Warmhouse

Admin Dashboard - Management system

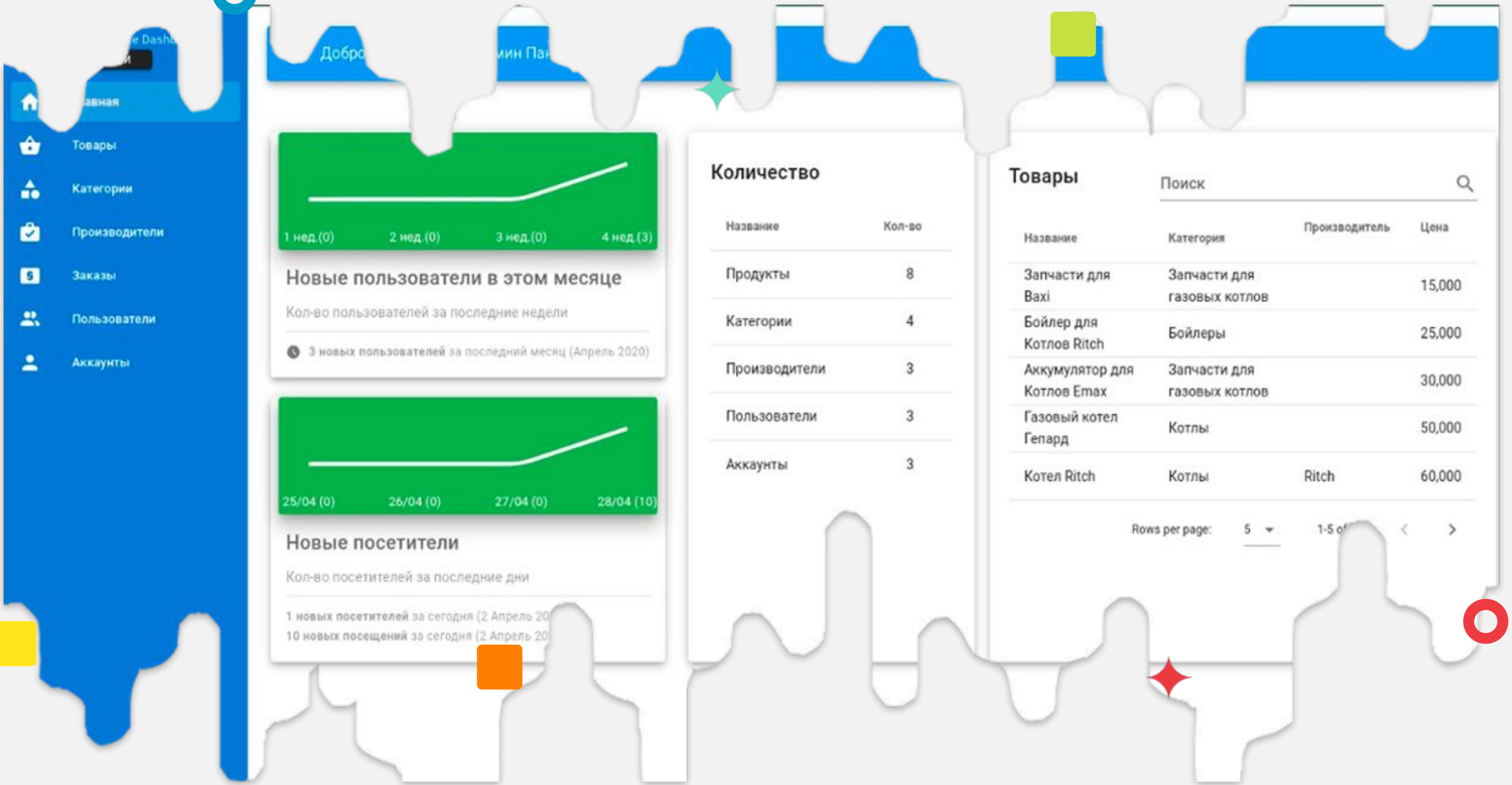
The project for Database Management system course 1

Using Oracle database, we created Admin Dashboard - Management system for the online store, Warmhouse

About project

Management system to solve business problems

The project is a management system of Warmhouse online store, which is located in Aktobe/Kazakhstan. The management system was created to solve the problems of this business. The online store is under construction. We have created an admin dashboard for this store. This panel is for managing goods, categories, users, orders, visitors and for viewing some statistics. We want the administrators of this store to be able to manage this data, and based on the proposed statistics, increase sales and do analytics.



