TEST

ZHAO WENQI

UCD_16206532BJUT_16372313

1. CATALOGUE

QUERY TEST

1.1 CREATE

1.1.1 CREATE DATABASE

1.1.1.1 CREATE DATABASE testdb;

1.1.1.1.1 CREATE DATABASE testdb;

```
CREATE DATABASE testdb;
------CREATE METHOD --------
CREATE : 53
DATABASE : 359
testdb : 376

----------STRUCTURE----------
CREATE
DATABASE
testdb
```

1.1.2 CREATE TABLE

1.1.2.1 CREATE TABLE table_name (column_name_1 data_type default value column_constraint, column_name_2 data_type default value column_constraint, ..., table_constraint);

//data_type：

TEXT:(**<VARCHAR>**|**<CHAR>**)(**<LBRACKET>**<**NUMBER**><**RBRACKET**>)?|**<BLOB>**

NUMER:|(**<INT>**|**<BIGINT>**)(**<LBRACKET>**<**NUMBER**><**RBRACKET**>)?|**<REAL>**

|(**<FLOAT>**|(**<DOUBLE>**|**<DECIMAL>**)(**<LBRACKET>**<**NUMBER**><**COMMA**><**NUMBER**><**RBRACKET**>)?

DATE:|**<DATE>**|**<TIMESTAMP>**|**<TIME>**|**<YEAT>**

// column_constraint :NOT NULL;PRIMARY KEY ;AUTO_INCREMENT;COMMENT

// table_constraint: PRIMARY KEY('id');PRIMARY KEY (id, name);

1.1.2.1.1 Create table mytable (id char,id1 char(12) primary key, phone int, phone1 int(64));

```
----------STRUCTURE----------
Create
table
mytable
[[id, char], [id1, char, 12, primary key], [phone, int], [phone1, int, 64]]
```

1.1.2.1.2 Create table mytable (id varchar,id1 varchar(12) , phone bigint, phone1 bigint(64) primary key);

```
Create
table
mytable
[[id, varchar], [id1, varchar, 12], [phone, bigint], [phone1, bigint, 64, primary key]]
```

1.1.2.1.3 Create table mytable (id varchar primary key,id1 varchar(12), phone bigint primary key, phone1 bigint(64));

```
----------STRUCTURE----------
Create
table
mytable
[[id, varchar, primary key], [id1, varchar, 12], [phone, bigint, primary key], [phone1, bigint, 64]]
```

1.1.2.1.4 Create table mytable (id float,id1 float(12,4), phone double, phone1 double(12,4), primary key (id));

```
Create
table
mytable
[[id, float], [id1, float, 12, 4], [phone, double], [phone1, double, 12, 4], [primary key, id]]
```

1.1.2.1.5 Create table mytable (id float,id1 float(12,4), phone bigint, phone1 bigint(64), detail_time date, primary key(id,phone));

```
----------STRUCTURE----------
Create
table
mytable
[[id, float], [id1, float, 12, 4], [phone, bigint], [phone1, bigint, 64], [detail_time, date], [primary key, id, phone]]
```

1.1.2.1.6 Create table mytable (id float,id1 float(12,4), phone bigint, phone1 bigint(64), detail_time date, primary key(id,phone, detail_time));

```
Create
table
mytable
[[id, float], [id1, float, 12, 4], [phone, bigint], [phone1, bigint, 64], [detail_time, date], [primary key, id, phone, detail_time]]
```

1.1.2.1.7 Create table mytable (id float,id1 float(12,4), phone bigint, phone1 bigint(64), detail_time date, aday year, primary key(id,phone, detail_time));

```
----------STRUCTURE----------
Create
table
mytable
[[id, float], [id1, float, 12, 4], [phone, bigint], [phone1, bigint, 64], [detail_time, date], [aday, year], [primary key, id, phone, detail_time]]
```

1.1.2.1.8 Create table mytable (id float comment "test",id1 float(12,4), phone bigint, phone1 bigint(64), detail_time date, aday year, primary key(id,phone, detail_time)) ;

```
----------STRUCTURE----------
Create
table
mytable
[[id, float, comment, "test"], [id1, float, 12, 4], [phone, bigint], [phone1, bigint, 64], [detail_time, date], [aday, year], [primary key, id, phone, d
etail_time]]
```

