

# Частное учреждение профессионального образования «Высшая школа предпринимательства»

---

«Курсовой проект»

«Разработка базы данных для магазина  
компьютерных комплектующих»

Колосов К.А





## Цель проекта

- Создание базы данных для магазина компьютерных комплектующих, которая позволит оптимизировать управление складом, контроль за товарным ассортиментом и повысить качество обслуживания клиентов.



# Задачи проекта

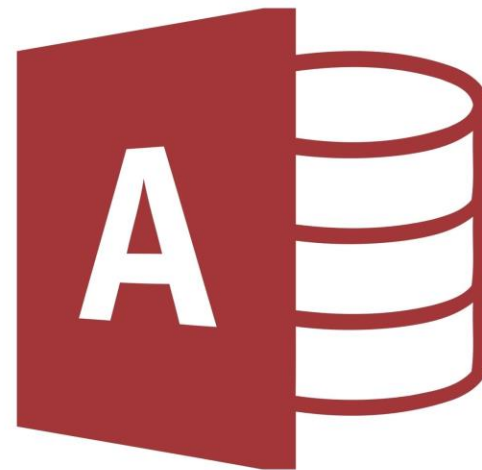
- Провести анализ предметной области.
- Определить требования к разрабатываемой базе данных.
- Спроектировать схему базы данных.
- Реализовать базу данных.
- Создать функциональные объекты базы данных.
- Назначить права пользователям базы данных.
- Проверить работоспособность и корректность функционирования базы данных.

# Анализ предметной области

- **Основные типы баз данных:**
- **Иерархические:** Древовидная структура данных.
- **Объектные:** Данные моделируются в виде объектов.
- **Реляционные:** Данные в виде двумерных таблиц.
- **Объектно-реляционные:** Расширение реляционной модели с объектно-ориентированными концепциями.
- **Сетевые:** Данные в виде сетевых структур.

