


МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
(ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ)
Факультет инноваций и высоких технологий
Кафедра «Алгоритмы и технологии программирования»

Семестровая работа по курсу “Базы данных”


Выполнил студент гр. Б05-822
Преподаватель

А.Г. Рухадзе
А.Д. Медведева

Цели работы

- 
- получение практических навыков работы с промышленными СУБД;
 - проектирование БД (концептуальное, логическое, физическое);
 - создание хранимых процедур, представлений, триггеров, индексов.

Основные понятия

- 
- **База данных** — совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.
 - **СУБД** — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Ход выполнения работы



Этапы выполнения задания:

- Выбор предметной области для моделирования;
- Проектирование базы данных;
- Подготовка DDL-скриптов и создание своей базы в СУБД;
- Наполнение созданной базы данными;
- Написание SELECT-запросов с использованием определенных функций;
- Подготовка представлений, хранимых процедур и триггеров;
- Защита семестрового проекта.

Выбор предметной области для моделирования



Выбранная тема

— база данных гитарного магазина.



Мотивация:

- ~~отсутствие фантазии;~~
- условная ознакомленность с темой;
- понимание, какую пользу может в реальной жизни принести база данных подобного рода.

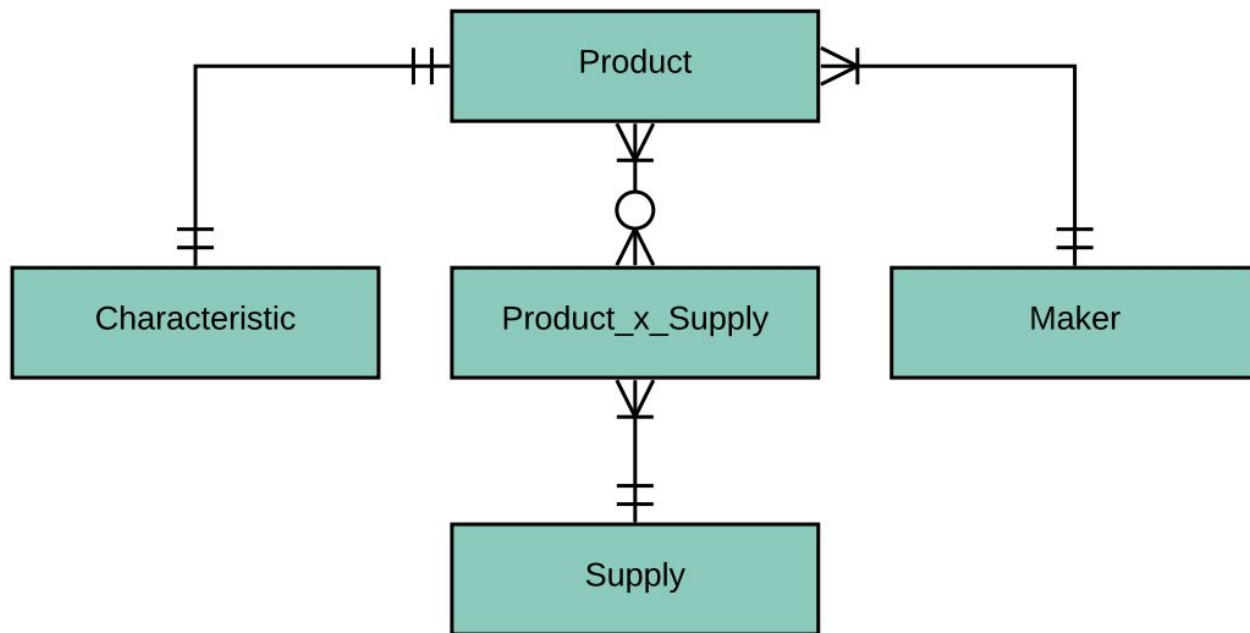


Польза подобной БД:

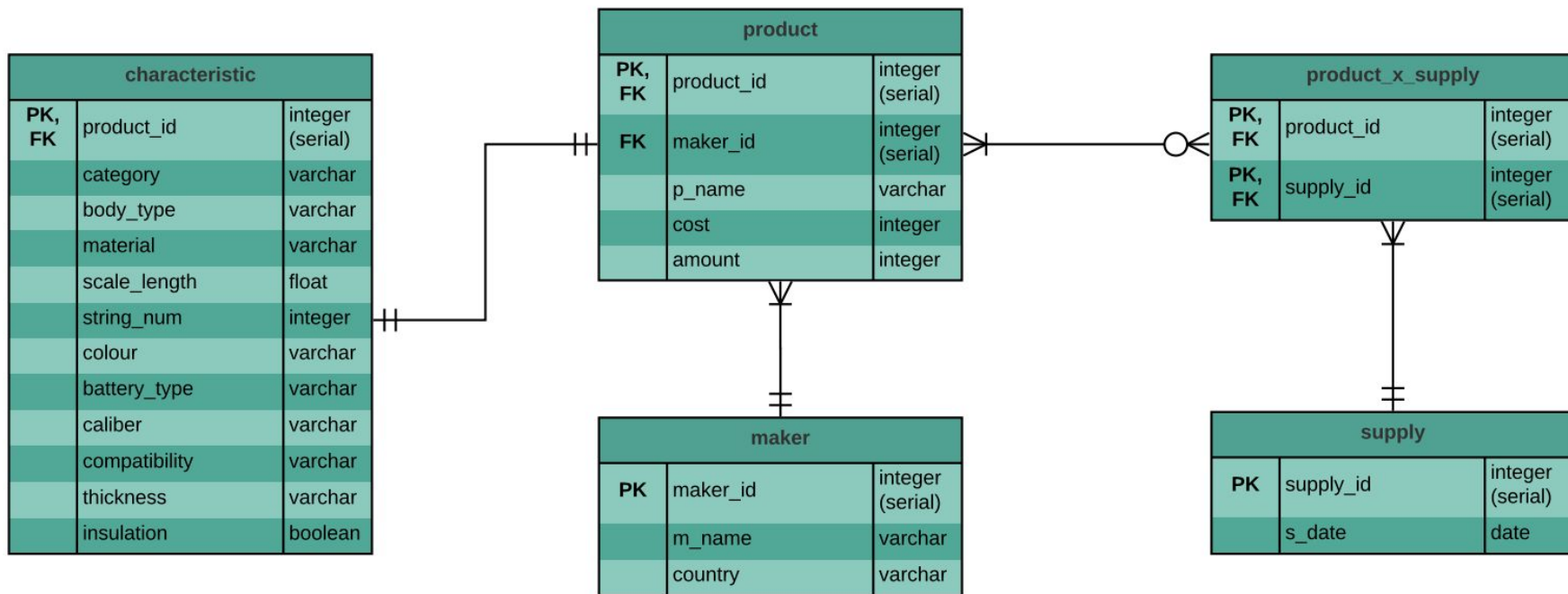
- автоматизация работы магазина;
- структуризация и систематизация данных;
- упрощение и ускорение работы с данными;
- большая надежность хранения данных.

Проектирование базы данных

Концептуальная модель



Логическая модель



Физическая модель



maker — информация о производителях

Название	Описание	Тип данных	Ограничение
maker_id	Идентификатор производителя	SERIAL	PRIMARY KEY
m_name	Имя производителя	VARCHAR(255)	NOT NULL UNIQUE
country	Страна	VARCHAR(255)	NOT NULL

product — информация о продуктах

Название	Описание	Тип данных	Ограничение
product_id	Идентификатор продукта	SERIAL	PRIMARY KEY FOREIGN KEY
maker_id	Идентификатор производителя	SERIAL	NOT NULL FOREIGN KEY
p_name	Наименование продукта	VARCHAR(255)	NOT NULL UNIQUE
cost	Цена продукта	INTEGER	CHECK (cost > 0)
amount	Количество продукта в наличии	INTEGER	CHECK (amount >= 0)

**Подготовка DDL-скриптов и
создание своей базы данных
в СУБД, наполнение
созданной базы данными**

—



Операторы Data Definition Language (DDL):

- CREATE;
- ALTER;
- TRUNCATE;
- DROP.



Пример запроса:

В результате данных запросов будет создана схема базы данных гитарного магазина, создана и заполнена таблица, содержащая информацию о поставках.

```
-- Создание схемы базы данных гитарного  
магазина
```

```
CREATE SCHEMA guitar_shop_db;
```

```
-- Информация о поставках
```

```
CREATE TABLE guitar_shop_db.supply(  
    supply_id SERIAL PRIMARY KEY,  
    s_date DATE NOT NULL  
);
```

```
-- Заполнение таблицы "supply"
```

```
INSERT INTO guitar_shop_db.supply(s_date)  
VALUES ('2019-05-31'), ('2019-06-28'),  
('2019-07-12'), ('2019-08-02'),  
('2019-08-30');
```

Написание SELECT-запросов с использованием определенных функций



Осмысленные SELECT-запросы с использованием:

- a. GROUP BY + HAVING
- b. ORDER BY
- c. func() OVER:
 - i. PARTITION BY
 - ii. ORDER BY
 - iii. PARTITION BY + ORDER BY



Пример запроса:

В результате данного запроса будет выведен список гитар, представленных в магазине, расположенных в порядке возрастания цены.