1.2 DROP

1.2.1 DROP DATABASE

1.2.1.1 DROP DATABASE database_name;

1.2.1.1.1 Drop database db1;

```
------DROP METHOD --------
Drop : 76
database : 359
db1 : 376
```

1.2.2 DROP TABLE

1.2.2.1 DROP TABLE tbname;

1.2.2.1.1 Drop table tb1;

```
----------STRUCTURE----------
Drop
table
[tb1]
```

1.2.2.2 DROP TABLE tbname1,tbname2,…;

1.2.2.2.1 Drop table tb1,tb2;

```
----------STRUCTURE----------
Drop
table
[tb1, tb2]
```

1.2.2.2.2 Drop table tb1,tb2,tb3,tb4;

```
----------STRUCTURE----------
Drop
table
[tb1, tb2, tb3, tb4]
```

1.2.3 DROP CCOLUMN

1.2.3.1 ALTER TABLE tbname DROP COLUMN columnName;

1.2.3.1.1 Alter table tb1 drop column c1;

```
----------STRUCTURE----------
Alter
table
tb1
[[drop, column, c1]]
result: test
sql is correct!
```

1.2.3.2 ALTER TABLE tbname DROP COLUMN columnName, DROP COLUMN columnName,…;

1.2.3.2.1 Alter table tb1 drop column c1, drop column c2;

```
----------STRUCTURE----------
Alter
table
tb1
[[drop, column, c1], [drop, column, c2]]
result: test
sql is correct!
```

1.2.3.2.2 Alter table tb1 drop column c1, drop column c2, drop column c3;

```
----------STRUCTURE----------
Alter
table
tb1
[[drop, column, c1], [drop, column, c2], [drop, column, c3]]
result: test
sql is correct!
```

1.3 RENAME

1.3.1 RENAME DATABASE

1.3.1.1 RENAME DATABASE old_name TO new_name;

1.3.1.1.1 Rename database db1 to db2;

```
----------STRUCTURE----------
Rename
database
db1
to
db2
result: test
sql is correct!
```

1.3.2 RENAME TABLE

1.3.2.1 RENAME TABLE tbname TO tbname1;

1.3.2.1.1 Rename table tb1 to tb2;

```
----------STRUCTURE----------
Rename
table
tb1
to
tb2
result: test
sql is correct!
```

1.4 USE

1.4.1 USE

1.4.1.1 USE database_name;

1.4.1.1.1 Use db1;

```
----------STRUCTURE----------
Use
db1
result: test
sql is correct!
```

1.5 SHOW

1.5.1 SHOW DATABASES

1.5.1.1 SHOW DATABASES;

1.5.1.1.1 Show databases;

```
----------STRUCTURE----------
show
databases
result: test
sql is correct!
```

1.5.2 SHOW TABLES

1.5.2.1 SHOW TABLES;

1.5.2.1.1 Show tables;

```
----------STRUCTURE----------
Show
tables
result: test
sql is correct!
```

1.6 ALTER

1.6.1 ALTER TABLE ADD column

1.6.1.1 ALTER TABLE tbname ADD new_column data_type;

1.6.1.1.1 Alter table t1 add c1 int;

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, int]]
result: test
sql is correct!
```

1.6.1.1.2 Alter table t1 add c1 int(64);

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, int, 64]]
result: test
sql is correct!
```

1.6.1.1.3 Alter table t1 add c1 double(12,4);

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4]]
result: test
sql is correct!
```

1.6.1.1.4 Alter table t1 add c1 double(12,4) not null;

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4, not null]]
result: test
sql is correct!
```

1.6.1.1.5 Alter table t1 add c1 double(12,4) primary key;

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4, primary key]]
result: test
sql is correct!
```

1.6.1.1.6 Alter table t1 add c1 double(12,4) primary key comment “test”;

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4, primary key, comment, "test"]]
result: test
sql is correct!
```

1.6.1.2 ALTER TABLE tbname ADD new_column data_type [AFTER existing_column]

1.6.1.2.1 Alter table t1 add c1 double(12,4) primary key comment "test" after c2;

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4, primary key, comment, "test", after, c2]]
result: test
sql is correct!

