Er-model功能测试

导入

支持导入

- PostgrsSQL
- MySQL
- openGauss

导入后自动识别为DBML出现在设计er图页面，右侧编辑区

示例：

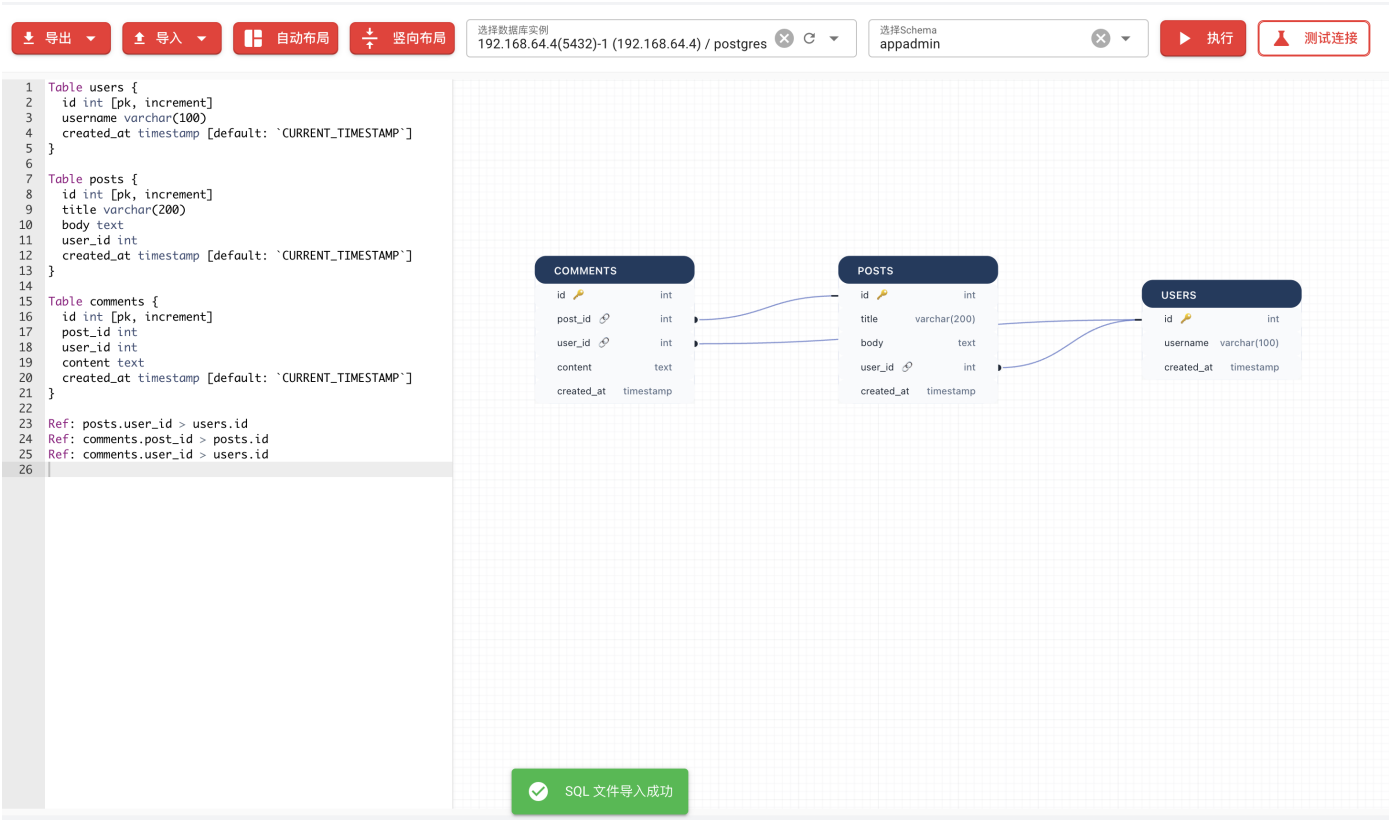
导入的MySQL语句

MySQL_demonstrate.sql

MySQL_demonstrate.sql

```sql
CREATE TABLE users (
    id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(100) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE posts (
    id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(200) NOT NULL,
    body TEXT,
    user_id INT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT fk_posts_user FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE comments (
    id INT PRIMARY KEY AUTO_INCREMENT,
    post_id INT,
    user_id INT,
    content TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT fk_comments_post FOREIGN KEY (post_id) REFERENCES posts(id),
    CONSTRAINT fk_comments_user FOREIGN KEY (user_id) REFERENCES users(id)
);
```

