

Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
«Костромской государственный университет»  
(ФГБОУ ВПО «КГУ»)  
Институт Автоматизированных систем и технологий  
Кафедра информационных систем и технологий  
Направление подготовки: «Информационные системы и технологии»  
Дисциплина: Базы данных

### **КУРСОВАЯ РАБОТА**

«Проектирование и реализация базы данных для формирования фискального чека в магазине»

Выполнил студент:

Глазов Даниил Игоревич

Группа: 20-ИМбо-3

Проверил: кандидат наук, доцент

Прядкина Нина Олеговна

Оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_

Кострома,

2022 г.

## СОДЕРЖАНИЕ

Введение	3
1. Проектирование базы данных	4
1.1. Описание предметной области	4
1.2. Анализ предметной области	5
1.3. Диаграмма «сущность—связь»	9
1.4. Реляционная схема базы данных	10
2. Реализация базы данных	12
2.1. Реализация БД на сервере	12
2.2. Реализация объектов базы данных	27
2.3. Реализация системы безопасности	51
3. Реализация приложения для базы данных	54
Заключение	56
Список литературы	57

# Введение

С древних времен у людей была необходимость в передаче и хранении разного рода информации. Развитие способов хранения информации шло в ногу с технологическим прогрессом человечества. Совсем недавно во всем мире информация, например, о купле-продаже материальных ценностей, хранилась на бумажных носителях. На сегодняшний день технологии ушли далеко вперед, теперь с помощью различного рода ЭВМ, человечество получило возможность не только удобно и компактно хранить информацию в специальных базах данных, но и быстро получать нужную часть хранящейся информации по каким-либо критериям отбора.

В работе рассматриваются этапы проектирования и реализации базы данных на сервере, а также создание приложения для работы с базой данных.

Цель – спроектировать и реализовать базу данных для формирования фискального чека в магазине.

Предмет – особенности проектирования и реализации базы данных для формирования фискального чека в магазине.

Задачи:

- Спроектировать базу данных выбранной предметной области.
- Реализовать на сервере базу данных выбранной предметной области.

# 1. Проектирование базы данных

## 1.1. Описание предметной области

Цель данного этапа проектирования - составить описание предметной области.

Задачи - проанализировать нормативные документы и акты, на основе которых можно составить описание предметной области.

Система для выписки чеков, которые продавец формирует на онлайн-кассе для передачи покупателю (клиенту) в результате произошедшей сделки купли/продажи товаров. Чеки включают в себя: название организации, ее местонахождение (город, адрес), вид чека, номер документа, ИНН (идентификационный номер налогоплательщика), кассир, который выдал чек, наименование и количество товаров, коды этих товаров, их стоимость, если один и тот же товар взят несколько раз, то показывается количество этого товара, цена за штуку и суммарная цена за эти товары, промежуточная сумма без скидки клиента (подитог), код клиента, его скидка в процентах и сумма скидки, итоговая скидка, итоговая сумма с учетом скидки клиента, статус оплаты, вид оплаты и сумма, внесенная клиентом, сдача клиенту, количество товарных строк, дата и время покупки. Оплату можно производить при помощи кредитной карты и наличными. После завершения сделки продажи/покупки товара информация о ней записывается в БД и покупателю выписывается чек. Покупатель имеет доступ к базе данных с возможностью просмотра только своих чеков, после покупки у него остается товарный чек с информацией о сделке. Кассир имеет доступ к БД в рамках, совершенных им продаж. При возврате товара покупателем проверяется его товарный чек на соответствие с информацией в базе данных по номеру чека. В базе данных должен храниться перечень товаров с их характеристиками: название товара, цена товара, количество товара в наличии, так же должна храниться информация о кассире, об организации, в которой кассир работает и о покупателе. Предусматриваются

следующие ограничения в системе: чек может быть сформирован только одним кассиром, от имени одной компании, в которой этот кассир работает, скидка может быть от 0 до 50 процентов от стоимости покупки, у каждого покупателя индивидуальная скидка на все товары (в процентах), чек составляется для сделки, в которой есть хотя бы один товар (количество товарных строк минимум 1), чек печатается после оплаты покупателем товаров, заявленных в чеке, чек формируется для каждого покупателя и покупки отдельно, в чек заносится информация о действительной дате и времени сделки на момент ее совершения.