------------------------------------------------------------
```

1.6.1.3 ALTER TABLE tbname ADD…, ADD …;ADD …;…..;

1.6.1.3.1 Alter table t1 add c1 double(12,4) primary key comment "test" after c2, add c2 int(64);

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4, primary key, comment, "test", after, c2], [add, c2, int, 64]]
result: test
sql is correct!
```

1.6.1.3.2 Alter table t1 add c1 double(12,4) primary key comment "test" after c2, add c2 float, add c3 char;

```
----------STRUCTURE----------
Alter
table
t1
[[add, c1, double, 12, 4, primary key, comment, "test", after, c2], [add, c2, float], [add, c3, char]]
result: test
sql is correct!
```

1.6.2 ALTER TABLE MODIFY

1.6.2.1 ALTER TABLE tbname MODIFY column_definition;

1.6.2.1.1 ALTER TABLE tb1 MODIFY fee double (10,2) NOT NULL;

```
----------STRUCTURE----------
ALTER
TABLE
tb1
MODIFY
[[fee, double, 10, 2, NOT NULL]]
result: test
sql is correct!
```

1.6.3 ALTER TABLE DROP

1.6.3.1 ALTER TABLE tbname DROP COLUMN column_name;

1.6.3.1.1 ALTER TABLE tbname DROP COLUMN fee;

```
----------STRUCTURE----------
ALTER
TABLE
tbname
[[DROP, COLUMN, fee]]
result: test
sql is correct!
```

1.6.3.2 ALTER TABLE tbname DROP COLUMN column_name, DROP COLUMN column_name.

1.6.3.2.1 ALTER TABLE tbname DROP COLUMN fee, DROP COLUMN fee2;

```
----------STRUCTURE----------
ALTER
TABLE
tbname
[[DROP, COLUMN, fee], [DROP, COLUMN, fee2]]
result: test
sql is correct!
```

1.7 TRUNCATE TABLE

1.7.1 TRUNCATE tbname;

1.7.1.1.1 TRUNCATE tb1;

```
----------STRUCTURE----------
TRUNCATE
tb1
result: test
sql is correct!
```

1.8 DELETE

1.8.1 DELETE ONE LINE

1.8.1.1.1 DELETE FROM departments WHERE department_id = 16;

```
----------STRUCTURE----------
DELETE
FROM
departments
WHERE
[[department_id, =, 16]]
result: test
sql is correct!
```

1.8.1.1.2 DELETE FROM departments WHERE department_name = 'a';

```
----------STRUCTURE----------
DELETE
FROM
departments
WHERE
[[department_name, =, "a"]]
result: test
sql is correct!
```

1.8.2 DELETE MULTIPLE LINES

1.8.2.1.1 DELETE FROM departments WHERE employee_id IN (100, 101, 102);

```
----------STRUCTURE----------
DELETE
FROM
departments
WHERE
[[employee_id, IN, [100, 101, 102]]]
result: test
sql is correct!
```

1.8.2.1.2 DELETE FROM departments WHERE employee_id BETWEEN 100 AND 102;

```
----------STRUCTURE----------
DELETE
FROM
departments
WHERE
[[employee_id, BETWEEN, 100, AND, 200]]
result: test
sql is correct!
```

1.8.2.1.3 DELETE FROM departments WHERE employee_id < 7 OR c1 = "c";

```
----------STRUCTURE----------
DELETE
FROM
departments
WHERE
[[employee_id, <, 7], OR, [c1, =, "c"]]
result: test
sql is correct!
```

1.9 INSERT

1.9.1 INSERT ONE LINE

1.9.1.1 INSERT INTO table1 (column1, coulumn2,…) VALUES (value1, value2 , …);//value = number or text;

1.9.1.1.1 Insert into t1(c1) values (3);

```
----------STRUCTURE----------
Insert
into
t1
[c1]
values
[[3]]
result: test
sql is correct!
```

1.9.1.1.2 Insert into t1(c1,c2,c5) values (3,2,4);

```
----------STRUCTURE----------
Insert
into
t1
[c1, c2, c5]
values
[[3], [2], [4]]
result: test
sql is correct!
```

1.9.1.2 INSERT INTO table1 VALUES (value1, value2,…)

1.9.1.2.1 Insert into t1 values (3,2,4);

```
----------STRUCTURE----------
Insert
into
t1
[]
values
[[3], [2], [4]]
result: test
sql is correct!
```

