmetica>
| <exp_aritmetica> MENOS <exp_aritmetica>
| <exp_aritmetica> MULTIPLICACION <exp_aritmetica>
| <exp_aritmetica> DIVISION <exp_aritmetica>
| <exp_aritmetica> MODULO <exp_aritmetica>
| <exp_aritmetica> POTENCIA <exp_aritmetica>
| <exp_aritmetica> BETWEEN <exp_aritmetica> AND <exp_aritmetica>
| <exp_aritmetica> NOT BETWEEN <exp_aritmetica> AND <exp_aritmetica>
| <exp_aritmetica> IN PARIZQUIERDO <lista_expresiones> PARDERECHO
| <exp_aritmetica> NOT IN PARIZQUIERDO <lista_expresiones> PARDERECHO
| <exp_aritmetica> IN <subquery>
| <exp_aritmetica> NOT IN <subquery>
| <exp_aritmetica> LIKE <exp_aritmetica>
| <exp_aritmetica> NOT LIKE <exp_aritmetica>
| <exp_aritmetica> ILIKE <exp_aritmetica>
| <exp_aritmetica> NOT ILIKE <exp_aritmetica>
| <exp_aritmetica> SIMILAR TO <exp_aritmetica>
| <exp_aritmetica> IS NULL
| <exp_aritmetica> IS NOT NULL
| <primitivo>

<primitivo> ::= ID
| ID PUNTO ID
| MAS <primitivo>
| MENOS <primitivo>
| PARIZQUIERDO <exp_operacion> PARDERECHO
| ENTERO
| DECIMAL_
| CADENA
| TRUE
| FALSE
| <funcion>

<update_table> ::= UPDATE ID SET <lista_seteos> PTCOMA
| UPDATE ID SET <lista_seteos> WHERE <exp_operacion> PTCOMA

<lista_seteos> ::= <lista_seteos> COMA <set_columna>
| <set_columna>

<set_columna> ::= ID IGUAL <exp_operacion>

<columnas> ::= <columnas> COMA <columna>
| <columna>

<columna> ::= ID <tipos> <opcional>
| PRIMARY KEY PARIZQUIERDO <identificadores> PARDERECHO
| FOREIGN KEY PARIZQUIERDO <identificadores> PARDERECHO REFERENCES ID PARIZQUIERDO
<identificadores> PARDERECHO

```
            |   UNIQUE PARIZQUIERDO <identificadores> PARDERECHO


<opcional>          ::=   DEFAULT <opcionNull>
            |   <opcionNull>


<opcionNull>       ::=   NOT NULL <opConstraint>
            |   <opConstraint>


<opConstraint>       ::= CONSTRAINT ID <opUniqueCheck>
            |    <opUniqueCheck>


<opUniqueCheck>    ::=   UNIQUE
            |    CHECK PARIZQUIERDO <condicion_check> PARDERECHO
            |    empty


<condicion_check>    ::=   ID MENOR_QUE <expresion>
            |    ID MENOR_IGUAL <expresion>
            |    ID MAYOR_QUE <expresion>
            |    ID MAYOR_IGUAL <expresion>
            |    ID DISTINTO <expresion>
            |    ID IGUAL <expresion>


<herencia>          ::= INHERITS PARIZQUIERDO ID PARDERECHO'
            |   empty


<identificadores>    ::= <identificadores> COMA ID
            |   ID

<opReplace>       ::= OR REPLACE'
            |   empty


<opExists>          ::= IF NOT EXISTS
            |   empty


<opDatabase>       ::= OWNER <opIgual> ID <mode>
            |   mode


<opIgual>          ::= IGUAL
            |   empty


<mode>             ::= MODE <opIgual> ENTERO
            |    empty
```

```
<alter_instr>       ::= ALTER DATABASE ID <opAlterDatabase> PTCOMA
                |   ALTER TABLE ID <alter_table_instr> PTCOMA


<opAlterDatabase>    ::= RENAME TO ID
                |    OWNER TO <ownerList>

<ownerList>       ::= ID
                |   CURRENT_USER
                |   SESSION_USER


<alter_table_instr>   ::= ADD <add_instr>
                |   <alter_columnas>
                |   <drop_columnas>


<alter_columnas>    ::= <alter_columnas> COMA <alter_columna>
                |   <alter_columna>


<alter_columna>    ::= ALTER COLUMN ID <alter_column_instr>


<drop_columnas>    ::= <drop_columnas> COMA <drop_columna>
                |   <drop_columna>


<drop_columna>       ::= DROP COLUMN ID


<alter_table_instr> ::= DROP CONSTRAINT ID
                |    SET NOT NULL
                |   SET NULL
                |   TYPE ID


<add_instr>       ::= CHECK PARIZQUIERDO <condicion_check> PARDERECHO
                |   CONSTRAINT ID UNIQUE PARIZQUIERDO ID PARDERECHO


<drop_instr>       ::= DROP DATABASE <si_existe> ID PTCOMA
                |   DROP TABLE ID PTCOMA

<si_existe>       ::= IF EXISTS
                |   empty

<inst_select>       ::=   <select_query>
                |   <select_query> UNION <select_query>
                |   <select_query> UNION ALL <select_query>
                |   <select_query> INTERSECT <select_query>
                |   <select_query> INTERSECT ALL <select_query>
                |   <select_query> EXCEPT <select_query>
```

```
            |   <select_query> EXCEPT ALL <select_query>


<select_query>      ::=   SELECT DISTINCT <select_list> FROM <from_query_list> <lista_condiciones_query>
            |   SELECT <select_list> FROM <from_query_list> <lista_condiciones_query>
            |   SELECT DISTINCT <select_list> FROM <from_query_list>
            |   SELECT <select_list> FROM <from_query_list>
            |   SELECT <select_list>


<select_list>       ::=   MULTIPLICACION
            |   <elementos_select_list>


<elementos_select_list>   ::=   <elementos_select_list> COMA <elemento_select>
                |   <elemento_select>


<elemento_select>     ::=   <dec_select_columna>
            |    <subquery> AS ID
            |   <subquery> ID
            |   <subquery>
            |    <funcion> AS ID
            |   <funcion> ID
            |   <funcion>

<dec_select_columna>::=   ID PUNTO ID AS ID
              |   ID PUNTO ID ID
              |   ID PUNTO ID
            |   ID


<funcion>           ::=   <funcion_time>
            |   <funcion_mate>
            |   <funcion_trig>
            |   <funcion_binstr>
            |   <funcion_exprecion>
            |   <funcion_agregacion>
            |   <dec_case>

<funcion_time>      ::=   EXTRACT PARIZQUIERDO <var_time> FROM <var_timeextract> CADENA PARDERECHO
            |   DATE_PART PARIZQUIERDO CADENA COMA <var_timeextract> CADENA PARDERECHO
            |   NOW PARIZQUIERDO PARDERECHO
            |   CURRENT_DATE
            |   CURRENT_TIME


<var_time>          ::=   YEAR
            |   MONTH
            |   DAY
            |   HOUR
            |   MINUTE
            |   SECOND
```