解析的DBML语句：



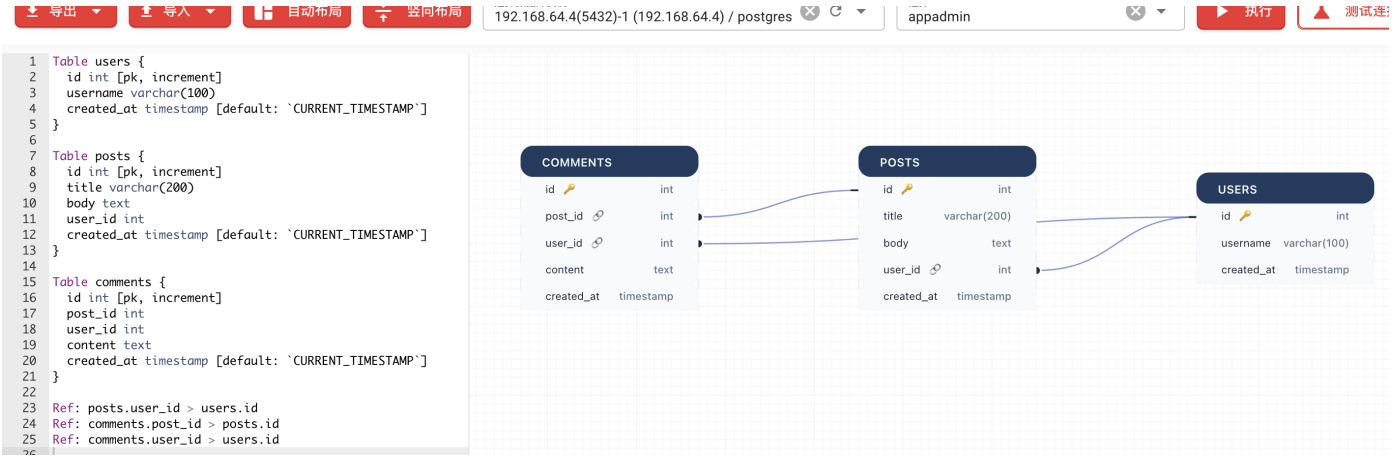
导出

根据DBML代码和由此生成的ER图，支持导出

- 导出 SQL(openGauss)
- 导出 SQL(PostgreSQL)
- 导出 SQL(MySQL)
- 导出 PNG 图片
- 导出 PDF 文档
- 导出 PlantUML

示例：

原始DBML代码和ER图：



导出的openGaussSQL：

er-export-opengauss (16).sql — UNREGISTERED

er-export-opengauss (16).sql

```sql
CREATE TABLE users (
  id SERIAL PRIMARY KEY,
  username VARCHAR(100),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE posts (
  id SERIAL PRIMARY KEY,
  title VARCHAR(200),
  body TEXT,
  user_id INTEGER,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  CONSTRAINT fk_posts_users FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE comments (
  id SERIAL PRIMARY KEY,
  post_id INTEGER,
  user_id INTEGER,
  content TEXT,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  CONSTRAINT fk_comments_posts FOREIGN KEY (post_id) REFERENCES posts(id),
  CONSTRAINT fk_comments_users FOREIGN KEY (user_id) REFERENCES users(id)
);

```

Line 25, Column 1 Spaces: 2 SQL

导出的png图片

导出的plantUML代码：

192.168.64.4(5432)-1 (192.168.64.4) _ postgres-plantuml.puml UNREGISTERED

192.168.64.4(5432)-1 (192.168.64.4) _ postgres-plantuml.puml

```
@startuml
left to right direction
hide circle
skinparam linetype ortho

entity "users" as T1 {
  *id : INT <<PK>>
   username : VARCHAR(100)
   created_at : TIMESTAMP = CURRENT_TIMESTAMP
}

entity "posts" as T2 {
  *id : INT <<PK>>
   title : VARCHAR(200)
   body : TEXT
   user_id : INT <<FK>>
   created_at : TIMESTAMP = CURRENT_TIMESTAMP
}

entity "comments" as T3 {
  *id : INT <<PK>>
   post_id : INT <<FK>>
   user_id : INT <<FK>>
   content : TEXT
   created_at : TIMESTAMP = CURRENT_TIMESTAMP
}

T1 ||--o{ T2
T2 ||--o{ T3
T1 ||--o{ T3
@enduml
```

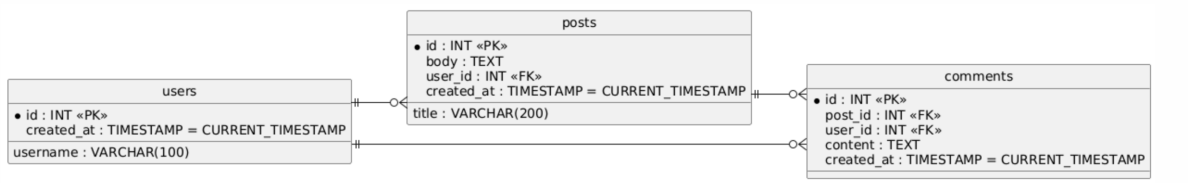
Line 24, Column 18 Spaces: 2 Plain Text

plantUML代码解析后生成图片：

```
@startuml
left to right direction
hide circle
skinparam linetype ortho

entity "users" as T1 {
  *id : INT <<PK>>
   username : VARCHAR(100)
   created_at : TIMESTAMP = CURRENT_TIMESTAMP
}

entity "posts" as T2 {
  *id : INT <<PK>>
   title : VARCHAR(200)
   body : TEXT
   user_id : INT <<FK>>
   created_at : TIMESTAMP = CURRENT_TIMESTAMP
}

entity "comments" as T3 {
  *id : INT <<PK>>
   post_id : INT <<FK>>
   user_id : INT <<FK>>
   content : TEXT
   created_at : TIMESTAMP = CURRENT_TIMESTAMP
}
```