# Популярные СУБД:

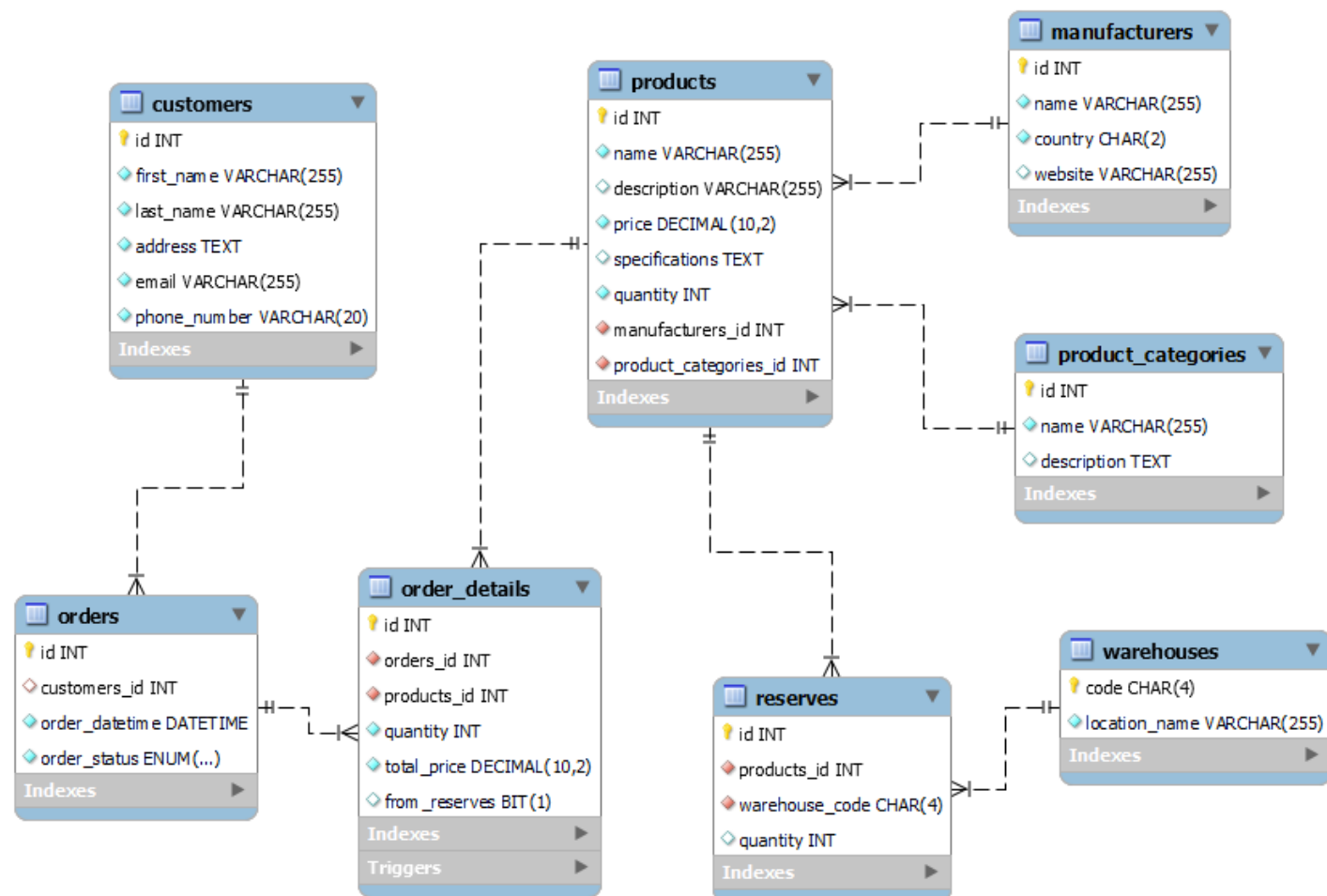
---



# Требования к разрабатываемой базе данных

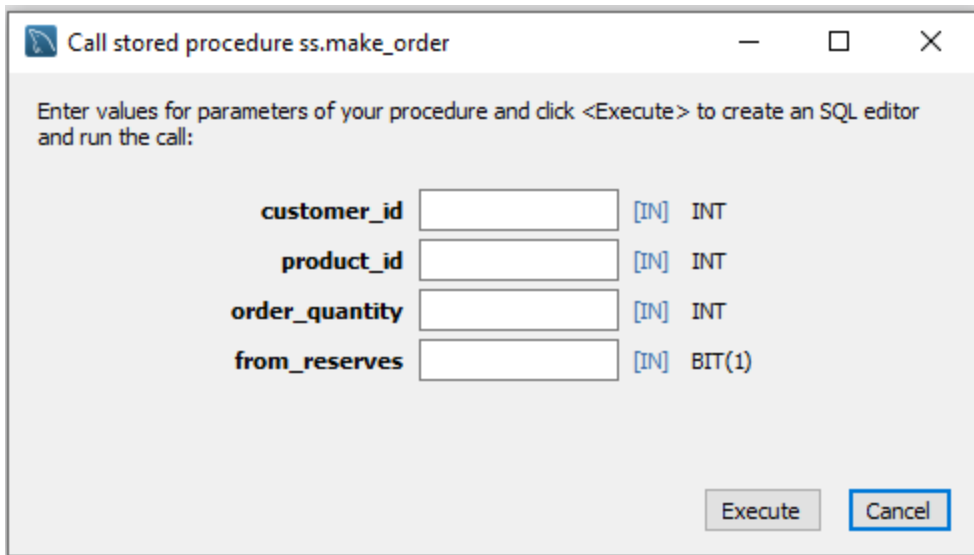
- **Основные требования:**
- **Надежность данных:** Обеспечение целостности и непротиворечивости данных.
- **Производительность:** Высокая скорость обработки запросов.
- **Безопасность:** Защита данных от несанкционированного доступа.
- **Масштабируемость:** Возможность увеличения объема данных и количества пользователей.
- **Удобство использования:** Простота в эксплуатации и администрировании.