<var_timeextract> ::= TIMESTAMP
| TIME
| DATE
| INTERVAL

<funcion_mate> ::= ABS PARIZQUIERDO <exp_operacion> PARDERECHO
| CBRT PARIZQUIERDO <exp_operacion> PARDERECHO
| CEIL PARIZQUIERDO <exp_operacion> PARDERECHO
| CEILING PARIZQUIERDO <exp_operacion> PARDERECHO
| DEGREES PARIZQUIERDO <exp_operacion> PARDERECHO
| DIV PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| EXP PARIZQUIERDO <exp_operacion> PARDERECHO
| FACTORIAL PARIZQUIERDO <exp_operacion> PARDERECHO
| FLOOR PARIZQUIERDO <exp_operacion> PARDERECHO
| GCD PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| LN PARIZQUIERDO <exp_operacion> PARDERECHO
| LOG PARIZQUIERDO <exp_operacion> PARDERECHO
| MOD PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| PI PARIZQUIERDO PARDERECHO
| POWER PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| RADIANS PARIZQUIERDO <exp_operacion> PARDERECHO
| ROUND PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| SIGN PARIZQUIERDO <exp_operacion> PARDERECHO
| SQRT PARIZQUIERDO <exp_operacion> PARDERECHO
| WIDTH_BUCKET PARIZQUIERDO <exp_operacion> COMA <exp_operacion> COMA <exp_operacion>
COMA <exp_operacion> PARDERECHO
| TRUNC PARIZQUIERDO <exp_operacion> PARDERECHO
| RANDOM PARIZQUIERDO <exp_operacion> PARDERECHO

