

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

```

1 CREATE TABLE users (
2     id INT PRIMARY KEY AUTO_INCREMENT,
3     username VARCHAR(100) NOT NULL,
4     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
5 );
6
7 CREATE TABLE posts (
8     id INT PRIMARY KEY AUTO_INCREMENT,
9     title VARCHAR(200) NOT NULL,
10    body TEXT,
11    user_id INT,
12    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
13    CONSTRAINT fk_posts_user FOREIGN KEY (user_id) REFERENCES users(id)
14 );
15
16 CREATE TABLE comments (
17     id INT PRIMARY KEY AUTO_INCREMENT,
18     post_id INT,
19     user_id INT,
20     content TEXT NOT NULL,
21     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
22     CONSTRAINT fk_comments_post FOREIGN KEY (post_id) REFERENCES posts(id),
23     CONSTRAINT fk_comments_user FOREIGN KEY (user_id) REFERENCES users(id)
24 );
25

```

解析的DBML语句：

选择数据库实例
192.168.64.4(5432)-1 (192.168.64.4) / postgres

选择Schema
appadmin

导出 导入 自动布局 竖向布局 执行 测试连接

```

1 Table users {
2     id int [pk, increment]
3     username varchar(100)
4     created_at timestamp [default: `CURRENT_TIMESTAMP`]
5 }
6
7 Table posts {
8     id int [pk, increment]
9     title varchar(200)
10    body text
11    user_id int
12    created_at timestamp [default: `CURRENT_TIMESTAMP`]
13 }
14
15 Table comments {
16     id int [pk, increment]
17     post_id int
18     user_id int
19     content text
20     created_at timestamp [default: `CURRENT_TIMESTAMP`]
21 }
22
23 Ref: posts.user_id > users.id
24 Ref: comments.post_id > posts.id
25 Ref: comments.user_id > users.id
26

```

SQL文件导入成功

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：

The screenshot shows a database management interface with the following components:

- Toolbar:** Includes buttons for Export (导出), Import (导入), Auto Layout (自动布局), Vertical Layout (竖向布局), and a connection status bar (192.168.64.4(5432)-1 (192.168.64.4) / postgres).
- Code Editor:** Displays the DBML code for three tables: users, posts, and comments.
- ER Diagram:** Shows the relationships between the three tables. The users table has a one-to-many relationship with the posts table, and the posts table has a one-to-many relationship with the comments table.

```

1 Table users {
2   id int [pk, increment]
3   username varchar(100)
4   created_at timestamp [default: `CURRENT_TIMESTAMP`]
5 }
6
7 Table posts {
8   id int [pk, increment]
9   title varchar(200)
10  body text
11  user_id int
12  created_at timestamp [default: `CURRENT_TIMESTAMP`]
13 }
14
15 Table comments {
16   id int [pk, increment]
17   post_id int
18   user_id int
19   content text
20   created_at timestamp [default: `CURRENT_TIMESTAMP`]
21 }
22
23 Ref: posts.user_id > users.id
24 Ref: comments.post_id > posts.id
25 Ref: comments.user_id > users.id
26

```

导出的openGaussSQL：

er-export-opengauss (16).sql UNREGISTERED

```
1 CREATE TABLE users (
2     id SERIAL PRIMARY KEY,
3     username VARCHAR(100),
4     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
5 );
6
7 CREATE TABLE posts (
8     id SERIAL PRIMARY KEY,
9     title VARCHAR(200),
10    body TEXT,
11    user_id INTEGER,
12    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
13    CONSTRAINT fk_posts_users FOREIGN KEY (user_id) REFERENCES users(id)
14 );
15
16 CREATE TABLE comments (
17     id SERIAL PRIMARY KEY,
18     post_id INTEGER,
19     user_id INTEGER,
20     content TEXT,
21     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
22     CONSTRAINT fk_comments_posts FOREIGN KEY (post_id) REFERENCES posts(id),
23     CONSTRAINT fk_comments_users FOREIGN KEY (user_id) REFERENCES users(id)
24 );
25
```