Новые пользователи в этом месяце

Кол-во пользователей за последние недели

3 новых пользователей за последний месяц (Апрель 2020)

Новые посетители

Кол-во посетителей за последние дни

1 новых посетителей за сегодня (2 Апрель 20

10 новых посещений за сегодня (2 Апрель 20

Количество

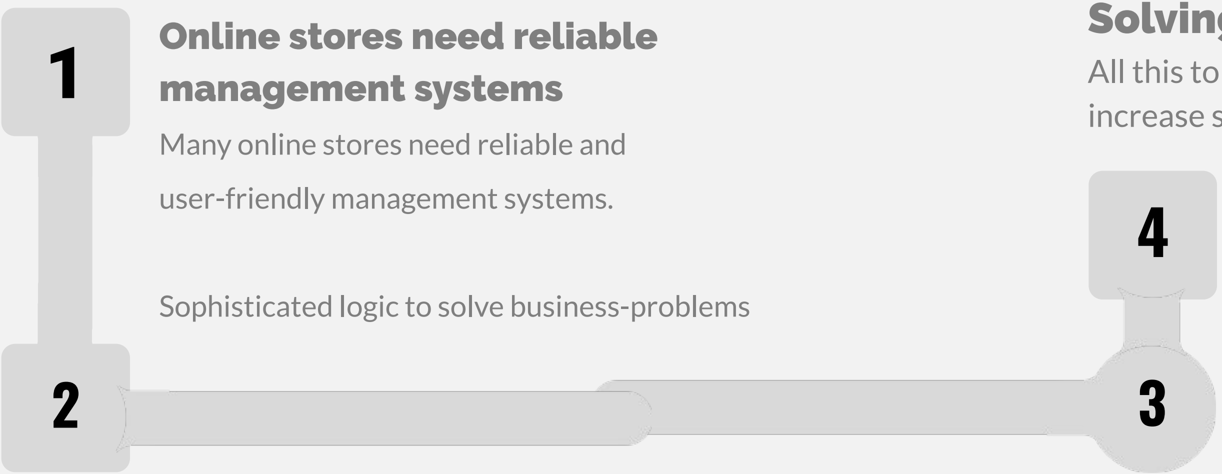
Название	Кол-во
Продукты	8
Категории	4
Производители	3
Пользователи	3
Аккаунты	3

Товары

Название	Категория	Производитель	Цена
Запчасти для Baxi	Запчасти для газовых котлов		15,000
Бойлер для Котлов Ritch	Бойлеры		25,000
Аккумулятор для Котлов Emax	Запчасти для газовых котлов		30,000
Газовый котел Гепард	Котлы		50,000
Котел Ritch	Котлы	Ritch	60,000

Idea - Functions

Where did we get the idea for our project ?



Online stores need reliable management systems

Many online stores need reliable and user-friendly management systems.

Sophisticated logic to solve business-problems

Database for storing many types of data

Records of users, customers, orders, purchases of each product, storage of product information, number of anonymous visitors, etc.

Optimization of logic to increase application speed

Solving business-tasks

All this to solve problems in business and to increase sales.

Meet our team

Makhmudov Bislam

Web Developer

Frontend part of the Project

Zhunussov Arsentiy

Web developer

Backend and Frontend part of the project. Webapp logic

Nakip Dias

Developer - Designer

Webapp design

Project Features

Authorization System for both sides

Authorization system for administrators where we use DBA users as administrators

Authorization system for end users, customers

Storing a wide variety of data types

Storing data on products, users, categories, manufacturers, number of visitors, administrators

Data management

Ability to create, delete, edit and view data

Providing statistics about users, visitors

Statistics are used to view the result of sales, analytics.

Tools

Used tools in the project

PHP for the Backend part

We use PHP programming language for the backend part of our project. We made logic of the Management system

Vue.js - JavaScript for the Frontend part

We use Vue.js framework for the frontend part of our project. Using this, we created awesome interface for the end users

OCI8 Extension for integrating Oracle DB

We use this library in PHP for work with Oracle database.