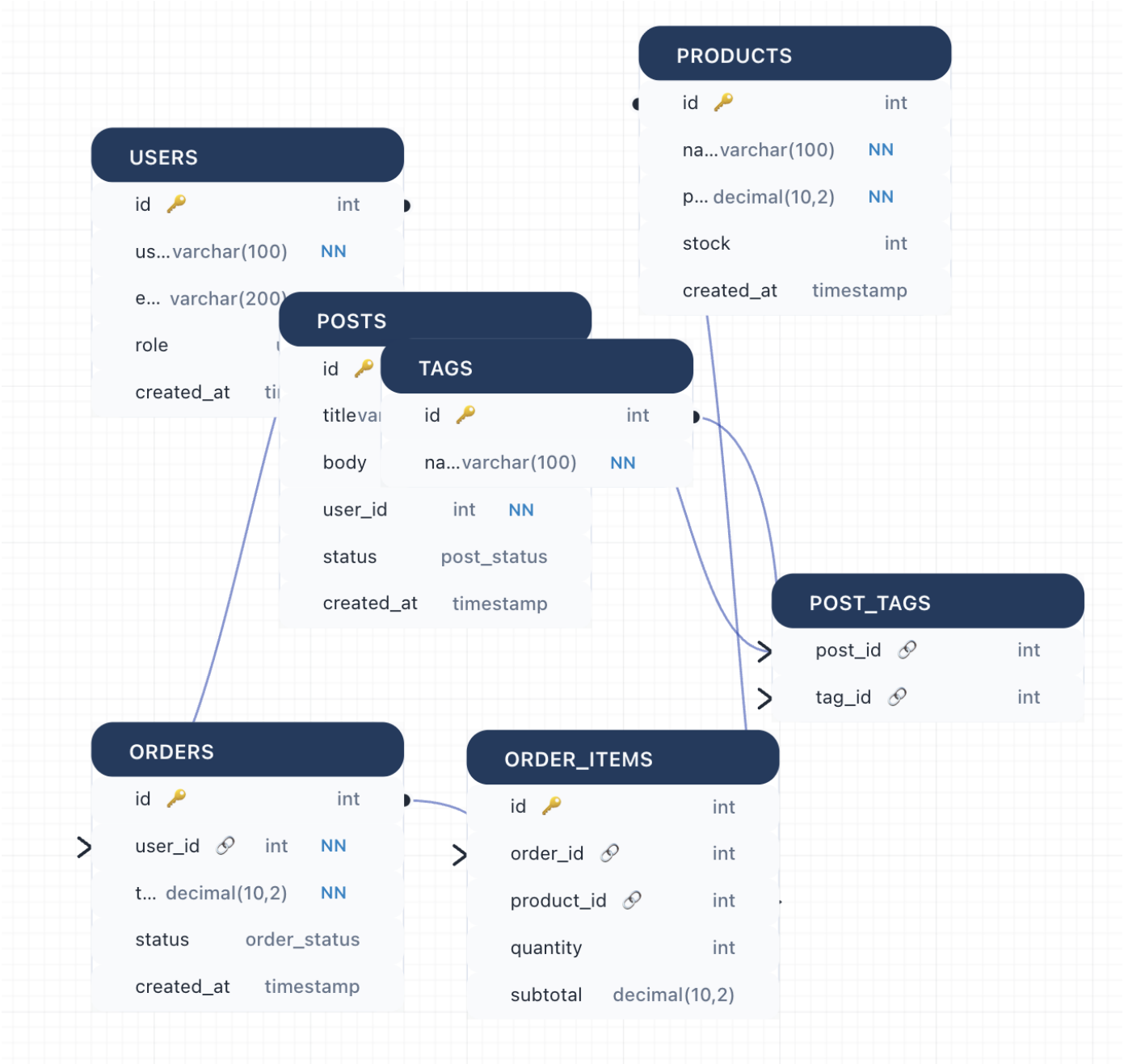
//www.plantuml.com/plantuml/png/dOzFIyCm68Vl-HGlEwamMDkRpB4oAWxP63MAjnARLnlCdvAyFPJDxsuBtUp17lIOFFdnF5a74fuQevd6Tm9ouDL7HI2LnvAKiwnI4g5KljJ8mgUojV32W5OMgQqHdAVACOQM5BKmQmBwC0CHW4TmP00tI