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

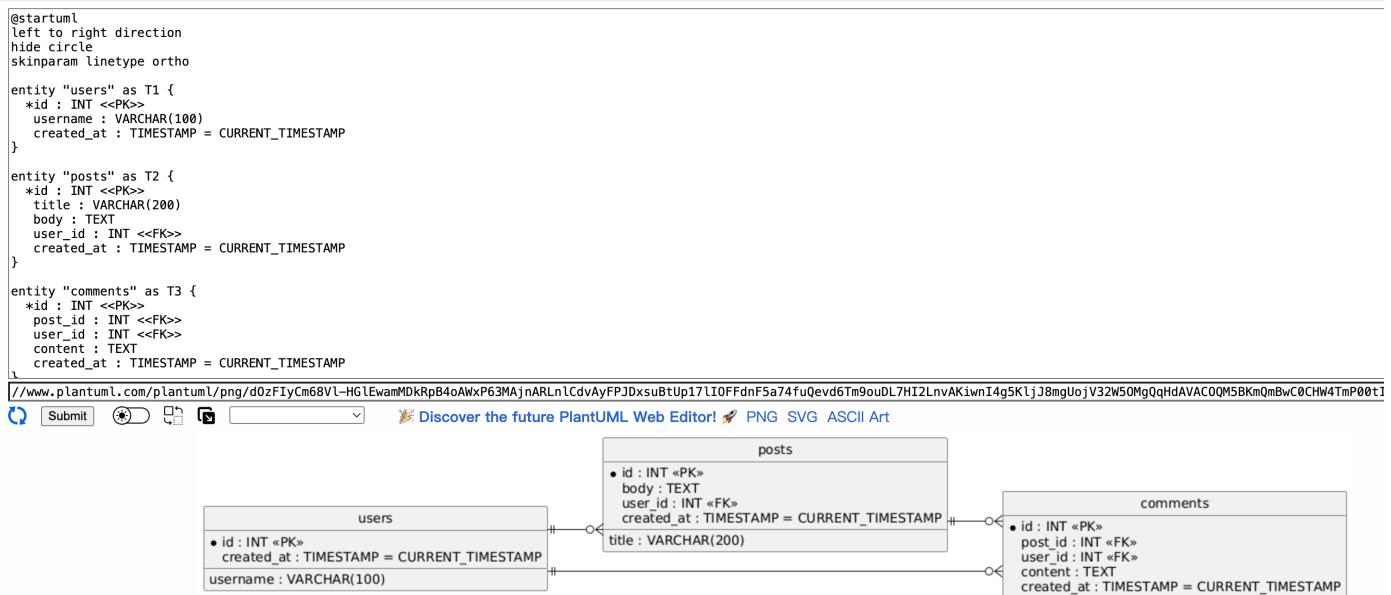
```

1 @startuml
2 left to right direction
3 hide circle
4 skinparam linetype ortho
5
6 entity "users" as T1 {
7     *id : INT <<PK>>
8     username : VARCHAR(100)
9     created_at : TIMESTAMP = CURRENT_TIMESTAMP
10 }
11
12 entity "posts" as T2 {
13     *id : INT <<PK>>
14     title : VARCHAR(200)
15     body : TEXT
16     user_id : INT <<FK>>
17     created_at : TIMESTAMP = CURRENT_TIMESTAMP
18 }
19
20 entity "comments" as T3 {
21     *id : INT <<PK>>
22     post_id : INT <<FK>>
23     user_id : INT <<FK>>
24     content : TEXT
25     created_at : TIMESTAMP = CURRENT_TIMESTAMP
26 }
27
28 T1 ||--o{ T2
29 T2 ||--o{ T3
30 T1 ||--o{ T3
31 @enduml
32