Oracle Database 11g Express Edition

We use this version of Oracle Database in our project

Oracle SQL Developer 19.4

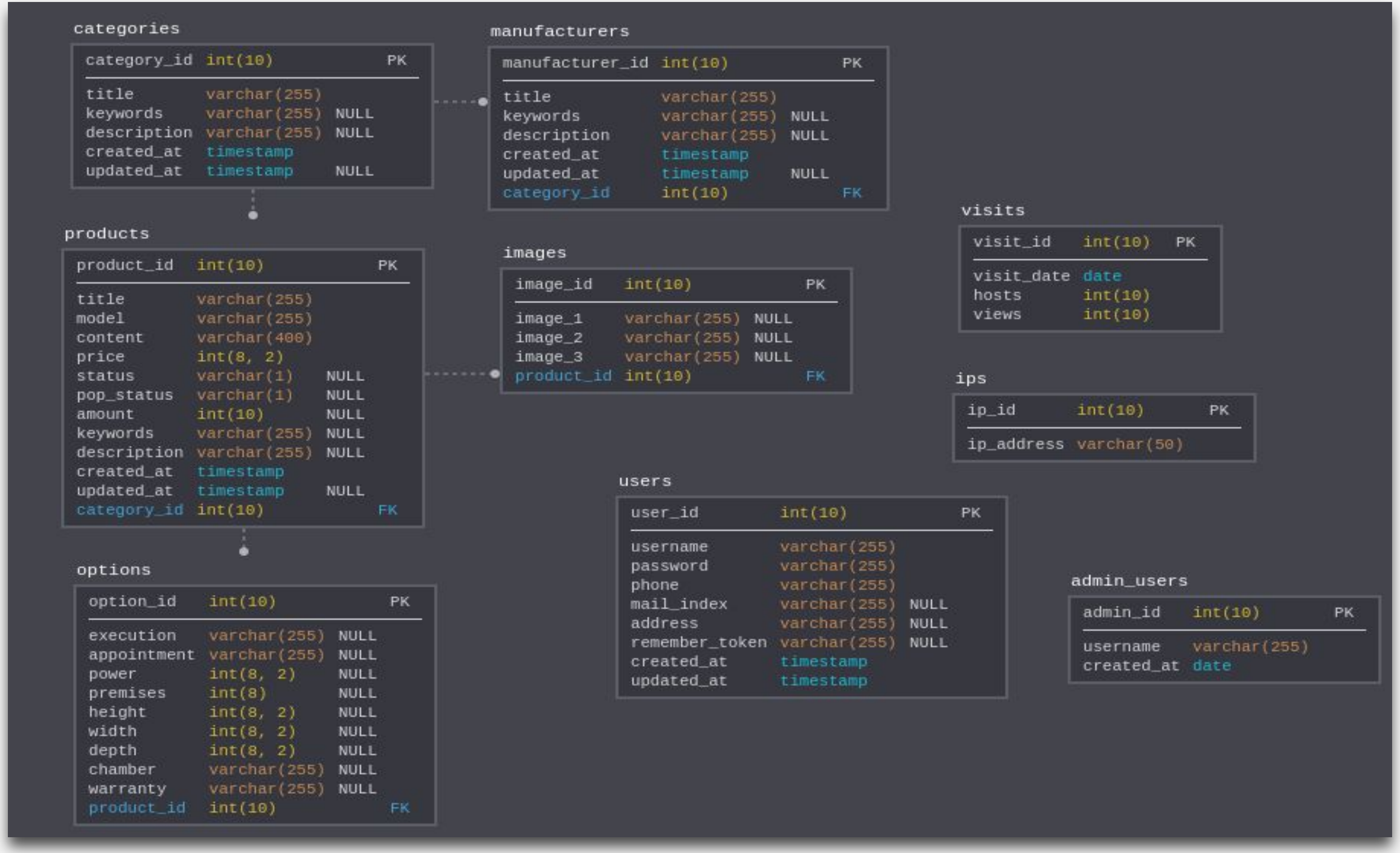
Oracle SQL Developer app for designing Database