Submit Discover the future PlantUML Web Editor! PNG SVG ASCII Art

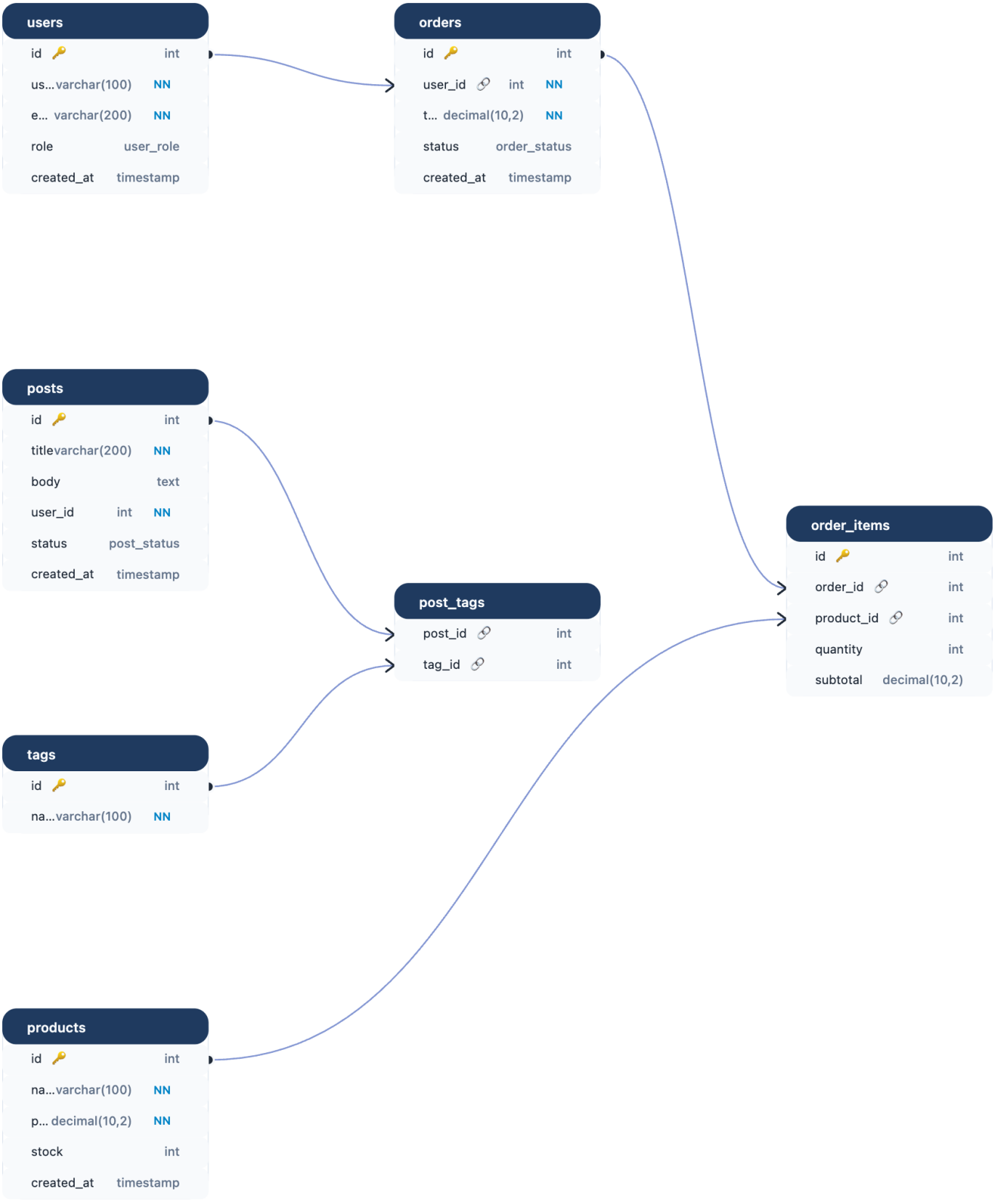


自动布局

初始杂乱无章的er图：

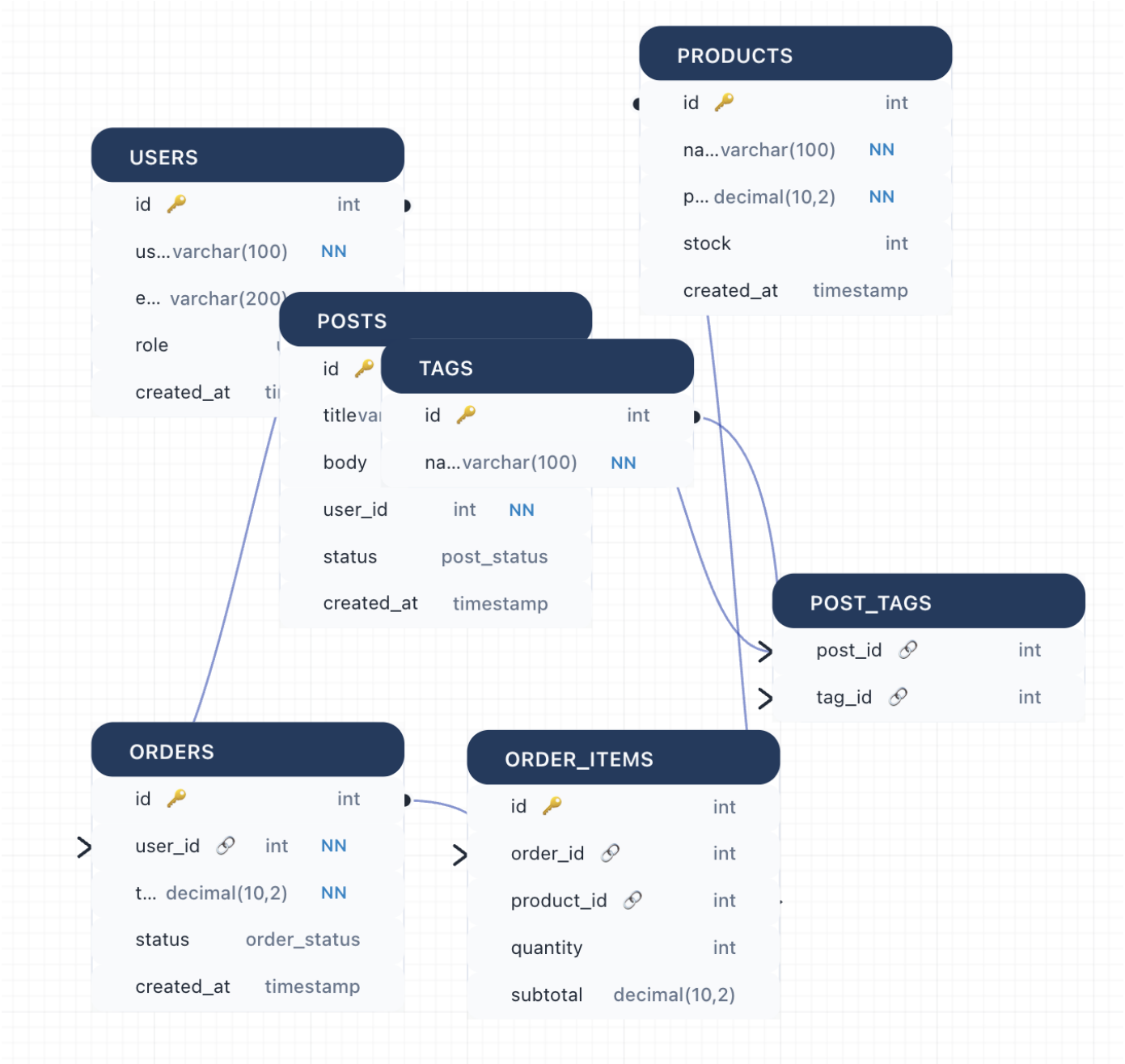


自动布局后：

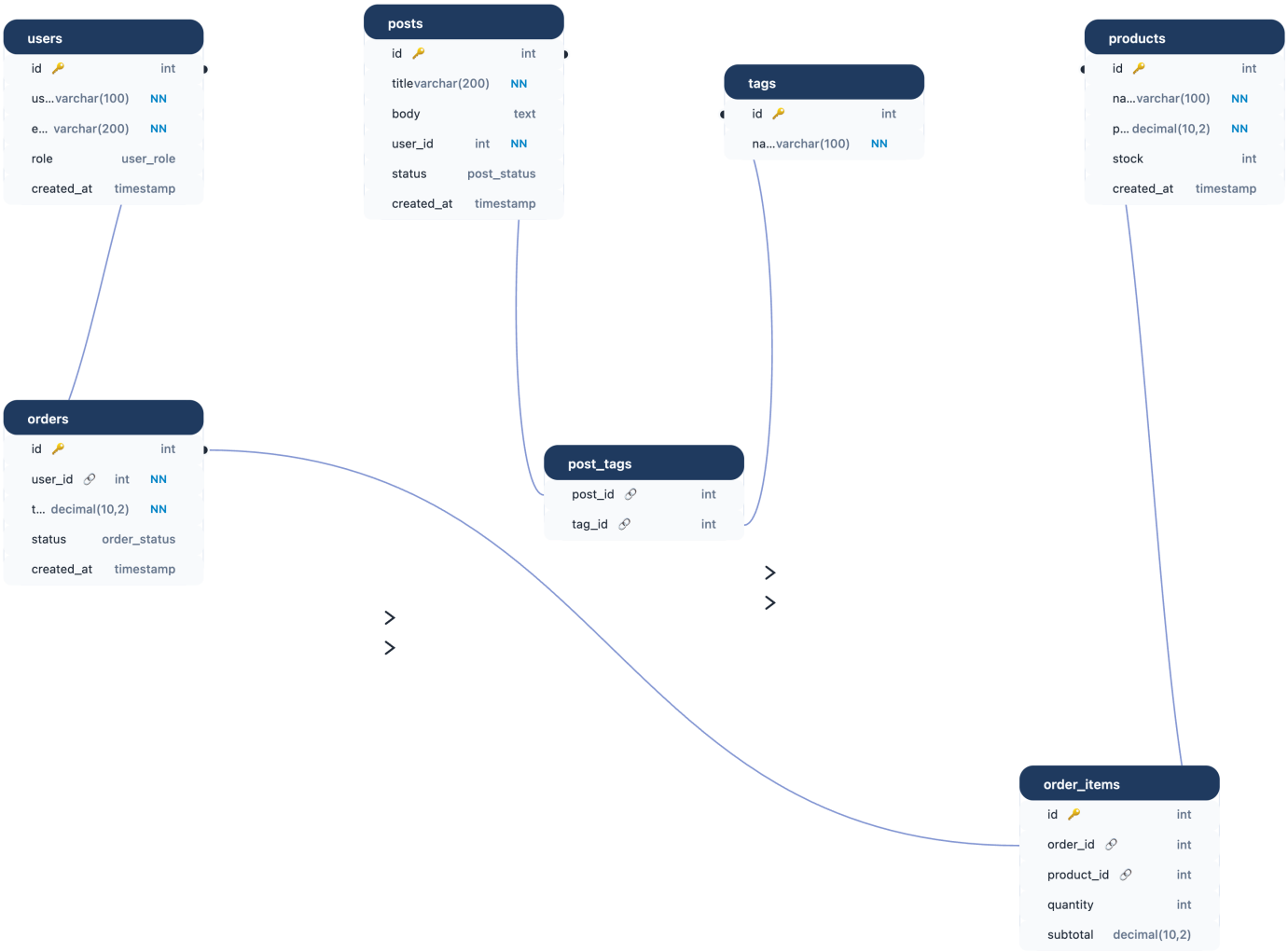


竖向布局

初始杂乱无章的er图：

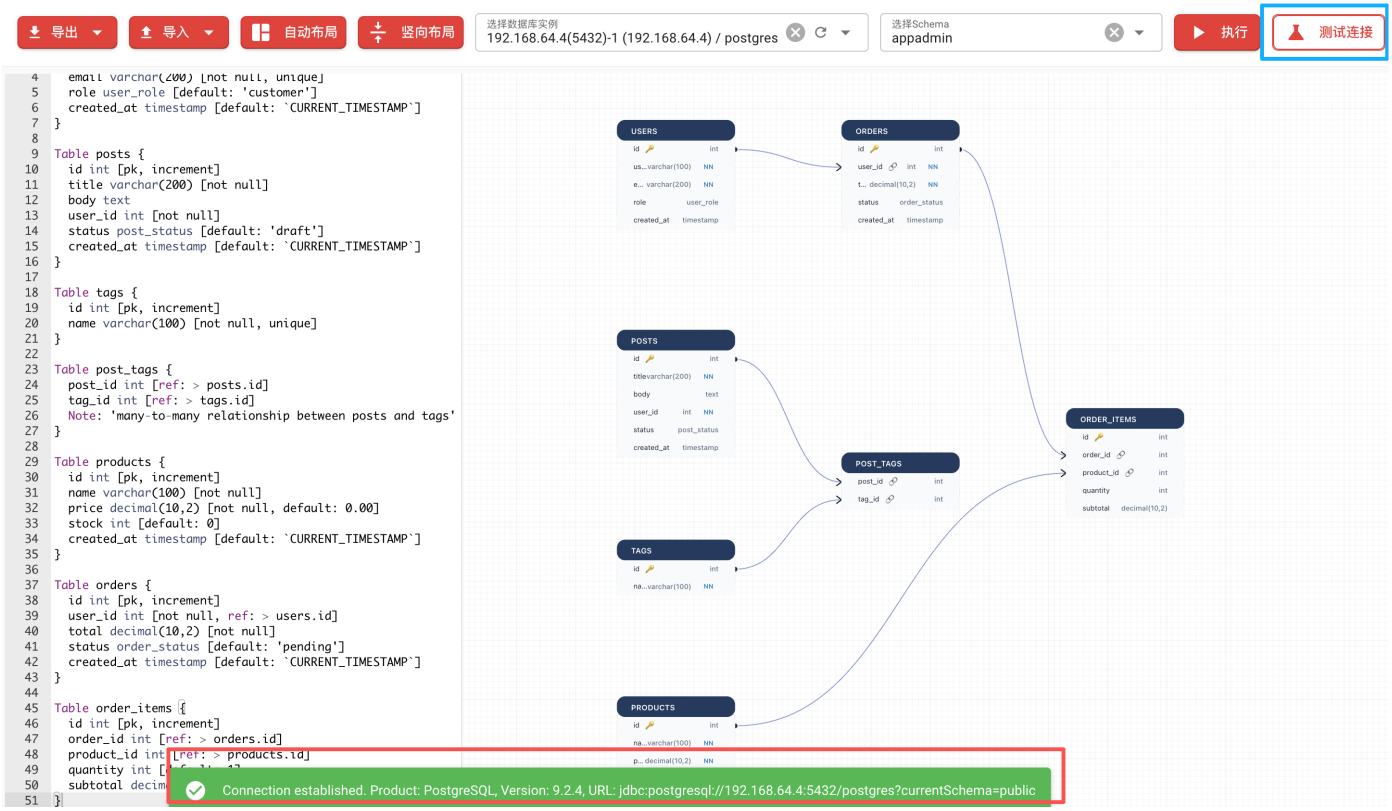

PRODUCTS
id int
na... varchar(100) NN
p... decimal(10,2) NN
stock int
created_at timestamp
USERS
id int
us... varchar(100) NN
e... varchar(200)
role
created_at
POSTS
id
titleva
body
user_id int NN
status post_status
created_at timestamp
TAGS
id int
na... varchar(100) NN
POST_TAGS
post_id int
tag_id int
ORDERS
id int
user_id int NN
t... decimal(10,2) NN
status order_status
created_at timestamp
ORDER_ITEMS
id int
order_id int
product_id int
quantity int
subtotal decimal(10,2)