1.9.2 INSERT MULTIPLE LINES

1.9.2.1 INSERT INTO table1 VALUES (value1, value2,…), (value1, value2,…),…;

1.9.2.1.1 Insert into t1 values (3,2,4),(324,324,324);

```
----------STRUCTURE----------
Insert
into
t1
[]
values
[[3], [2], [4]]
[[324], [324], [324]]
result: test
sql is correct!
```

1.9.2.1.2 Insert into t1 values (3,2,"b"),(324,324,"a");

```
----------STRUCTURE----------
Insert
into
t1
[]
values
[[3], [2], ["b"]]
[[324], [324], ["a"]]
result: test
sql is correct!
```

1.9.2.2 INSERT INTO table1 (name1, name2) VALUES (value1, value2), (value1, value2),…;

1.9.2.2.1 Insert into t1(c1,c2,c5) values (3,2,4),(23,324,324);

```
----------STRUCTURE----------
Insert
into
t1
[c1, c2, c5]
values
[[3], [2], [4]]
[[23], [324], [324]]
result: test
sql is correct!
```

1.9.3 INSERT FROM OTHER TABLE

1.9.3.1 INSERT INTO table1 (column1, column2) SELECT column1, column2 FROM table2 where condition1;

1.9.3.1.1 Insert into t1(c1,c2,c5) select c1 from t2 where c3 = 2;

```
----------STRUCTURE----------
Insert
into
t1
[c1, c2, c5]
[select, [[c1]], from, [[t2]], where, [c3, =, 2]]
result: test
sql is correct!
```

1.9.3.1.2 Insert into t1(c1,c2,c5) select c1,c2 from t2 where c3 = 2;

```
----------STRUCTURE----------
Insert
into
t1
[c1, c2, c5]
[select, [[c1], [c2]], from, [[t2]], where, [c3, =, 2]]
result: test
sql is correct!
```

1.9.3.1.3 Insert into t1(c1,c2,c5) select * from t2 where c3 = 2;

```
----------STRUCTURE----------
Insert
into
t1
[c1, c2, c5]
[select, *, from, [[t2]], where, [c3, =, 2]]
result: test
sql is correct!
```

*1.9.3.2 INSERT INTO table1 SELECT * FROM departments;*

1.9.3.2.1 Insert into t1 select * from t2 where c3 = 2;

```
----------STRUCTURE----------
Insert
into
t1
[]
[select, *, from, [[t2]], where, [c3, =, 2]]
result: test
sql is correct!
```

1.10 UPDATE

1.10.1 UPDATE table SET

1.10.1.1 UPDATE table SET column1 = value1, column2 = value2 WHERE condition;

1.10.1.1.1 UPDATE T1 SET column1 = value1, column2 = value2 WHERE column3 is not null;

```
----------STRUCTURE----------
UPDATE
T1
SET
[[column1, =, value1], [column2, =, value2]]
WHERE
[[column3, is, not null]]
result: test
sql is correct!
```

1.10.1.1.2 UPDATE T1 SET column1 = value1, column2 = value2 WHERE column3 like "%3";

```
----------STRUCTURE----------
UPDATE
T1
SET
[[column1, =, value1], [column2, =, value2]]
WHERE
[[column3, like, "%3"]]
result: test
sql is correct!
```

1.10.1.1.3 UPDATE T1 SET column1 = value1, column2 = value2 WHERE column3 in (2,6);

```
----------STRUCTURE----------
UPDATE
T1
SET
[[column1, =, value1], [column2, =, value2]]
WHERE
[[column3, in, [2, 6]]]
result: test
sql is correct!
```

1.10.2 UPDATE table with JOIN

1.10.2.1 UPDATE table INNER JOIN (LEFT /RIGHT/FULL./CROSS JOIN)table1 ON table.column1 = table2.column1 SET table.column2 = table1.coulmn2,… (WHERE);

1.10.2.1.1 UPDATE T1 INNER JOIN table1 ON table.column1 = table2.column1 SET table.column2 = table1.coulmn2;

```
----------STRUCTURE----------
UPDATE
T1
INNER
JOIN
table1
ON
[table.column1, =, table2.column1]
SET
[table.column2, =, table1.coulmn2]
result: test
sql is correct!
```