Vuetify for the Design part

We use Vuetify extension for the design part of our project

E--R Diagram

Tables and relationships



categories

category_id	int(10)		PK
title	varchar(255)		
keywords	varchar(255)	NULL	
description	varchar(255)	NULL	
created_at	timestamp		
updated_at	timestamp	NULL	

manufacturers

manufacturer_id	int(10)		PK
title	varchar(255)		
keywords	varchar(255)	NULL	
description	varchar(255)	NULL	
created_at	timestamp		
updated_at	timestamp	NULL	
category_id	int(10)		FK

products

product_id	int(10)		PK
title	varchar(255)		
model	varchar(255)		
content	varchar(400)		
price	int(8, 2)		
status	varchar(1)	NULL	
pop_status	varchar(1)	NULL	
amount	int(10)	NULL	
keywords	varchar(255)	NULL	
description	varchar(255)	NULL	
created_at	timestamp		
updated_at	timestamp	NULL	
category_id	int(10)		FK

images

image_id	int(10)		PK
image_1	varchar(255)	NULL	
image_2	varchar(255)	NULL	
image_3	varchar(255)	NULL	
product_id	int(10)		FK

visits

visit_id	int(10)	PK
visit_date	date	
hosts	int(10)	
views	int(10)	

ips

ip_id	int(10)	PK
ip_address	varchar(50)	

options

option_id	int(10)		PK
execution	varchar(255)	NULL	
appointment	varchar(255)	NULL	
power	int(8, 2)	NULL	
premises	int(8)	NULL	
height	int(8, 2)	NULL	
width	int(8, 2)	NULL	
depth	int(8, 2)	NULL	
chamber	varchar(255)	NULL	
warranty	varchar(255)	NULL	
product_id	int(10)		FK

users

user_id	int(10)		PK
username	varchar(255)		
password	varchar(255)		
phone	varchar(255)		
mail_index	varchar(255)	NULL	
address	varchar(255)	NULL	
remember_token	varchar(255)	NULL	
created_at	timestamp		
updated_at	timestamp		

admin_users

admin_id	int(10)	PK
username	varchar(255)	
created_at	date	

CONCEPTUAL LEVEL

- **Used primary keys on each table, for ID column**
  - **product_id - products**
  - **category_id - categories**
  - **manufacturer_id - manufacturers**
  - **image_id - images**
  - **option_id - options**
  - **user_id - users**
  - **ip_id - ips**
  - **visit_id - visits**
  - **admin_id - admin_users**

- **Used sequences for ID column on each table**
  - **products_id_seq**
  - **categories_id_seq**
  - **etc.**

- **Used not null constraints on each column**
  - **products_title_nn**
  - **products_price_nn**
  - **etc.**

- **Used triggers for auto incrementing in ID columns**
  - **categories_id_trg**
  - **images_id_trg**
  - **etc.**

- **Used foreign keys**
  - **product_id - on images, options**
  - **category_id - on products, manufacturers**

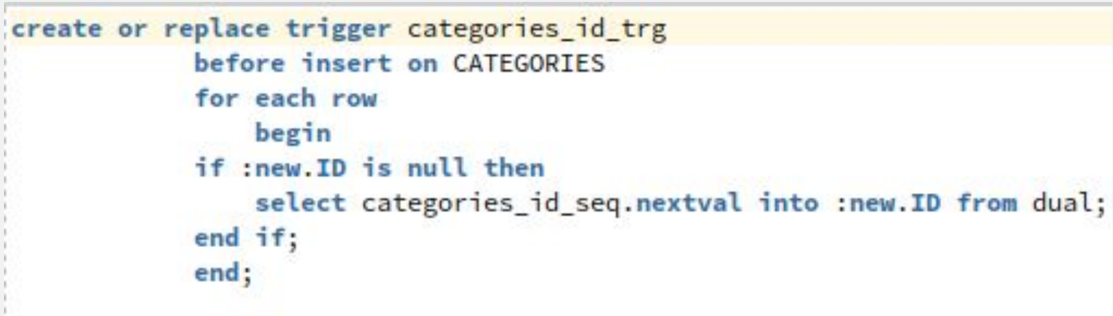
- **Used indexes**
  (Custom indexes, not including auto indexes on columns with primary key and unique)
  - **lower_user_username_idx**

- **Used check constraints with regexp**
  - **users_username_chk with regexp for checking that the value contains only the letters of the English and Russian alphabets**

- **Used view**
  - **products_view which contains data from products, categories, manufacturers using joins**

Example of using triggers

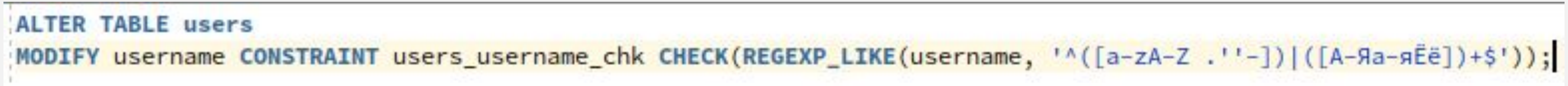
CATEGORIES_ID_TRG



```
create or replace trigger categories_id_trg
            before insert on CATEGORIES
            for each row
                begin
            if :new.ID is null then
                select categories_id_seq.nextval into :new.ID from dual;
            end if;
            end;
```