<funcion_trig> ::= ACOS PARIZQUIERDO <exp_operacion> PARDERECHO
| ACOSD PARIZQUIERDO <exp_operacion> PARDERECHO
| ASIN PARIZQUIERDO <exp_operacion> PARDERECHO
| ASIND PARIZQUIERDO <exp_operacion> PARDERECHO
| ATAN PARIZQUIERDO <exp_operacion> PARDERECHO
| ATAND PARIZQUIERDO <exp_operacion> PARDERECHO
| ATAN2 PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| ATAN2D PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
| COS PARIZQUIERDO <exp_operacion> PARDERECHO
| COSD PARIZQUIERDO <exp_operacion> PARDERECHO
| SIN PARIZQUIERDO <exp_operacion> PARDERECHO
| SIND PARIZQUIERDO <exp_operacion> PARDERECHO
| TAN PARIZQUIERDO <exp_operacion> PARDERECHO
| TAND PARIZQUIERDO <exp_operacion> PARDERECHO
| SINH PARIZQUIERDO <exp_operacion> PARDERECHO
| COSH PARIZQUIERDO <exp_operacion> PARDERECHO
| TANH PARIZQUIERDO <exp_operacion> PARDERECHO
| ASINH PARIZQUIERDO <exp_operacion> PARDERECHO
| ACOSH PARIZQUIERDO <exp_operacion> PARDERECHO
| ATANH PARIZQUIERDO <exp_operacion> PARDERECHO

<funcion_binstr>	::=	LENGTH PARIZQUIERDO <exp_operacion> PARDERECHO
	| SUBSTRING PARIZQUIERDO <exp_operacion> COMA ENTERO COMA ENTERO PARDERECHO
	| TRIM PARIZQUIERDO <exp_operacion> PARDERECHO
	| MD5 PARIZQUIERDO <exp_operacion> PARDERECHO
	| SHA256 PARIZQUIERDO <exp_operacion> PARDERECHO
	| SUBSTR PARIZQUIERDO <exp_operacion> COMA ENTERO COMA ENTERO PARDERECHO
	| GET_BYTE PARIZQUIERDO <exp_operacion> COMA ENTERO PARDERECHO
	| SET_BYTE PARIZQUIERDO <exp_operacion> COMA ENTERO COMA ENTERO PARDERECHO
	| CONVERT PARIZQUIERDO <exp_operacion> AS <tipos> PARDERECHO
	| ENCODE PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO
	| DECODE PARIZQUIERDO <exp_operacion> COMA <exp_operacion> PARDERECHO

<funcion_agregacion>::=	SUM PARIZQUIERDO <exp_operacion PARDERECHO
	| COUNT PARIZQUIERDO <exp_operacion> PARDERECHO
	| COUNT PARIZQUIERDO MULTIPLICACION PARDERECHO
	| AVG PARIZQUIERDO <exp_operacion> PARDERECHO
	| MAX PARIZQUIERDO <exp_operacion> PARDERECHO
	| MIN PARIZQUIERDO <exp_operacion> PARDERECHO

<funcion_exprecion>	::=	GREATEST PARIZQUIERDO <lista_exp> PARDERECHO
	| LEAST PARIZQUIERDO <lista_exp> PARDERECHO

<dec_case>	::=	CASE <lista_when_case> ELSE <exp_operacion> END
	| CASE <lista_when_case> END

<lista_when_case>	::=	<lista_when_case> <clausula_case_when>
	| <clausula_case_when>

<clausula_case_when>::=	WHEN <exp_operacion> THEN <exp_operacion>

<from_query_list>	::=	<from_query_list> COMA <from_query_element>
	| <from_query_element>

<from_query_element>::=	<dec_id_from>
	| <subquery> AS ID
	| <subquery> ID
	| <subquery>

<dec_id_from>	::=	ID AS ID
	| ID ID
	| ID

<lista_condiciones_query>	::=	<lista_condiciones_query> <condicion_query>
	| <condicion_query>

```
<condicion_query>    ::=   WHERE <exp_operacion>
                |   GROUP BY <lista_ids>
                |   HAVING <exp_operacion>
                |   ORDER BY <lista_order_by>
                |   LIMIT <condicion_limit> OFFSET <exp_operacion>
                |   LIMIT <condicion_limit>


<condicion_limit>    ::=   <exp_operacion>
                |    ALL

<lista_ids>          ::=   <lista_ids> COMA <dec_select_columna>
                |   <dec_select_columna>


<lista_order_by>     ::=   <lista_order_by> COMA <elemento_order_by>
                |   <elemento_order_by>

<elemento_order_by>   ::=   <exp_operacion> <asc_desc> NULLS <condicion_null>
                |    <exp_operacion> <asc_desc>

<asc_desc>           ::=   ASC
                |   DESC

<condicion_null>     ::=   FIRST
                |   LAST

<subquery>           ::=   PARIZQUIERDO <select_query> PARDERECHO

<lista_exp>          ::=   <lista_exp> COMA <exp_operacion>
                |   <exp_operacion>

<tipos>           ::=   SMALLINT
                | INTEGER
                | BIGINIT
                | DECIMAL
                | NUMERIC
                | REAL
                | DOUBLE
                | MONEY
                | VARCHAR
                | CHARACTER
                | TEXT
                | TIMESTAMP
                | TIME
                | DATE
                | INTERVAL
                | BOOLEAN
```