Таким образом, на данном этапе была рассмотрена и описана предметная область, все цели и задачи данного этапа были достигнуты.

## 1.2. Анализ предметной области

Цель данного этапа проектирования - произвести анализ предметной области.

Задачи - определить цель создания базы данных, измеримость цели, определить категории данных, категории пользователей, бизнес-правила и ограничения.

На основе описания предметной области был проведен анализ предметной области. Была выделена цель: облегчить и сделать более эффективными процесс формирования фискальных чеков в результате сделки купли/продажи, и измеримость цели: уменьшение времени на формирование чека. Были приведены категории данных (рис. 1.1, 1.2). Далее были приведены категории пользователей (рис. 1.3) и бизнес-правила и ограничения (рис. 1.4, 1.5, 1.6).

Категория	Сведения о категории	Тип данных	Объём данных	Темпы роста	Комментарии
Кассир	1. Код кассира	Целое число	12 человек	1 чел. в год	Уникальный код кассира
	2. Фамилия	Строка			
	3. Имя	Строка			
	4. Отчество	Строка			
	5. Организация	Значение категории "Организация"			организация, в которой работает кассир и от лица которой выписывается чек
Организация	1. Номер организации	Целое число	5 фирм	1 фирма в год	Уникальный код организации
	2. Название организации	Строка			
	3. Адрес	Строка			Город, улица, дом
	4. ИНН	Целое число			идентификационный номер налогоплательщика
	5. Регистрационный номер	Целое число			Регистрационный номер организации
	6. График работы	Строка			
Покупатель	1. Код клиента	Целое число	Несколько тысяч человек	10% в год	Уникальный код его дисконтной карты
	2. Фамилия	Строка			
	3. Имя	Строка			
	4. Отчество	Строка			
	5. Скидка	Вещественное число			Рассчитывается в зависимости от суммы покупки
Товар	1. Код товара	Целое число	Несколько сотен единиц	10% в год	Уникальный код товара
	2. Название товара	Строка			
	3. Цена товара	Вещественное число			Вещественное число для случаев если цена указана "с копейками": 899.9
	4. Количество товара в наличии	Целое число			
	5. Категория	Строка			Например, ноутбуки, телевизоры, крупы, бытовая химия и так далее
	6. Единицы измерения	Строка			Единицы исчисления товара: килограммов, литров, штук.

Рис. 1.1. Категории данных категории кассир, организация, покупатель и товар

Чек	1. Фискальный номер чека	Целое число	от 2600 до 5200 чеков в квартал	10% в год	Уникальный номер чека
	2. Кассир	Значение категории "Кассир"			
	3. товары	Множество значений категории "товар"			
	4. Количество товара	Целое число			Для каждой группы одинаковых товаров отдельное количество
	5. Цена за товар	Вещественное число			Суммарная цена за каждый вид товара в соответствии с количеством этого товара в чеке
	5. Подитог	Вещественное число			Вычисляется из суммы цен за товар
	6. Итого	Вещественное число			Подитог с учетом скидки клиента
	7. Покупатель	Значения категории "Покупатель"			
	8. Вид оплаты	Строка			Наличные / Карта
	9. Статус оплаты	Строка			Отслеживание статуса оплаты сделки
	10. Количество товарных строк	Целое число			Количество видов товаров в чеке
	11. Дата и время	Дата			Дата и время совершения сделки
	12. Кассовый аппарат	Строка			Кассовый аппарат, на котором выписан чек. Наименование кассового аппарата: фирма, модель
	13. Внесено	Вещественное число			
	14. Сдача	Вещественное			
ЭКЛЗ лента	15. ЭКЛЗ лента	Множество значений			
	1. Регистрационный номер ЭКЛЗ ленты	Целое число			Регистрационный номер электронно-контрольной ленты
	2. Номер КПК	Целое число			Криптографический Проверочный Код используется для проверки правильности установки ЭКЛЗ на кассовом аппарате
	3. Значение КПК	Строка			

Рис. 1.2. Категории данных категории чек

Категория пользователей	Количество	Темпы роста	Права	Задачи (д-е)
Кассир			1. Доступ к информации, необходимой для формирования чека, в рамках своей организации	1. Формирование и выдача покупателю чека
			2. Частичный доступ к чекам	2. Проверка подлинности чека при возврате товара покупателем
Покупатель			1. Частичный доступ к чекам	1. Просмотр своих чеков