# Схема базы данных



# Создание функциональных объектов

- **Создание функциональных объектов:**
- Хранимые процедуры (для поиска товара, обновления цены, обработки заказа)
- Пользовательские функции
- Назначение прав пользователям

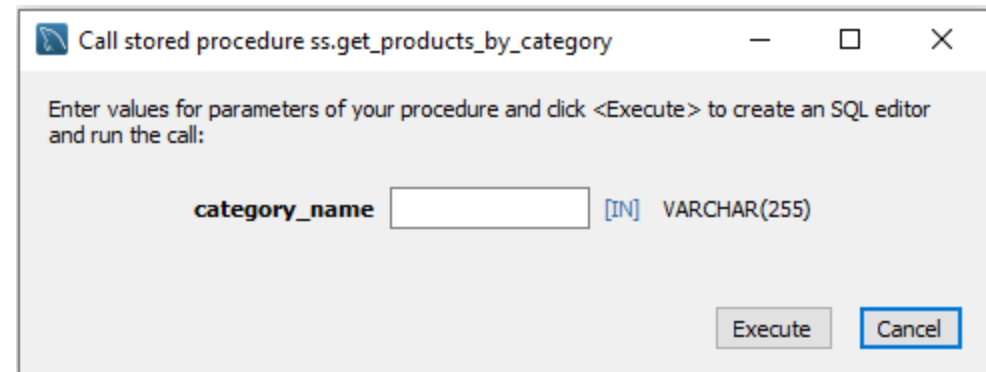


Call stored procedure ss.make\_order

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

customer_id	<input type="text"/>	[IN]	INT
product_id	<input type="text"/>	[IN]	INT
order_quantity	<input type="text"/>	[IN]	INT
from_reserves	<input type="text"/>	[IN]	BIT(1)

Execute Cancel

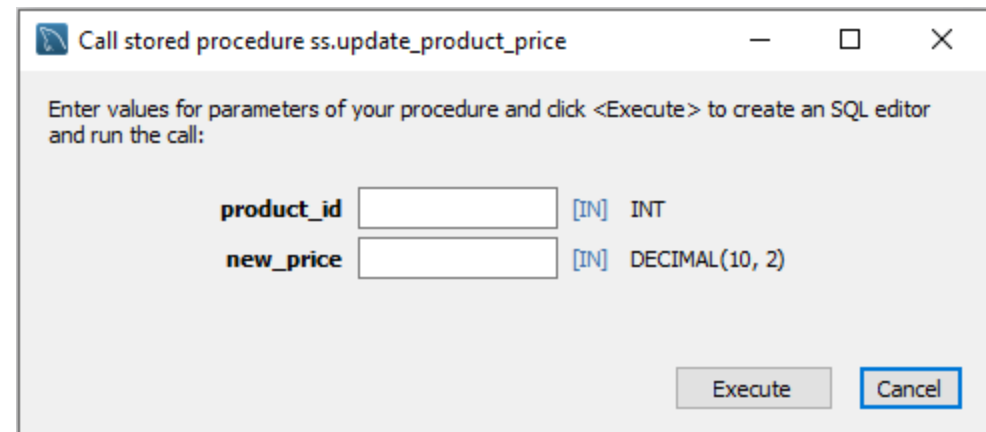


Call stored procedure ss.get\_products\_by\_category

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

category_name	<input type="text"/>	[IN]	VARCHAR(255)
---------------	----------------------	------	--------------

Execute Cancel




Call stored procedure ss.update\_product\_price

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

product_id	<input type="text"/>	[IN]	INT
new_price	<input type="text"/>	[IN]	DECIMAL(10, 2)

Execute Cancel





# Пользовательская функция

---

```
DELIMITER $$

CREATE FUNCTION calculate_order_total(product_id INT)
RETURNS DECIMAL(10,2)
DETERMINISTIC
BEGIN
    DECLARE product_quantity INT;
    DECLARE reserve_quantity INT;
    DECLARE product_price DECIMAL(10,2);
    DECLARE total_price DECIMAL(10,2);

    -- Получаем количество и цену продукта из таблицы products
    SELECT quantity, price INTO product_quantity, product_price FROM products WHERE id = product_id;

    -- Получаем количество товара из таблицы reserves
    SELECT SUM(quantity) INTO reserve_quantity FROM reserves WHERE product_id = product_id;

    -- Если товар не найден в таблице reserves, устанавливаем его количество как 0
    IF reserve_quantity IS NULL THEN
        SET reserve_quantity = 0;
    END IF;

    -- Вычисляем общее количество товаров и общую стоимость
    SET total_price = (product_quantity + reserve_quantity) * product_price;

    RETURN total_price;
END $$

DELIMITER ;
```