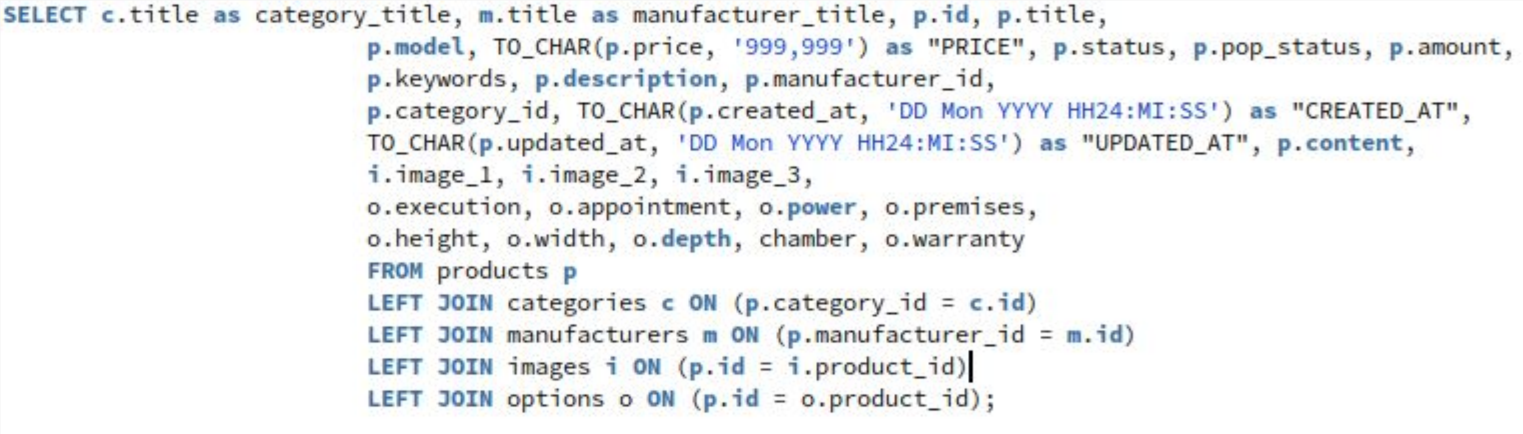
Example of using check and regexp

users_username_chk



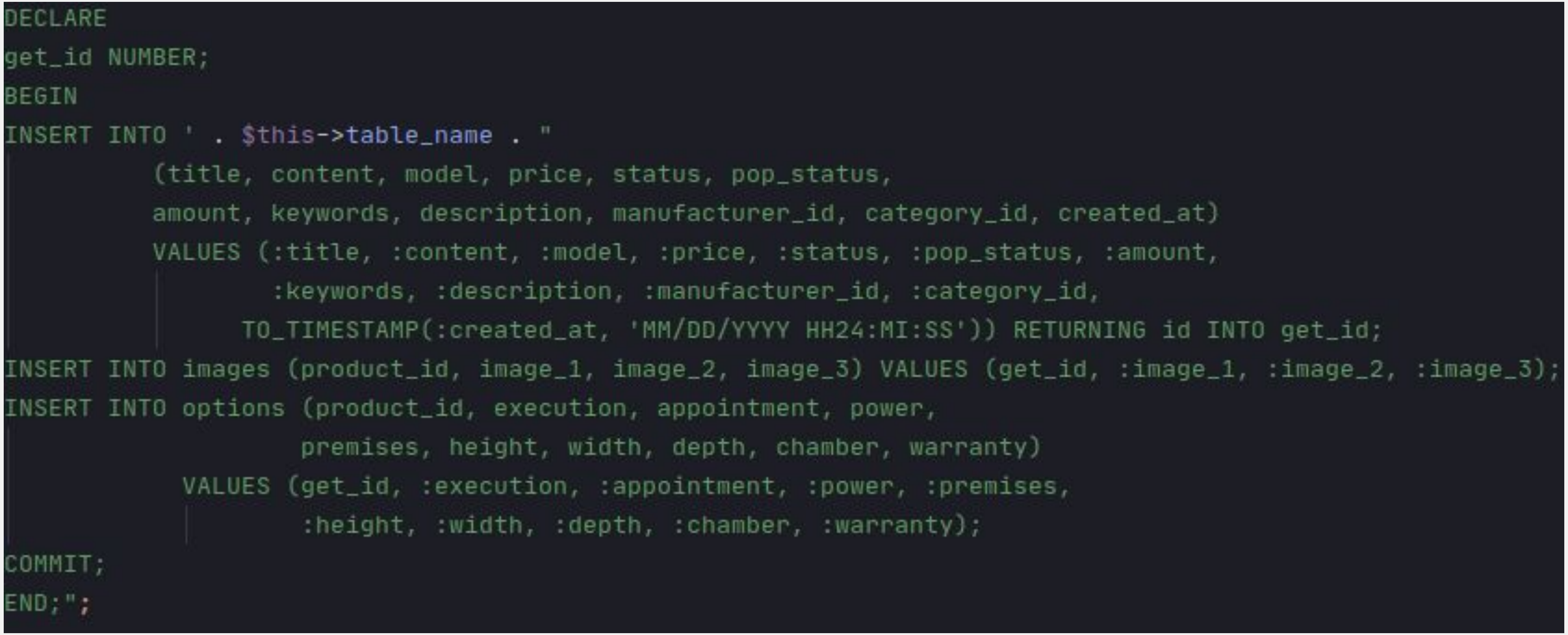
```
ALTER TABLE users
MODIFY username CONSTRAINT users_username_chk CHECK(REGEXP_LIKE(username, '^([a-zA-Z .''-])|([А-Яа-яЁё])+$'));
```