1.10.2.1.2 UPDATE T1 LEFT JOIN table1 ON table.column1 = table2.column1 SET table.column2 = table1.coulmn2 WHERE c3 = 4;

```
----------STRUCTURE----------
UPDATE
T1
LEFT
JOIN
table1
ON
[table.column1, =, table2.column1]
SET
[table.column2, =, table1.coulmn2]
WHERE
[[c3, =, 4]]
result: test
sql is correct!
```

1.10.2.1.3 UPDATE T1 RIGHT JOIN table1 ON table.column1 = table2.column1 SET table.column2 = table1.coulmn2 WHERE c3 = 4;

```
----------STRUCTURE----------
UPDATE
T1
RIGHT
JOIN
table1
ON
[table.column1, =, table2.column1]
SET
[table.column2, =, table1.coulmn2]
WHERE
[[c3, =, 4]]
result: test
sql is correct!
```

1.10.2.1.4 UPDATE T1 FULL JOIN table1 ON table.column1 = table2.column1 SET table.column2 = table1.coulmn2 WHERE c3 = 4;

```
----------STRUCTURE----------
UPDATE
T1
FULL
JOIN
table1
ON
[table.column1, =, table2.column1]
SET
[table.column2, =, table1.coulmn2]
WHERE
[[c3, =, 4]]
result: test
sql is correct!
```

1.10.2.1.5 UPDATE T1 CROSS JOIN table1 ON table.column1 = table2.column1 SET table.column2 = table1.coulmn2 WHERE c3 = 4;

```
----------STRUCTURE----------
UPDATE
T1
CROSS
JOIN
table1
ON
[table.column1, =, table2.column1]
SET
[table.column2, =, table1.coulmn2]
WHERE
[[c3, =, 4]]
result: test
sql is correct!
```

1.11 SELECT

1.11.1 SELECT SINGLE COLUMN

1.11.1.1.1 SELECT column1 FROM tbname;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.2 SELECT MULTIPAL COLUMN

1.11.2.1.1 SELECT column1,column2,column3 FROM tbname;

```
----------STRUCTURE----------
SELECT
[[column1], [column2], [column3]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.3 SELECT ALL COLUMN

1.11.3.1.1 SELECT * FROM tbname;

```
----------STRUCTURE----------
SELECT
*
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.3.1.2 SELECT ALL FROM tbname;

```
----------STRUCTURE----------
SELECT
ALL
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.4 SELECT WITH AS AND FROM

1.11.4.1.1 SELECT column1,column2,column3 AS a FROM tbname;

```
----------STRUCTURE----------
SELECT
[[column1], [column2], [column3, AS, a]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.4.1.2 SELECT column1,column2,column3 FROM tbname;

```
----------STRUCTURE----------
SELECT
[[column1], [column2], [column3]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.4.1.3 SELECT column1 FROM tbname AS cs;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname, AS, cs]]
result: test
sql is correct!
```

1.11.4.1.4 SELECT column1 FROM tbname cs;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname, cs]]
result: test
sql is correct!
```

1.11.5 SELECT FROM MULTIPLE TABLES

1.11.5.1.1 SELECT column1 FROM a,b;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[a], [b]]
result: test
sql is correct!
```

1.11.6 SELECT WITH FUNTION

1.11.6.1.1 SELECT column1， FLOOR (DATEDIFF(NOW(), xxx_date)) FROM tbname;

```
----------STRUCTURE----------
SELECT
[[column1], [NOW, xxx_date]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.6.1.2 SELECT column1， FLOOR (DATEDIFF(NOW(), xxx_date)) AS a FROM tbname;

```
----------STRUCTURE----------
SELECT
[[column1], [NOW, xxx_date, AS, a]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.7 SELECT WITH ORDER BY

1.11.7.1.1 SELECT column1， column2 FROM tbname order by column1;

```
----------STRUCTURE----------
SELECT
[[column1], [column2]]
FROM
[[tbname]]
order by
[[column1]]
result: test
sql is correct!
```