竖向布局后：

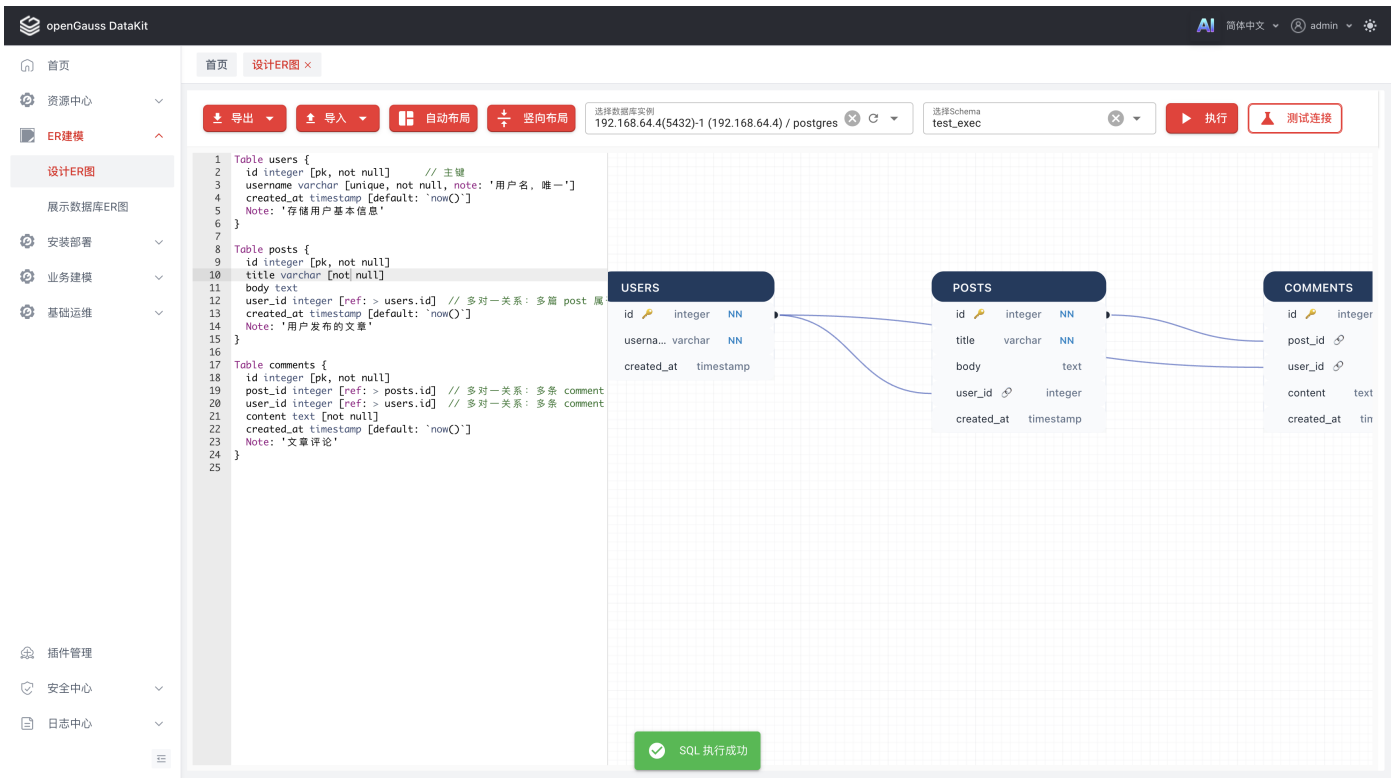


执行到执行数据库

用户在执行到数据库前可以先点击“”检查连接（程序在执行前也会自动检查一次）：



执行到指定数据库的指定schema：



后台输出：

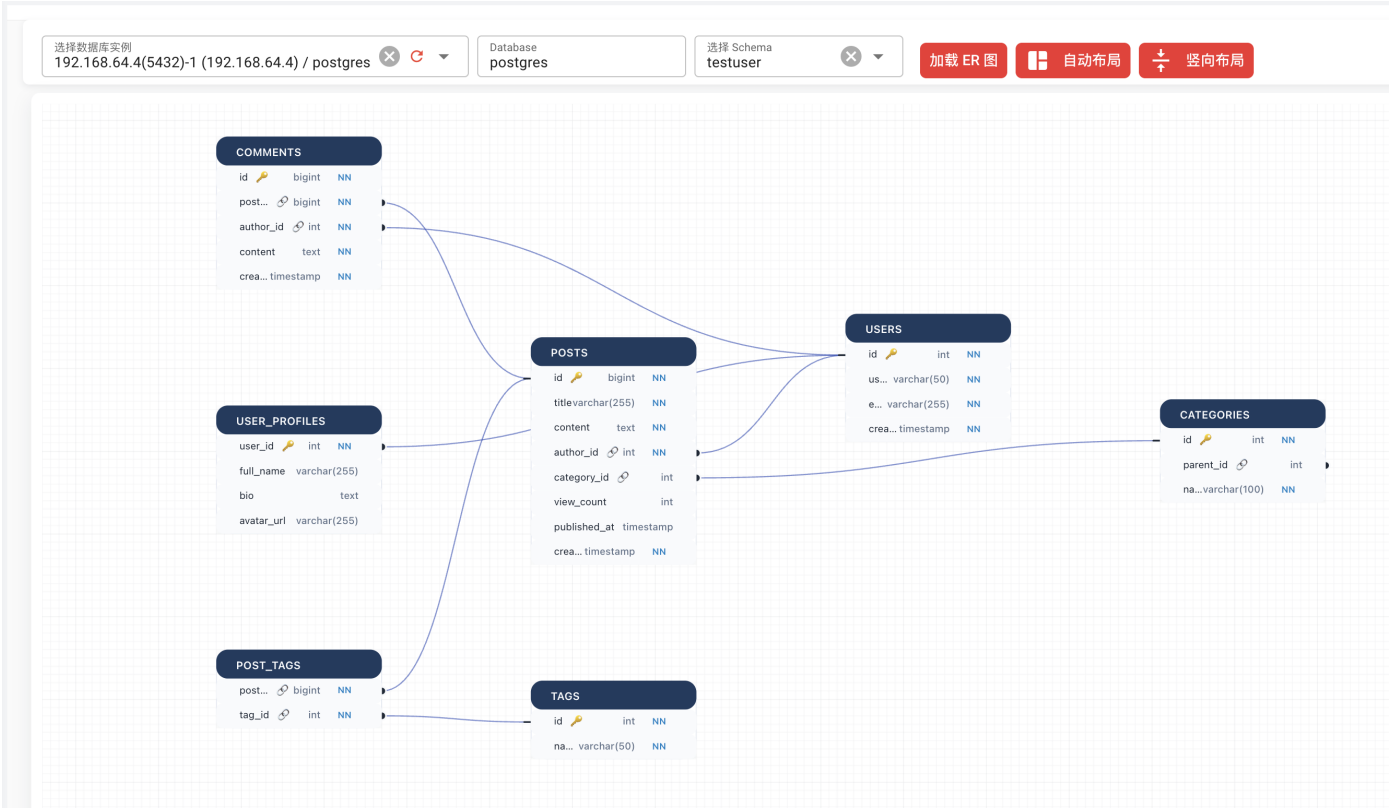
```
2025-10-29 01:31:40 [https-jsse-nio-9494-exec-8] [org.opengauss.admin.plugin.controller.ExecuteController] - [INFO] Received DBML execution request -> clusterId=1955254880234926082,
 nodeId=1955254880650162177, dialect=opengauss, schema=test_exec
2025-10-29 01:31:40 [https-jsse-nio-9494-exec-8] [org.opengauss.admin.plugin.controller.ExecuteController] - [INFO] Generated SQL based on DBML:\nCREATE SCHEMA IF NOT EXISTS "test_exec";\nSET
 search_path TO "test_exec", public;\nCREATE TABLE "users" (\n  "id" INTEGER NOT NULL,\n  "username" VARCHAR(1) NOT NULL UNIQUE,\n  "created_at" TIMESTAMP WITHOUT TIME ZONE NULL DEFAULT now(),\n
 CONSTRAINT "pk_users" PRIMARY KEY ("id")\n);\n\nCOMMENT ON TABLE "users" IS '存储用户基本信息';\nCOMMENT ON COLUMN "users"."username" IS '用户名，唯一';\n\nCREATE TABLE "posts" (\n  "id" INTEGER NOT
 NULL,\n  "title" VARCHAR(1) NOT NULL,\n  "body" TEXT NULL,\n  "user_id" INTEGER NULL,\n  "created_at" TIMESTAMP WITHOUT TIME ZONE NULL DEFAULT now(),\n  CONSTRAINT "pk_posts" PRIMARY KEY
 ("id")\n);\n\nCOMMENT ON TABLE "posts" IS '用户发布的文章';\n\nALTER TABLE "posts" ADD CONSTRAINT "fk_posts_users" FOREIGN KEY ("user_id") REFERENCES "users" ("id");\n\nCREATE TABLE "comments" (\n