Example of using joins



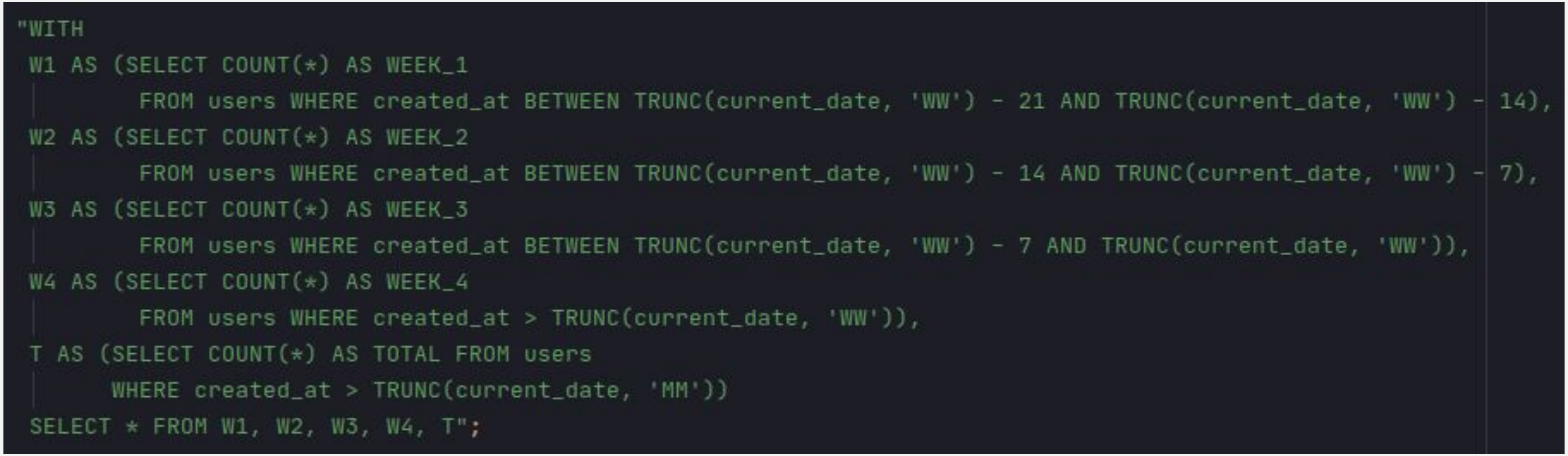
```
SELECT c.title as category_title, m.title as manufacturer_title, p.id, p.title,
                          p.model, TO_CHAR(p.price, '999,999') as "PRICE", p.status, p.pop_status, p.amount,
                          p.keywords, p.description, p.manufacturer_id,
                          p.category_id, TO_CHAR(p.created_at, 'DD Mon YYYY HH24:MI:SS') as "CREATED_AT",
                          TO_CHAR(p.updated_at, 'DD Mon YYYY HH24:MI:SS') as "UPDATED_AT", p.content,
                          i.image_1, i.image_2, i.image_3,
                          o.execution, o.appointment, o.power, o.premises,
                          o.height, o.width, o.depth, chamber, o.warranty
                          FROM products p
                          LEFT JOIN categories c ON (p.category_id = c.id)
                          LEFT JOIN manufacturers m ON (p.manufacturer_id = m.id)
                          LEFT JOIN images i ON (p.id = i.product_id)
                          LEFT JOIN options o ON (p.id = o.product_id);
```