1.11.7.1.2 SELECT column1， column2 FROM tbname order by column1 ASC, column2 DESC;

```
----------STRUCTURE----------
SELECT
[[column1], [column2]]
FROM
[[tbname]]
order by
[[column1, ASC], [column2, DESC]]
result: test
sql is correct!
```

1.11.7.1.3 SELECT DISTINCT column1,column2 FROM tbname;

```
----------STRUCTURE----------
SELECT
[[DISTINCT, column1], [column2]]
FROM
[[tbname]]
result: test
sql is correct!
```

1.11.8 SELECT WITH LIMIT

1.11.8.1.1 SELECT column1, column2 FROM tbname ORDER BY xx LIMIT 5 OFFSET3;

```
----------STRUCTURE----------
SELECT
[[column1], [column2]]
FROM
[[tbname]]
ORDER BY
[[xx]]
LIMIT
5
OFFSET
3
result: test
sql is correct!
```

1.11.9 SELECT WITH FETCH

1.11.9.1.1 SELECT column1, column2 FROM tbname ORDER BY xx FETCH NEXT 3 ROWS ONLY;

```
----------STRUCTURE----------
SELECT
[[column1], [column2]]
FROM
[[tbname]]
ORDER BY
[[xx]]
FETCH
NEXT
3
ROWS
ONLY
result: test
sql is correct!
```

1.11.9.1.2 SELECT column1, column2 FROM tbname ORDER BY xx ASC LIMIT 4 OFFSET 2 ROWS FETCH NEXT 3 ROWS ONLY ;

```
----------STRUCTURE----------
SELECT
[[column1], [column2]]
FROM
[[tbname]]
ORDER BY
[[xx, ASC]]
LIMIT
4
OFFSET
2
ROWS
FETCH
NEXT
3
ROWS
ONLY
result: test
sql is correct!
```

1.11.9.1.3 SELECT column1, column2 FROM tbname ORDER BY xx DESC, XX2 ASC LIMIT 7 OFFSET 2ROWS FETCH NEXT 3 ROWS;

```
----------STRUCTURE----------
SELECT
[[column1], [column2]]
FROM
[[tbname]]
ORDER BY
[[xx, DESC], [XX2, ASC]]
LIMIT
7
OFFSET
2
ROWS
FETCH
NEXT
3
ROWS
ONLY
result: test
sql is correct!
```

1.11.9.1.4 SELECT AVG(column1), COUNT(column2) FROM tbname ORDER BY xx DESC, XX2 ASC LIMIT 7 OFFSET 2ROWS FETCH NEXT 3 ROWS ONLY;

```
----------STRUCTURE----------
SELECT
[[AVG, column1], [COUNT, column2]]
FROM
[[tbname]]
ORDER BY
[[xx, DESC], [XX2, ASC]]
LIMIT
7
OFFSET
2
ROWS
FETCH
NEXT
3
ROWS
ONLY
result: test
sql is correct!
```

1.11.9.1.5 SELECT MAX(column1), MIN(column2) FROM tbname ORDER BY xx DESC, XX2 ASC LIMIT 7 OFFSET 2ROWS FETCH NEXT 3 ROWS ONLY;

```
----------STRUCTURE----------
SELECT
[[MAX, column1], [MIN, column2]]
FROM
[[tbname]]
ORDER BY
[[xx, DESC], [XX2, ASC]]
LIMIT
7
OFFSET
2
ROWS
FETCH
NEXT
3
ROWS
ONLY
result: test
sql is correct!
```

1.11.9.1.6 SELECT SUM(column1), column2 FROM tbname WHERE ASDF = 3 AND ADSF >5 AND AFAFAFDA IN (234,234,234) ORDER BY xx DESC, XX2 ASC;

```
----------STRUCTURE----------
SELECT
[[SUM, column1], [column2]]
FROM
[[tbname]]
WHERE
[[ASDF, =, 3], AND, [ADSF, >, 5], AND, [AFAFAFDA, IN, [234, 234, 234]]]
ORDER BY
[[xx, DESC], [XX2, ASC]]
result: test
sql is correct!

------------------------------------------------------------
```

1.11.10 SELECT WITH WHERE

1.11.10.1.1 SELECT column1 FROM tbname WHERE a>b;(<,>,=,<=,>=,!=);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, >, b]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, <, b]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, <=, b]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, >=, b]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, =, b]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, !=, b]]
result: test
sql is correct!
```

1.11.10.1.2SELECT column1 FROM tbname WHERE c1>"b" AND c2>"c";

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[c1, >, "b"], AND, [c2, >, "c"]]
result: test
sql is correct!
```