```

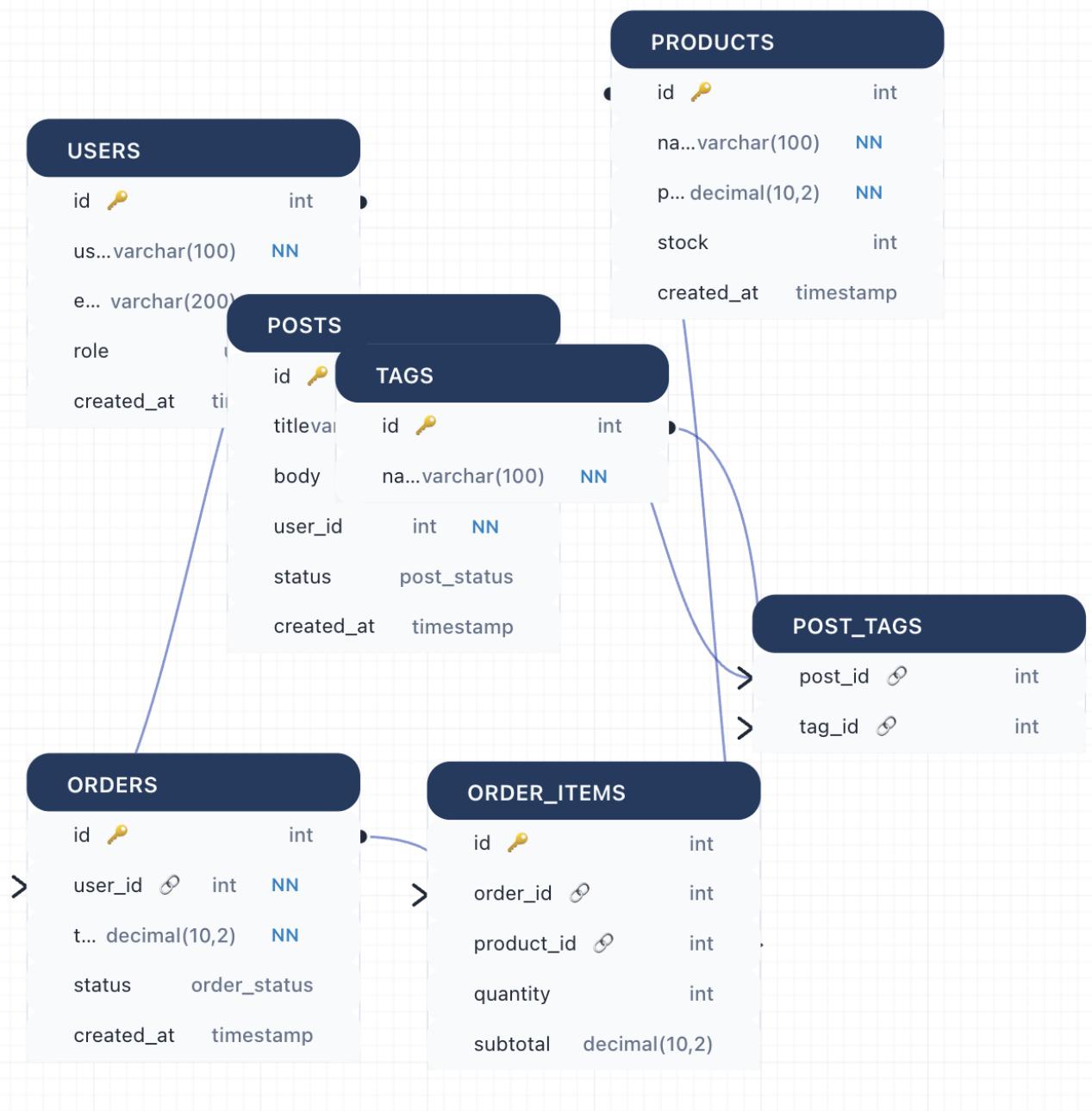
Line 24, Column 18 Spaces: 2 Plain Text

plantUML代码解析后生成图片：

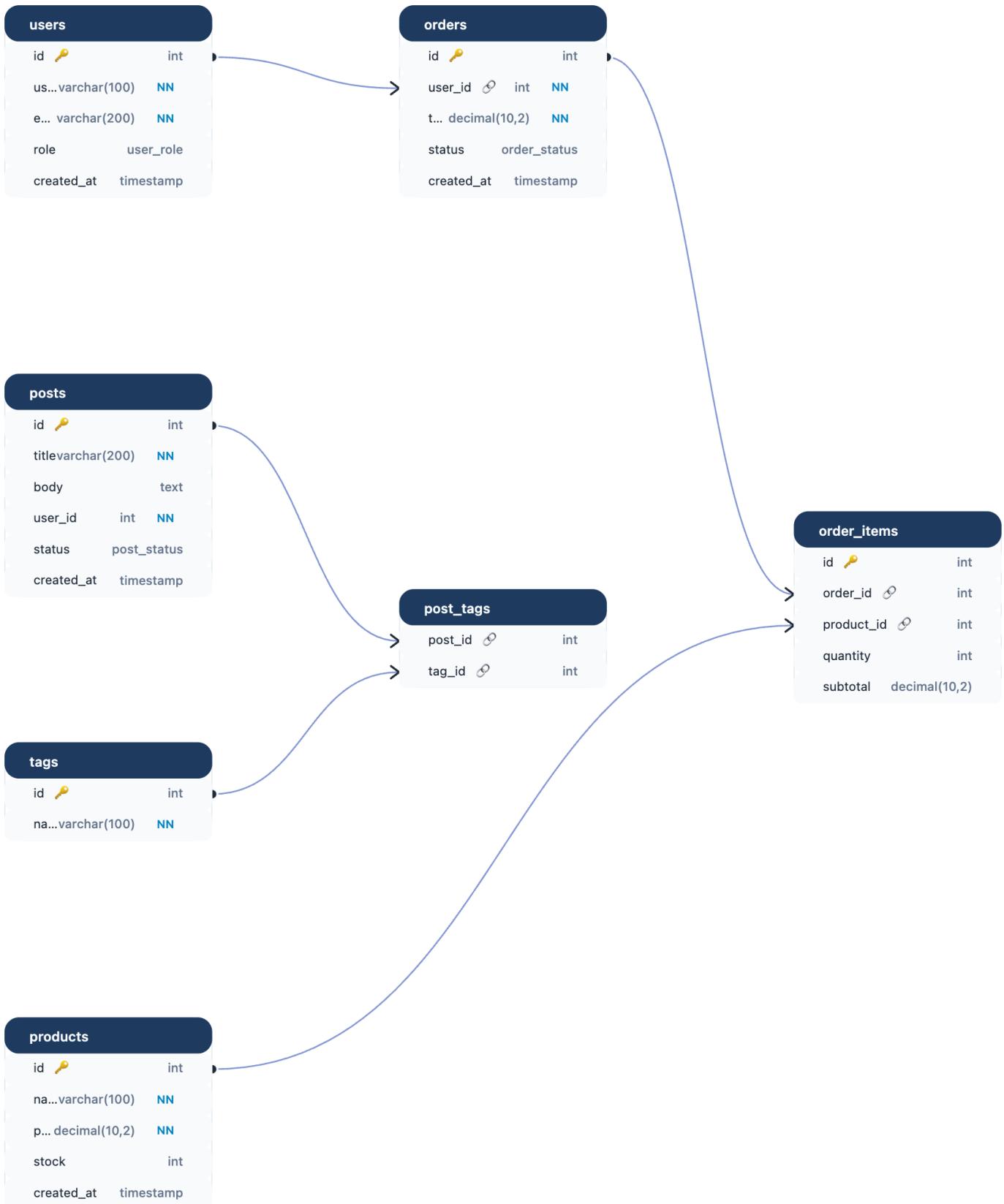


自动布局

初始杂乱无章的er图：

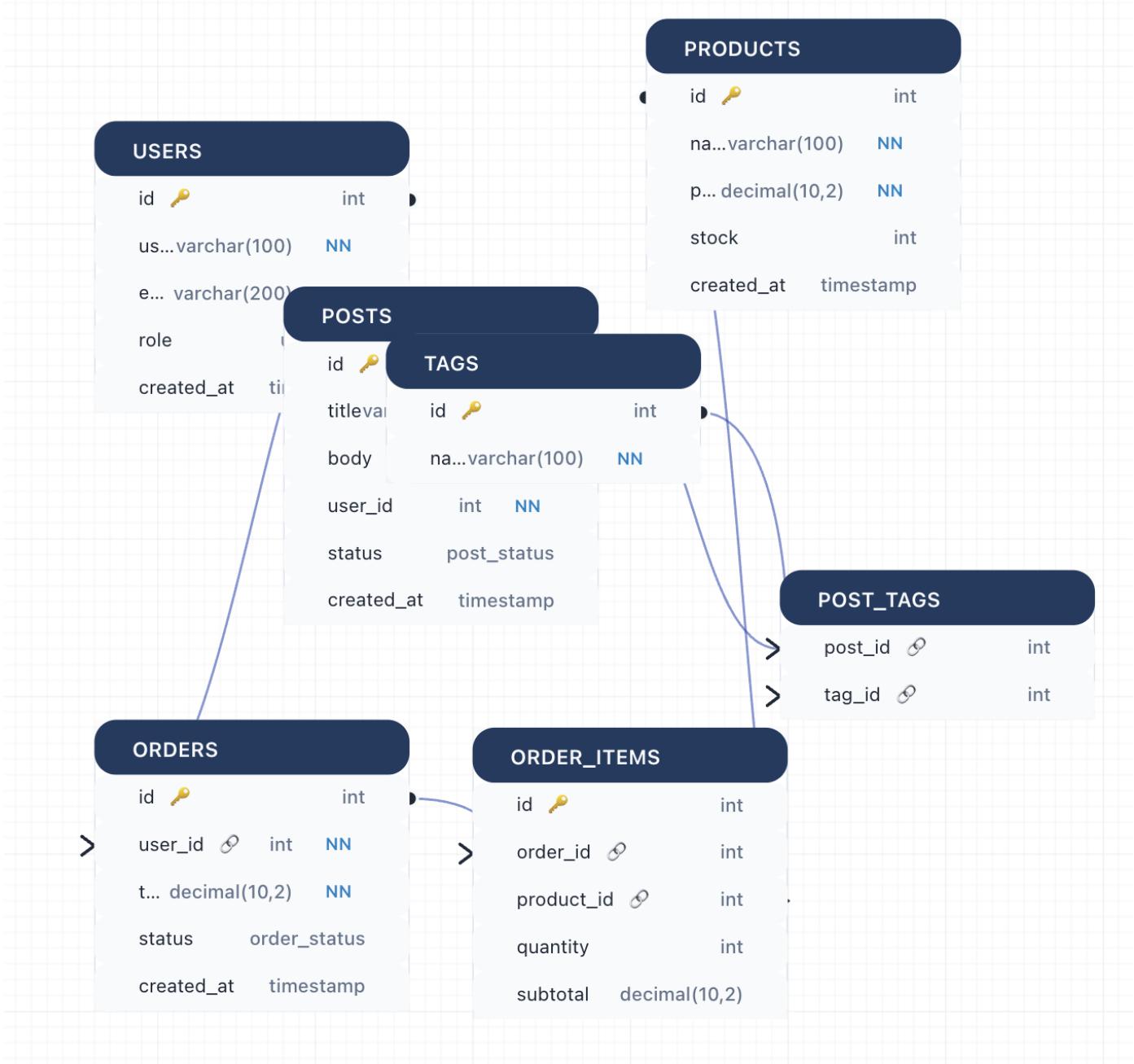


自动布局后：

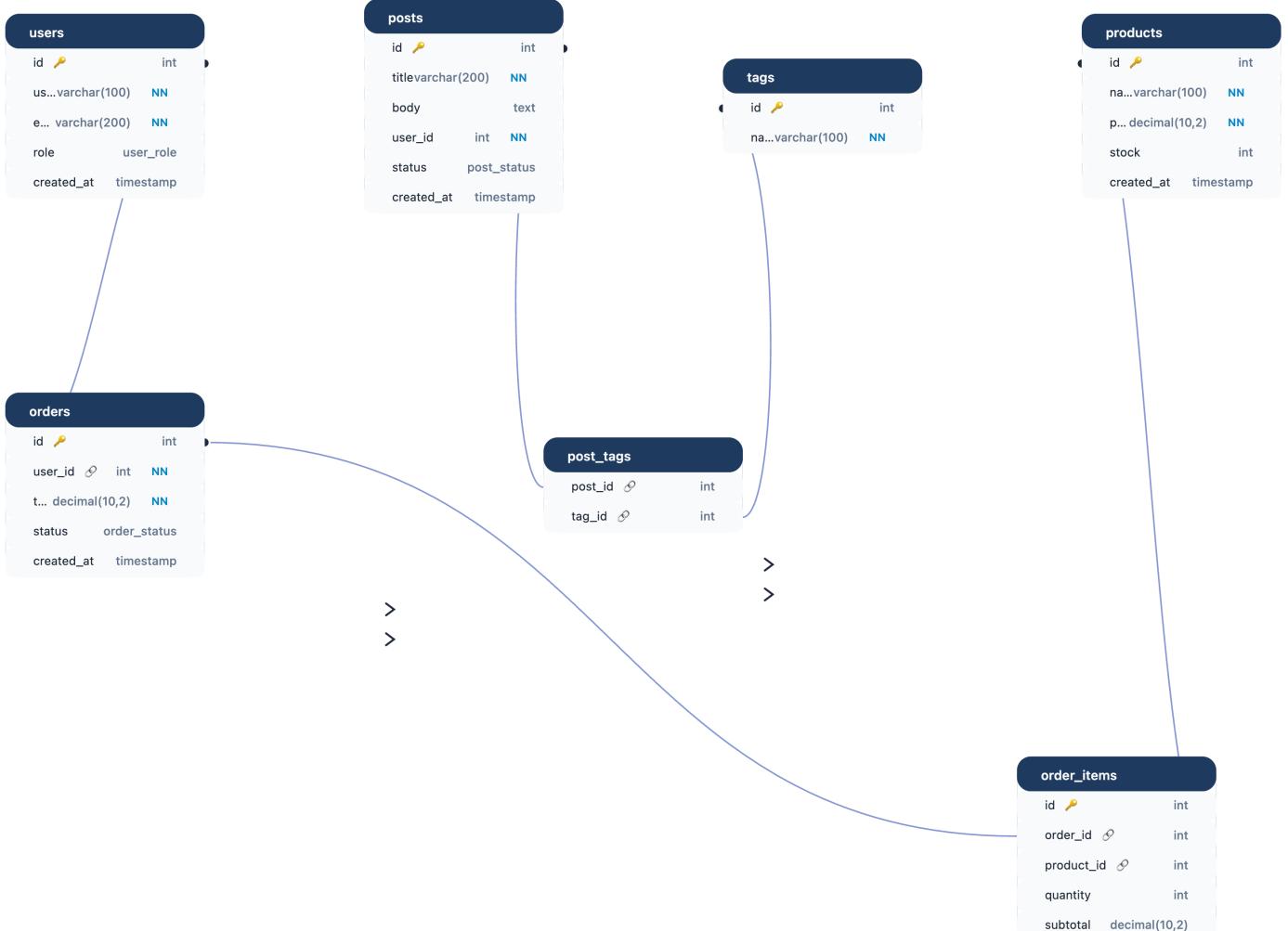


竖向布局

初始杂乱无章的er图：



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）：

选择数据库实例 192.168.64.4(5432)-1 (192.168.64.4) / postgres

选择Schema appadmin

执行 测试连接

```

