**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА**

**И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**Смоленский филиал РАНХиГС**

Специальность:

09.02.07 Информационные системы и программирование

**ПРОЕКТНАЯ РАБОТА**

по дисциплине **«**Управление и автоматизация баз данных»

на тему: «Проектирование элементов системы автоматизации продаж музыкального оборудования»

**Автор:**

обучающийся

группы 37/11К-ИТО

очной формы обучения

Махницкий Д.С.

**Руководитель:**

преподаватель

Харламова О.Е.

Смоленск 2024

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc181971278)

[1 Информационное, алгоритмическое и программное обеспечение задачи автоматизации 4](#_Toc181971279)

[2 Тестирование элементов системы и формирование технологической среды 5](#_Toc181971280)

[3 Анализ проблем автоматизации процесса продаж музыкального оборудования и пути их решения 6](#_Toc181971281)

[ЗАКЛЮЧЕНИЕ 7](#_Toc181971282)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 8](#_Toc181971283)

[ПРИЛОЖЕНИЕ А 9](#_Toc181971284)

[ПРИЛОЖЕНИЕ Б 10](#_Toc181971285)

[ПРИЛОЖЕНИЕ В 11](#_Toc181971286)

# ВВЕДЕНИЕ

# 1 Информационное, алгоритмическое и программное обеспечение задачи автоматизации

Согласно предметной области (продажа музыкального оборудования), была разработана база данных (БД) с помощью системы управления базами данных СУБД MySQL. На рисунках 1.1 и 1.2 изображены логическая и физическая модели базы данных соответственно.

Изображение выглядит как текст, снимок экрана, диаграмма, число

Автоматически созданное описание

Рисунок 1.1 – Логическая модель базы данных в нотации Мартина

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

На данной модели (рисунок 1.1) можно заметить определенную симметрию – каждый вид музыкального товара, а именно аксессуары (accessories), аудио-оборудование (audio\_equipment\_units), нотные издания (sheet\_music\_editions) и музыкальные инструменты (musical\_instruments) имеет собственную таблицу для типов с шаблонным названием имя\_вида\_товара\_specific\_types, а так же свою связующую таблицу для отношения (тоже с шаблонным названием имя\_вида\_товара\_sale) многие-ко-многим между собой и таблицей «продажи» (sales).

Опишем преимущество использования отдельной таблицы со специфичными типами (подтипами) для каждого вида товара. Каждый вид товара имеет уникальные подтипы, которые, в данном случае, могут пересекаться с названиями других подтипов (например, данная структура позволят добавить в БД гитару (музыкальный инструмент) с подтипом «акустика», а также колонки (аудио-оборудование) с таким же подтипом, поддерживая ограничение уникальности названия для каждого вида товара. Из альтернатив данной реализации можно рассмотреть создание единой таблицы со всеми подтипами. В таком случае необходимо было бы создать вторую таблицу со всеми видами товара, и в каждой строке подтипа хранить информацию о виде товара (дискриминатор). Например, строка, которая в текущей реализации выглядит как отображено в таблице 1.1, в альтернативном варианте выглядела бы как в таблице 1.2.

Таблица 1.1 – Пример строки из таблицы musical\_instrument\_specific\_type

|  |  |
| --- | --- |
| specific\_type\_id | name |
| 1 | Акустика |

Таблица 1.2 – Пример строки из альтернативного варианта таблицы подтипов specific\_types

|  |  |  |
| --- | --- | --- |
| specific\_type\_id | name | kind\_of\_goods |
| 1 | Акустика | Музыкальные инструменты |

Таким образом, каждая строка должна была бы хранить лишнюю (с точки зрения текущей реализации) информацию (пренебрегая нормализацией). В текущей реализации вид товара «автоматически» хранится в таблице в соответствии с ее названием. Второй плюс данной реализации – более быстрый запрос подтипов товара определенного вида – достаточно выполнения следующей команды:

SELECT \*

FROM `musical\_instrument\_specific\_types`;

вместо:

SELECT \* `specific\_types`

WHERE `kind\_of\_goods` = ‘Музыкальные инструменты’;

Другими словами, в текущей реализации подтипы уже отфильтрованы, что увеличивает скорость выполнения запроса на получение всех подтипов определенного вида товара.