Example of using procedure



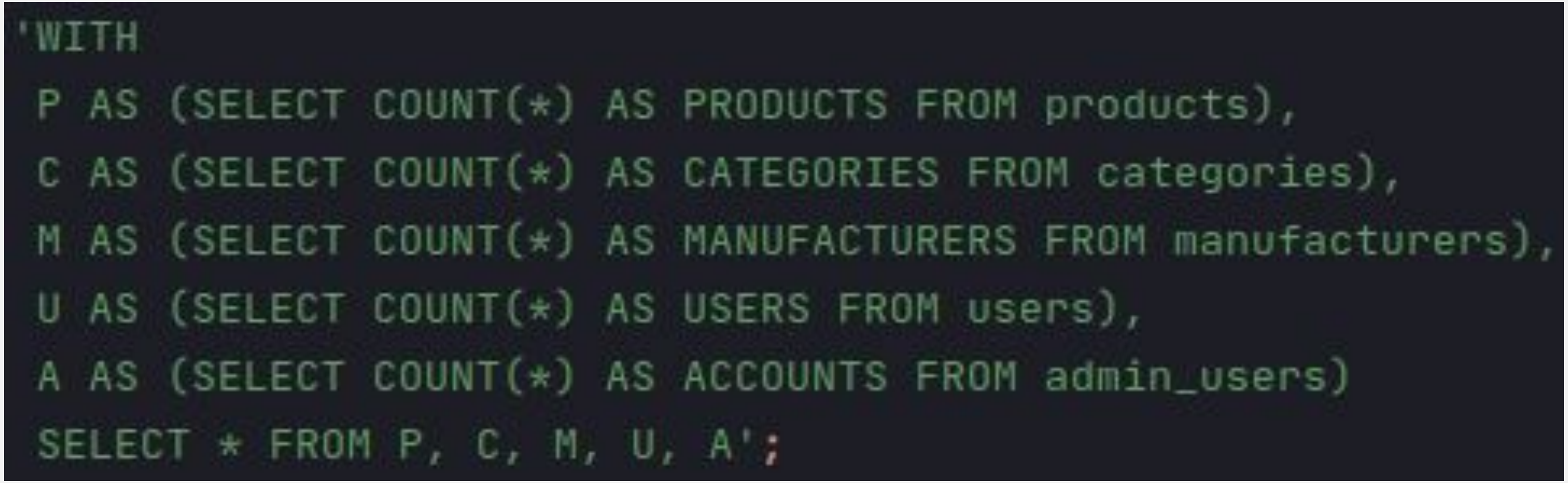
```
DECLARE
get_id NUMBER;
BEGIN
INSERT INTO ' . $this->table_name . "
          (title, content, model, price, status, pop_status,
          amount, keywords, description, manufacturer_id, category_id, created_at)
          VALUES (:title, :content, :model, :price, :status, :pop_status, :amount,
                  :keywords, :description, :manufacturer_id, :category_id,
                TO_TIMESTAMP(:created_at, 'MM/DD/YYYY HH24:MI:SS')) RETURNING id INTO get_id;
INSERT INTO images (product_id, image_1, image_2, image_3) VALUES (get_id, :image_1, :image_2, :image_3);
INSERT INTO options (product_id, execution, appointment, power,
                    premises, height, width, depth, chamber, warranty)
            VALUES (get_id, :execution, :appointment, :power, :premises,
                    :height, :width, :depth, :chamber, :warranty);
COMMIT;
END;";
```