4 email varchar(200) [not null, unique]
5 role user_role [default: 'customer']
6 created_at timestamp [default: 'CURRENT_TIMESTAMP']
7 }

8
9 Table posts {
10 id int [pk, increment]
11 title varchar(200) [not null]
12 body text
13 user_id int [not null]
14 status post_status [default: 'draft']
15 created_at timestamp [default: 'CURRENT_TIMESTAMP']
16 }

17
18 Table tags {
19 id int [pk, increment]
20 name varchar(100) [not null, unique]
21 }

22
23 Table post_tags {
24 post_id int [ref: > posts.id]
25 tag_id int [ref: > tags.id]
26 Note: 'many-to-many relationship between posts and tags'
27 }

28
29 Table products {
30 id int [pk, increment]
31 name varchar(100) [not null]
32 price decimal(10,2) [not null, default: 0.00]
33 stock int [default: 0]
34 created_at timestamp [default: 'CURRENT_TIMESTAMP']
35 }

36
37 Table orders {
38 id int [pk, increment]
39 user_id int [not null, ref: > users.id]
40 total decimal(10,2) [not null]
41 status order_status [default: 'pending']
42 created_at timestamp [default: 'CURRENT_TIMESTAMP']
43 }

44
45 Table order_items [
46 id int [pk, increment]
47 order_id int [ref: > orders.id]
48 product_id int [ref: > products.id]
49 quantity int [default: 1]
50 subtotal decimal
51 ]

```

Connection established. Product: PostgreSQL, Version: 9.2.4, URL: jdbc:postgresql://192.168.64.4:5432/postgres?currentSchema=public