1.11.10.1.3SELECT column1 FROM tbname WHERE c1>"b" or c2>"c";

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[c1, >, "b"], OR, [c2, >, "c"]]
result: test
sql is correct!
```

1.11.10.1.4SELECT column1 FROM tbname WHERE YEAR(c1)>3 AND c2>4;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[YEAR, c1, >, 3], AND, [c2, >, 4]]
result: test
sql is correct!
```

1.11.10.1.5 SELECT column1 FROM tbname WHERE YEAR(c1)>3 OR NOT c2>4;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[YEAR, c1, >, 3], OR, [c2, >, 4]]
result: test
sql is correct!
```

1.11.10.1.6 SELECT column1 FROM tbname WHERE a1 BETWEEN 231 AND 324;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, BETWEEN, 231, AND, 324]]
result: test
sql is correct!
```

1.11.10.1.7 SELECT column1 FROM tbname WHERE a1 NOT BETWEEN 231 AND 324;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, BETWEEN, 231, AND, 324]]
result: test
sql is correct!
```

1.11.10.1.8SELECT column1 FROM tbname WHERE a1 IN(213,123,123);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, IN, [213, 123, 123]]]
result: test
sql is correct!
```

1.11.10.1.9SELECT column1 FROM tbname WHERE a1 NOT IN(213,123,123);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, not, IN, [213, 123, 123]]]
result: test
sql is correct!
```

1.11.10.1.10 SELECT column1 FROM tbname WHERE a1 LIKE 'ab'

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, LIKE, "ab"]]
result: test
sql is correct!
```

1.11.10.1.11 SELECT column1 FROM tbname WHERE a1 LIKE '%na'('adf%','%da%')/'a__ 'a_'

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, LIKE, "%na"]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, LIKE, "a%"]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, LIKE, "__a"]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, LIKE, "__a__"]]
result: test
sql is correct!
```

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, LIKE, "a__"]]
result: test
sql is correct!
```

1.11.10.1.12 SELECT column1 FROM tbname WHERE a1 NOT LIKE '%na'

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, LIKE, "%a"]]
result: test
sql is correct!
```

1.11.10.1.13 SELECT column1 FROM tbname WHERE a1 LIKE "%a_";

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, LIKE, "%a_"]]
result: test
sql is correct!
```

1.11.10.1.14 SELECT column1 FROM tbname WHERE a1 LIKE "%a___";

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, NOT, LIKE, "%a___"]]
result: test
sql is correct!
```

1.11.10.1.15 SELECT column1 FROM tbname WHERE a1 IS NOT NULL;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, IS, NOT NULL]]
result: test
sql is correct!
```

1.11.10.1.16 SELECT column1 FROM tbname WHERE a1 IS NULL;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a1, IS, NULL]]
result: test
sql is correct!

----------------------------------------------------------
```

1.11.10.1.17 SELECT column1 FROM tbname WHERE a>b ORDER BY c1;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, >, b]]
ORDER BY
[[c1]]
result: test
sql is correct!
```

1.11.10.1.18 SELECT column1 FROM tbname WHERE a>b ORDER BY c1,d1;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, >, b]]
ORDER BY
[[c1], [d1]]
result: test
sql is correct!
```

1.11.11 SELECT WITH SUBQUERY

1.11.11.1.1 SELECT column1 FROM tbname WHERE a IN (SELECT * From tb1);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, IN, [[SELECT, *, From, [[tb1]]]]]]
result: test
sql is correct!
```

1.11.11.1.2 SELECT column1 FROM tbname WHERE a IN (SELECT c1 From tb1 where c1=3);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, IN, [[SELECT, [[c1]], From, [[tb1]], where, [c1, =, 3]]]]]
result: test
sql is correct!
```

1.11.11.1.3 SELECT column1 FROM tbname WHERE a IN (SELECT c1,c2 From tb1 where c1=3 and c2>100);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, IN, [[SELECT, [[c1], [c2]], From, [[tb1]], where, [[c1, =, 3], and, [c2, >, 100]]]]]]
result: test
sql is correct!
```