Для хранения информации о товаре разных типов используется подход Table Per Class (TPC) – для каждого класса (сущности) создается отдельная таблица со всеми свойствами (атрибутами). Объясняется это тем, что в данном случае ожидается, что поиск товара (относительно сложный по своей структуре запрос) будет осуществляться довольно часто, причем всегда по одному виду товара (с точки зрения предметной области нет смысла одновременно пытаться искать музыкальный инструмент и нотное издание (например, по фразе «синтезатор») – при поиске конечный пользователь всегда знает какой вид товара его интересует) и в результате, при поиске товара запрос так же, как в предыдущем случае с подтипами, не будет содержать дополнительной фильтрации по виду товара.

Оба сделанных выбора вариантов хранения информации в БД основаны на эмпирическом методе (так как данная работа – учебная), в реальной ИС необходимо выполнить тесты для определения лучшего подхода.

Важная особенность данной БД заключается в том, что таблица sales (продажи) имеет отношение с таблицами товаров разного вида «многие-ко-многим», а таблица goods\_delivery (доставка товара) – «один-ко-многим». Рассмотрим два похожих по смыслу случая:

- Клиент магазина купил музыкальный инструмент, но позже вернул его. Затем этот инструмент купил другой клиент.

- Поставщик доставил нотное издание, которое через некоторое время купил определенный клиент, а затем это нотное издание «чудом» (крайне маловероятный случай) оказалось на складе у поставщика, который затем снова выполнит поставку товара с этим нотным изданием.

В результате определенный товар может побывать сразу в двух и более разных покупках и на практике это происходит регулярно. Возврат товара клиентом – обыденное дело. Также книга действительно может побывать в двух и более поставках товара, но вероятность данного случая настолько мала, что ей можно пренебречь. Соответственно, в первом случае реализовано отношение «многие-ко-многим», а во втором – «один-ко-многим».

Рисунок 1.2 – Физическая модель базы данных в нотации Мартина

1. Описание базы данных (структуры таблиц и др.) или структуры объектов
2. Концептуальная и логическая модели данных
3. Физическая модель данных или объектная модель данных
4. Описание общего алгоритма работы информационной системы
5. Подробно описать один наиболее интересный алгоритм информационной системы (сложный). Алгоритмы проиллюстрировать блок-схемами. Алгоритмы обработки информации, описание связей модулей и БД и т.д.
6. В зависимости от технологии реализации представить дерево функций и сценарий диалога, диаграмму классов, схему взаимосвязи программных модулей и информационных файлов и/или другие модели, служащие для описания программного обеспечения. Описать данные модели.

# 2 Тестирование элементов системы и формирование технологической среды

1. Обоснование выбора технологии тестирования, наиболее соответствующей особенностям разработанной системы, и ее характеристика.
2. Разработать тесты для спроектированного в ПО (объяснить выбор вида тестирования, почему применяются именно эти, а не другие тестовые комбинации пути прохождения вычислительного процесса).
3. Привести результаты тестирования в виде таблиц и экранных форм. Описание тестов, результатов тестирования и их анализ. В тестировании предусмотреть проверку выполнения требований по надежности и защите информации. Результаты тестирования рекомендуется оформлять в виде таблицы, в которой указываются тестовые воздействия и соответствующие реакции ИС. При целесообразности можно добавить экранные формы для иллюстрации результатов некоторых тестов.

# 3 Анализ проблем автоматизации процесса продаж музыкального оборудования и пути их решения

# ЗАКЛЮЧЕНИЕ

Таким образом, все поставленные в работе задачи выполнены, и, соответственно, цель достигнута.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ



# **ПРИЛОЖЕНИЕ А**

Экранные формы системы автоматизации

# **ПРИЛОЖЕНИЕ Б**

Программный код элементов системы автоматизации

# ПРИЛОЖЕНИЕ В

Тест-кейсы для тестирования программного обеспечения