 "id" INTEGER NOT NULL,\n  "post_id" INTEGER NULL,\n  "user_id" INTEGER NULL,\n  "content" TEXT NOT NULL,\n  "created_at" TIMESTAMP WITHOUT TIME ZONE NULL DEFAULT now(),\n  CONSTRAINT "pk_comments"
  PRIMARY KEY ("id")\n);\n\nCOMMENT ON TABLE "comments" IS '文章评论';\n\nALTER TABLE "comments" ADD CONSTRAINT "fk_comments_posts" FOREIGN KEY ("post_id") REFERENCES "posts" ("id");\n\nALTER TABLE
 "comments" ADD CONSTRAINT "fk_comments_users" FOREIGN KEY ("user_id") REFERENCES "users" ("id");\n\n
2025-10-29 01:31:40 [https-jsse-nio-9494-exec-8] [org.opengauss.core.v3.ConnectionFactoryImpl] - [INFO] [7d073cdf-3771-4a5c-b8bb-1d561fa15813] Try to connect. IP: 192.168.64.4:5432
2025-10-29 01:31:40 [https-jsse-nio-9494-exec-8] [org.opengauss.core.v3.ConnectionFactoryImpl] - [INFO] [*.*.64.1:62342/*.*.64.4:5432] Connection is established. ID:
 7d073cdf-3771-4a5c-b8bb-1d561fa15813
2025-10-29 01:31:40 [https-jsse-nio-9494-exec-8] [org.opengauss.core.v3.ConnectionFactoryImpl] - [INFO] Connect complete. ID: 7d073cdf-3771-4a5c-b8bb-1d561fa15813
2025-10-29 01:31:42 [https-jsse-nio-9494-exec-8] [org.opengauss.admin.plugin.service.impl.SqlExecutionServiceImpl] - [INFO] SQL script executed successfully and transaction committed.
2025-10-29 01:31:42 [https-jsse-nio-9494-exec-8] [org.opengauss.admin.plugin.controller.ExecuteController] - [INFO] SQL executed successfully -> clusterId=1955254880234926082,
 nodeId=1955254880650162177, schema=test_exec
```