Example of using ‘with’ clause



```
"WITH
 W1 AS (SELECT COUNT(*) AS WEEK_1
        FROM users WHERE created_at BETWEEN TRUNC(current_date, 'WW') - 21 AND TRUNC(current_date, 'WW') - 14),
 W2 AS (SELECT COUNT(*) AS WEEK_2
        FROM users WHERE created_at BETWEEN TRUNC(current_date, 'WW') - 14 AND TRUNC(current_date, 'WW') - 7),
 W3 AS (SELECT COUNT(*) AS WEEK_3
        FROM users WHERE created_at BETWEEN TRUNC(current_date, 'WW') - 7 AND TRUNC(current_date, 'WW')),
 W4 AS (SELECT COUNT(*) AS WEEK_4
        FROM users WHERE created_at > TRUNC(current_date, 'WW')),
 T AS (SELECT COUNT(*) AS TOTAL FROM users
      WHERE created_at > TRUNC(current_date, 'MM'))
 SELECT * FROM W1, W2, W3, W4, T";
```

Example of using ‘with’ clause

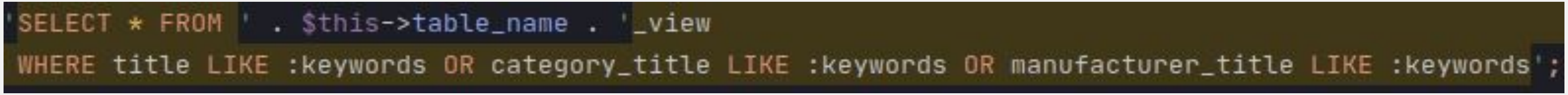


```
'WITH
 P AS (SELECT COUNT(*) AS PRODUCTS FROM products),
 C AS (SELECT COUNT(*) AS CATEGORIES FROM categories),
 M AS (SELECT COUNT(*) AS MANUFACTURERS FROM manufacturers),
 U AS (SELECT COUNT(*) AS USERS FROM users),
 A AS (SELECT COUNT(*) AS ACCOUNTS FROM admin_users)
 SELECT * FROM P, C, M, U, A';
```

ACTUAL IMPLEMENTATION

Example of using search

Here used products_view

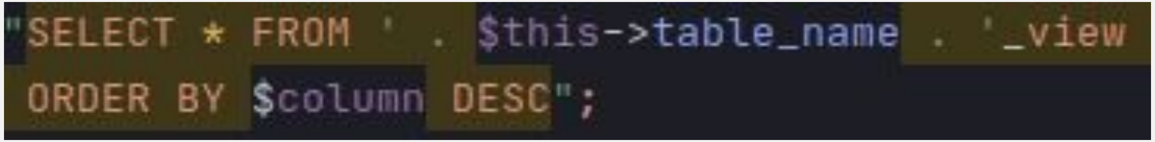


```
'SELECT * FROM ' . $this->table_name . '_view
 WHERE title LIKE :keywords OR category_title LIKE :keywords OR manufacturer_title LIKE :keywords';
```

Example of using filter

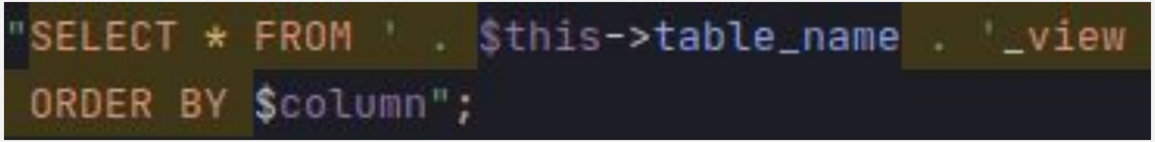
Here used products_view

DESC



```
"SELECT * FROM ' . $this->table_name . '_view
 ORDER BY $column DESC";
```

ASC



```
"SELECT * FROM ' . $this->table_name . '_view
 ORDER BY $column";
```

User types

CUSTOMERS (END USERS)

- **Table - users**
- **Role - guest_role**
- **Capabilities:**
  - Authorization (registration, login)
  - Read from all tables except - admin_users, visits, ips

admin_role capabilities:

- SELECT, UPDATE, INSERT, DELETE on all tables
- CREATE, DROP, ALTER users, tables, views, seqs, synonyms and etc.

ADMINISTRATORS (DBA USERS)

- **Table - admin_users**
- **Roles - admin_role, manager_role, guest_role**

manager_role capabilities:

- SELECT, UPDATE on all tables except admin_users

guest_role capabilities:

- SELECT on all tables except admin_users

Thank you

FOR YOUR ATTENTION AND FOR THIS COURSE