# Создание пользователей и назначение привилегий

## Создание менеджера

```
-- создание роли manager_role
CREATE ROLE IF NOT EXISTS 'manager_role';

-- назначение привилегий роли manager_role
GRANT SELECT, INSERT, UPDATE, DELETE ON shop.products TO 'manager_role';
GRANT SELECT, INSERT, UPDATE, DELETE ON shop.customers TO 'manager_role';
GRANT SELECT, INSERT, UPDATE, DELETE ON shop.orders TO 'manager_role';
GRANT SELECT, INSERT, UPDATE, DELETE ON shop.order_details TO 'manager_role';
GRANT SELECT ON shop.manufacturers TO 'manager_role';
GRANT SELECT ON shop.product_categories TO 'manager_role';
GRANT SELECT ON shop.reserves TO 'manager_role';
GRANT SELECT ON shop.warehouses TO 'manager_role';

-- назначение привилегий на выполнение хранимых процедур
GRANT EXECUTE ON PROCEDURE shop.get_products_by_manufacturer TO 'manager_role';
GRANT EXECUTE ON PROCEDURE shop.get_products_by_category TO 'manager_role';
GRANT EXECUTE ON PROCEDURE shop.update_product_price TO 'manager_role';
GRANT EXECUTE ON PROCEDURE shop.calculate_order_total TO 'manager_role';
GRANT EXECUTE ON PROCEDURE shop.make_order TO 'manager_role';

-- создание пользователя manager и назначение ему роли manager_role
CREATE USER IF NOT EXISTS 'manager'@'localhost' IDENTIFIED BY 'Pa$sw0rd';
GRANT 'manager_role' TO 'manager'@'localhost';

-- установка роли manager_role по умолчанию для пользователя manager
SET DEFAULT ROLE 'manager_role' FOR 'manager'@'localhost';

-- применение изменений
FLUSH PRIVILEGES;
```

## Создание клиента

```
-- создание роли customer_role
CREATE ROLE IF NOT EXISTS 'customer_role';

-- назначение привилегий роли customer_role
GRANT SELECT ON shop.products TO 'customer_role';
GRANT SELECT ON shop.product_categories TO 'customer_role';
GRANT SELECT ON shop.manufacturers TO 'customer_role';

-- назначение привилегий на выполнение хранимых процедур
GRANT EXECUTE ON PROCEDURE shop.make_order TO customer_role;

-- создание пользователя customer и назначение ему роли customer_role
CREATE USER IF NOT EXISTS 'customer'@'localhost' IDENTIFIED BY 'SecurePassword';
GRANT 'customer_role' TO 'customer'@'localhost';

-- установка роли customer_role по умолчанию для пользователя customer
SET DEFAULT ROLE 'customer_role' FOR 'customer'@'localhost';

-- применение изменений
FLUSH PRIVILEGES;
```

## Создание администратора

```
-- создание роли admin_role
CREATE ROLE IF NOT EXISTS admin_role;

-- Предоставление полных прав администратору на схему housing
GRANT ALL PRIVILEGES ON shop.* TO admin_role;

-- создание пользователя и назначение ему роли admin_role
CREATE USER IF NOT EXISTS 'admin'@'localhost' IDENTIFIED BY 'Pa$sw0rd';
GRANT admin_role TO 'admin'@'localhost';

-- установка роли admin_role по умолчанию для пользователя admin
SET DEFAULT ROLE admin_role TO 'admin'@'localhost';

FLUSH PRIVILEGES;
```



# Заключение

- Разработка базы данных для магазина компьютерных комплектующих успешно реализована. Система обеспечивает надежное и эффективное управление данными, что способствует улучшению бизнес-процессов и повышению конкурентоспособности магазина.

СПАСИБО ЗА ВНИМАНИЕ