Torpedo

Sintaxis consulta SQL

```
SELECT [ALL | DISTINCT] * | [COUNT, MAX, MIN, SUM, AVG](col1), col2, ...
       [AS alias_column]
FROM table1
     [JOIN table2 ON table1.common_field = table2.common_field]
WHERE condition1 [AND | OR condition2]
      [BETWEEN value1 AND value2]
      [IN (value1, value2, ...)]
      [LIKE pattern]
      [= | <> | <= | >= | < | >] value
GROUP BY column1, column2, ...
HAVING aggregate_condition [COUNT, MAX, MIN, SUM, AVG]
ORDER BY column1 [ASC | DESC], column2
UNION | INTERSECT | EXCEPT
[SELECT column1 FROM another_table]
LIMIT row_count;
```

Sintaxis Stored Procedure

```
CREATE OR REPLACE FUNCTION <nombre_función>(<argumentos>)
RETURNS [VOID | TABLE(...) ] AS $$
DECLARE
    <declaración_variables>
BEGIN
    <sentencias_SQL>
END; $$ language plpgsql
```

Sintaxis Trigger

```
CREATE TRIGGER <nombre_trigger>
[BEFORE | AFTER | INSTEAD OF] [ DELETE | UPDATE | INSERT] ON <tabla>
[FOR EACH ROW | FOR EACH STATEMENT]
BEGIN
    UPDATE <tabla_a_actualizar>
    SET <atributo> = <valor>
    WHERE <condición>
END; $$ language plpgsql
```

Sintaxis View

```
CREATE VIEW <nombre_view> AS (
    <sentencias_SQL>
); $$ language plpgsql
```

Sintaxis PL/pgSQL

```
BEGIN
    IF <condición> THEN
        <cuerpo_if>
    ELSE
        <cuerpo_else>
    END IF;
    INSERT INTO <tabla> VALUES (<tupla_valores>);
    FOR <variable> IN <iterable>
    LOOP
        <cuerpo_loop>
    END LOOP;
    RETURN QUERY <consulta_SQL>;
END; $$ language plpgsql
```