Будут выведены: название товара, категория, стоимость товара.

```
-- Запрос с использованием ORDER BY:
SELECT product.p_name,
characteristic.category, product.cost

FROM guitar_shop_db.product INNER JOIN
guitar_shop_db.characteristic
ON product.product_id =
characteristic.product_id

WHERE
guitar_shop_db.characteristic.category
LIKE 'ukulele' OR
guitar_shop_db.characteristic.category
LIKE 'classical guitar' OR
guitar_shop_db.characteristic.category
LIKE 'acoustic guitar' OR
guitar_shop_db.characteristic.category
LIKE 'electroacoustic guitar'

ORDER BY cost ASC;
```

Подготовка представлений, хранимых процедур и триггеров



Цель создания представлений (view)

— получение осмысленной сводной таблицы.



Пример запроса:

В результате данного запроса будет создано представление, в котором будут указаны производители и их товары, которые продаются в магазине.

Также выводится стоимость каждого товара и средняя стоимость товаров этого производителя.

(Производители расположены в алфавитном порядке, товары в порядке возрастания цены в пределах каждого окна)

```
CREATE VIEW guitar_shop_db.makers_view AS

SELECT maker.m_name, product.p_name,
product.cost, avg(cost) OVER (PARTITION
BY maker.m_name) AS "average_price"

FROM guitar_shop_db.product INNER JOIN
guitar_shop_db.maker

ON product.maker_id = maker.maker_id;
```




Цели создания хранимых процедур:

- упрощение многократного использования необходимой функции;
- дальнейшее использование в написании триггеров.



Пример запроса:

Функция, возвращающая стоимость конкретного продукта, переведенную в рубли по курсу на 24.04.19.

В результате указанного вызова данной функции будет выведена стоимость Yamaha C40 в рублях по курсу на 24.04.19.

```
CREATE OR REPLACE FUNCTION  
guitar_shop_db.func_usd_to_rub(prod  
VARCHAR) RETURNS NUMERIC AS
```

```
$$
```

```
SELECT (cost * 64.31)  
FROM guitar_shop_db.product  
WHERE p_name LIKE prod  
$$ LANGUAGE SQL;
```

```
SELECT guitar_shop_db.func_usd_to_rub  
( 'Yamaha C40' );
```



Пример запроса:

Функция, возвращающая суммарную стоимость всех товаров в магазине.
(для триггера 1)

```
CREATE OR REPLACE FUNCTION
guitar_shop_db.prod_sum() RETURNS BIGINT
AS
$$
SELECT sum(cost * amount)
FROM guitar_shop_db.product
$$ LANGUAGE SQL;

SELECT guitar_shop_db.prod_sum();
```



Цели создания триггеров:

- обеспечение целостности данных и реализации сложной бизнес-логики;
- проведение вспомогательных расчетов.



Пример запроса:

При вызове INSERT/DELETE/UPDATE для таблицы product в таблицу sum_cost добавляется суммарная стоимость всей продукции в магазине на текущий момент времени.

(Срабатывает после выполнения транзакции, часовой пояс - GMT).

```
CREATE OR REPLACE FUNCTION
guitar_shop_db.trigger_sum() RETURNS
TRIGGER AS $$
BEGIN
    IF (TG_OP = 'INSERT') OR (TG_OP =
'DELETE') OR (TG_OP = 'UPDATE') THEN
        INSERT INTO guitar_shop_db.sum_cost
VALUES
            ((SELECT
guitar_shop_db.prod_sum()), current_date,
current_time);
        RETURN NEW;
    END IF;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trigger_prod_sum
AFTER INSERT OR UPDATE OR DELETE ON
guitar_shop_db.product
FOR EACH ROW
EXECUTE PROCEDURE
guitar_shop_db.trigger_sum();
```

Заключение и выводы



Зачем все это нужно?

- Базы данных:
 - удобное хранение различных данных для последующей работы с ними.
- СУБД:
 - хранение и систематизация огромного количества информации;
 - быстрая обработка клиентских запросов и выдача свежей и актуальной информации.