执行到指定数据库的指定schema:

openGauss DataKit

首页 设计ER图

导出 导入 自动布局 垂直布局 选择数据库实例 192.168.64.4(5432)-1 (192.168.64.4) / postgres 选择Schema test_exec 执行 测试连接

```

1 Table users {
2 id integer [pk, not null] // 主键
3 username varchar [unique, not null, note: '用户名, 唯一']
4 created_at timestamp [default: 'now()']
5 Note: '存储用户基本信息'
6 }

7
8 Table posts {
9 id integer [pk, not null]
10 title varchar [not null]
11 body text
12 user_id integer [ref: > users.id] // 多对一关系: 多篇 post 属于一个 user
13 created_at timestamp [default: 'now()']
14 Note: '用户发布的文章'
15 }

16
17 Table comments {
18 id integer [pk, not null]
19 post_id integer [ref: > posts.id] // 多对一关系: 多条 comment 属于一篇 post
20 user_id integer [ref: > users.id] // 多对一关系: 多条评论属于一个 user
21 content text [not null]
22 created_at timestamp [default: 'now()']
23 Note: '文章评论'
24 }

```

SQL 执行成功

后台输出:

```

2025-10-29 01:31:48 [https-jsse-nio-9494-exec-8] [org.opengauss.admin.plugin.controller.ExecuteController] - [INFO] Received DBML execution request -> clusterId=1955254880234926082,
nodeId=1955254880650162177, dialect=openauss, schema=test_exec
2025-10-29 01:31:48 [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);\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);\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);\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 01:31:48 [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 01:31:48 [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 01:31:48 [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：

```

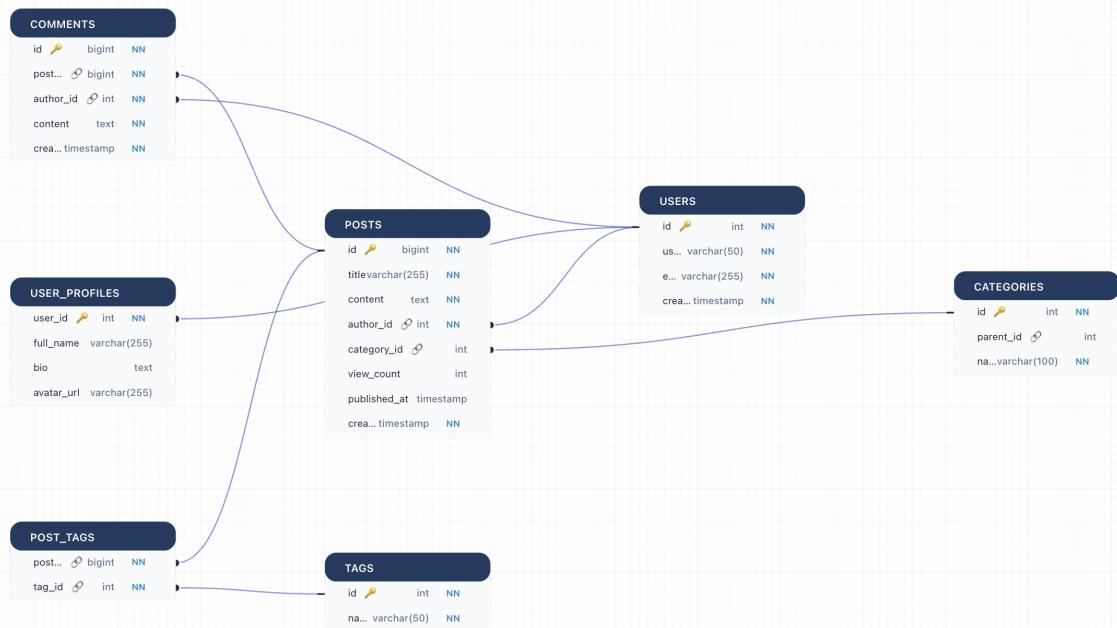
The connection to the service was successfully completed successfully.

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# 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：

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	nextval('comments_id_seq'::regclass)	NO
testuser	comments	id	bigint			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		now()	YES
testuser	posts	created_at	timestamp without time zone			NO
testuser	post_tags	post_id	bigint			NO
testuser	post_tags	tag_id	integer			NO
testuser	tags	id	integer	50	nextval('tags_id_seq'::regclass)	NO
testuser	tags	name	character varying			NO
testuser	user_profiles	user_id	integer	255		NO
testuser	user_profiles	full_name	character varying			YES
testuser	user_profiles	bio	text			YES
testuser	user_profiles	avatar_url	character varying	255	'/images/default.png'::character varying	YES
testuser	users	username	character varying	50	nextval('users_id_seq'::regclass)	NO
testuser	users	email	character varying	255		NO
testuser	users	created_at	timestamp without time zone		now()	NO

(28 rows)

postgres=#