数据库schema展示：

```
openGauss=# SELECT
    table_schema,
    table_name,
    column_name,
    data_type
FROM information_schema.columns
WHERE table_schema = 'test_exec'
ORDER BY table_name, ordinal_position;
openGauss-# openGauss-# openGauss-# openGauss-# openGauss-# openGauss-# openGauss-#  table_schema | table_n
ame | column_name |          data_type
-------------+-------------+-------------+-----------------------------
 test_exec   | comments    | id          | integer
 test_exec   | comments    | post_id     | integer
 test_exec   | comments    | user_id     | integer
 test_exec   | comments    | content     | text
 test_exec   | comments    | created_at  | timestamp without time zone
 test_exec   | posts       | id          | integer
 test_exec   | posts       | title       | character varying
 test_exec   | posts       | body        | text
 test_exec   | posts       | user_id     | integer
 test_exec   | posts       | created_at  | timestamp without time zone
 test_exec   | users       | id          | integer
 test_exec   | users       | username    | character varying
 test_exec   | users       | created_at  | timestamp without time zone
(13 rows)
```

展示数据库er图

选择数据库与schema后，点击加载er图，将展示数据库对于er图：



数据库存储字段：

```
postgres=# SELECT
    table_schema,
    table_name,
    column_name,
    data_type,
    character_maximum_length,
    column_default,
    is_nullable
FROM information_schema.columns
WHERE table_schema = 'testuser'
ORDER BY table_name, ordinal_position;
 table_schema |  table_name   | column_name  |          data_type          | character_maximum_length |             column_default             | is_nullable
--------------+---------------+--------------+-----------------------------+--------------------------+----------------------------------------+-------------
 testuser     | categories    | id           | integer                     |                          | nextval('categories_id_seq'::regclass) | NO
 testuser     | categories    | parent_id    | integer                     |                          |                                        | YES
 testuser     | categories    | name         | character varying           |                      100 |                                        | NO
 testuser     | comments      | id           | bigint                      |                          | nextval('comments_id_seq'::regclass)   | NO
 testuser     | comments      | post_id      | bigint                      |                          |                                        | NO
 testuser     | comments      | author_id    | integer                     |                          |                                        | NO
 testuser     | comments      | content      | text                        |                          |                                        | NO
 testuser     | comments      | created_at   | timestamp without time zone |                          | now()                                  | NO
 testuser     | posts         | id           | bigint                      |                          | nextval('posts_id_seq'::regclass)      | NO
 testuser     | posts         | title        | character varying           |                      255 |                                        | NO
 testuser     | posts         | content      | text                        |                          |                                        | NO
 testuser     | posts         | author_id    | integer                     |                          |                                        | NO
 testuser     | posts         | category_id  | integer                     |                          |                                        | YES
 testuser     | posts         | view_count   | integer                     |                          | 0                                      | YES
 testuser     | posts         | published_at | timestamp without time zone |                          |                                        | YES
 testuser     | posts         | created_at   | timestamp without time zone |                          | now()                                  | NO
 testuser     | post_tags     | post_id      | bigint                      |                          |                                        | NO
 testuser     | post_tags     | tag_id       | integer                     |                          |                                        | NO
 testuser     | tags          | id           | integer                     |                          | nextval('tags_id_seq'::regclass)       | NO
 testuser     | tags          | name         | character varying           |                       50 |                                        | NO
 testuser     | user_profiles | user_id      | integer                     |                          |                                        | NO
 testuser     | user_profiles | full_name    | character varying           |                      255 |                                        | YES
 testuser     | user_profiles | bio          | text                        |                          |                                        | YES
 testuser     | user_profiles | avatar_url   | character varying           |                      255 | '/images/default.png'::character varying | YES
 testuser     | users         | id           | integer                     |                          | nextval('users_id_seq'::regclass)      | NO
 testuser     | users         | username     | character varying           |                       50 |                                        | NO
 testuser     | users         | email        | character varying           |                      255 |                                        | NO
 testuser     | users         | created_at   | timestamp without time zone |                          | now()                                  | NO
(28 rows)

postgres=#
```