1.11.11.1.4SELECT column1 FROM tbname WHERE a IN (SELECT c1,c2 From tb1 where c1=3 and c2 in (2,3));

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[c2, in, [2, 3]], [a, IN, [[SELECT, [[c1], [c2]], From, [[tb1]], where, [[c1, =, 3], and]]]]]
result: test
sql is correct!
```

1.11.11.1.5SELECT column1 FROM tbname WHERE a IN (SELECT all From tb1 where c1 like “ab”);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, IN, [[SELECT, all, From, [[tb1]], where, [c1, like, "ab"]]]]]
result: test
sql is correct!
```

1.11.11.1.6SELECT column1 FROM tbname WHERE a IN (SELECT all From tb1 where c1 like “ab” order by c4);

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
WHERE
[[a, IN, [[SELECT, all, From, [[tb1]], where, [c1, like, "ab"], order by, [[c4]]]]]]
result: test
sql is correct!

------------------------------------------------------------
```

1.11.12 SELECT WITH JOIN

1.11.12.1.1 SELECT column1 FROM tbname INNER JOIN cvad ON adf.adf = df.adf;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
INNER
JOIN
cvad
ON
[adf.adf, =, df.adf]
result: test
sql is correct!
```

1.11.12.1.2 SELECT column1 FROM tbname INNER JOIN cvad ON adf.adf = df.adf order by aa;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
INNER
JOIN
cvad
ON
[adf.adf, =, df.adf]
order by
[[aa]]
result: test
sql is correct!
```

1.11.12.1.3SELECT column1 FROM tbname INNER JOIN cvad ON adf.adf = df.adf group by caf;

```
----------STRUCTURE----------
SELECT
[[column1]]
FROM
[[tbname]]
INNER
JOIN
cvad
ON
[adf.adf, =, df.adf]
group by
[[caf]]
result: test
sql is correct!

-----------------------------------------------------------
```

1.11.12.1.4SELECT a1.n FROM a1 INNER JOIN b1 ON b1.n = a1.n INNER JOIN c ON c1.n = a1.n;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
INNER
JOIN
b1
ON
[b1.n, =, a1.n]
INNER
JOIN
c1
ON
[c1.n, =, a1.n]
result: test
sql is correct!
```

1.11.12.1.5SELECT a1.n FROM a1 INNER JOIN b1 ON b1.n = a1.n INNER JOIN c ON c1.n = a1.n order by c2;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
INNER
JOIN
b1
ON
[b1.n, =, a1.n]
INNER
JOIN
c1
ON
[c1.n, =, a1.n]
order by
[[c2]]
result: test
sql is correct!
```

1.11.12.1.6SELECT a1.n FROM a1 INNER JOIN b1 ON b1.n = a1.n INNER JOIN c1 ON c1.n = a1.n where c1=3;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
INNER
JOIN
b1
ON
[b1.n, =, a1.n]
INNER
JOIN
c1
ON
[c1.n, =, a1.n]
where
[[c1, =, 3]]
result: test
sql is correct!

----------------------------------------------------------
```

1.11.12.1.7SELECT a1.n FROM a1 LEFT JOIN b1 ON b1.n = a1.n;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
LEFT
JOIN
b1
ON
[b1.n, =, a1.n]
result: test
sql is correct!
```

1.11.12.1.8 SELECT a1.n FROM a1 LEFT JOIN b1 ON b1.n = a1.n LEFT JOIN c1 ON c1.n = a1.n ;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
LEFT
JOIN
b1
ON
[b1.n, =, a1.n]
LEFT
JOIN
c1
ON
[c1.n, =, a1.n]
result: test
sql is correct!
```

1.11.12.1.9 SELECT a1.n FROM a1 FULL JOIN b1 ON b1.n = a1.n;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
FULL
JOIN
b1
ON
[b1.n, =, a1.n]
result: test
sql is correct!
```

1.11.12.1.10 SELECT a1.n FROM a1 CROSS JOIN b1 ON b1.n = a1.n;

```
----------STRUCTURE----------
SELECT
[[a1.n]]
FROM
[[a1]]
CROSS
JOIN
b1
ON
[b1.n, =, a1.n]
result: test
sql is correct!
```