Рис. 1.3. Категории пользователей

Категория	Сведения о категории	Условие на значение	Уникальность	Обязательность	Связь	Бизнес-правило
Кассир	1. Код кассира		Уникальный	Обязательно		
	2. Фамилия					
	3. Имя					
	4. Отчество					
	5. Организация			Обязательно	Кассир работает только в одной организации	
Организация	1. Номер организации		Уникальный	Обязательно		
	2. Название организации					
	3. Адрес					
	4. ИНН	12 символов - цифры	Уникальный			
	5. Регистрационный номер	12 символов - цифры	Уникальный			
	6. График работы					

Рис. 1.4. Бизнес-правила и ограничения

Покупатель	1. Код клиента	6 символов - цифры	Уникальный	Обязательно		
	2. Фамилия					
	3. Имя					
	4. Отчество					
	5. Скидка	В соответствии с бизнес-правилом вычисляется скидка. Значение по умолчанию Null		Обязательно		При средней цене чека покупателя в 500 скидка 2%, при средней цене чека в 1000 скидка 5%, при средней цене чека в 5000 скидка 10%, при средней цене чека в 25000 скидка 15%, от 50000 скидка 20%. Скидки между собой не складываются. Скидка может быть использована только один раз за покупку и при чем только одна. Всегда применяется максимально возможная скидка, в соответствии с ценой покупки. Для скидки покупателю нужно предъявить дисконтную карту.
Товар	1. Код товара	13 символов - цифры	Уникальный	Обязательно		
	2. Название товара					
	3. Цена товара	>0				
	4. Количество товара в н	>=0				
	5. Категория товара	Ноутбуки, телевизоры, крупы,				
	6. Единицы товара	кг, л, шт				

Рис. 1.5. Бизнес-правила и ограничения



Чек	1. Фискальный номер чека	от 5 до 9 символов - цифры	Уникальный	Обязательно		При создании чека автоматически присваивается значение.
	2. Кассир				В чеке фигурирует только один кассир	
	3. Товар				В чеке может быть разное количество товаров, но	товар появляется в чеке, если количество, покупаемого покупателем товара $\geq 1$ , а так же если товар есть в наличии в нужном количестве
	4. Количество товара	$\geq 1$		Обязательно		
	5. Цена за товар	$\geq 1$				
	6. Подитог					Сумма цен товаров с учетом количества каждого товара, без учета скидки покупателя
	7. Итого					Сумма цен товаров с учетом количества каждого товара, с учетом скидки покупателя
	8. Покупатель				В чеке фигурирует только один покупатель	
	9. Вид оплаты	Наличные / Карта				При выборе покупателем способа оплаты наличными или банковской картой, выбранный способ указывается в чеке. Нельзя использовать оба способа одновременно.
	10. Статус оплаты	Оплачен / Не оплачен				Если покупатель произвел оплату, то статус равен "оплачен", если нет - "не оплачен" (оплатой считается внесенная сумма денег для оплаты товара, в расчете из которой рассчитывается сдача)
	11. Количество товарных строк	$\geq 1$				Есть количество разных товаров в чеке
	12. Дата и время создания чека					Присваивается значение, после оплаты чека
	12. Кассовый аппарат				Чек выписывается на одном Кассовом аппарате	
	13. Внесено					Заполняется в момент оплаты
	14. Сдача					Вычисляется из разности внесенной суммы и итога
ЭКЛЗ лента	15. Множество значений ЭКЛЗ ленты			Обязательно		
	1. Регистрационный номер ЭКЛЗ ленты	10 символов - цифры	Уникальный	Обязательно		
	2. Номер КПК	8 символов - цифры				
	3. Значение КПК	7 символов - первый символ # далее 6 цифр				

Рис. 1.6. Бизнес-правила и ограничения

Таким образом, на данном этапе был проведен анализ предметной области, все цели и задачи данного этапа были достигнуты.

### 1.3. Диаграмма «сущность–связь»

Цель данного этапа проектирования - создать ER диаграмму.

На основе проведенного анализа предметной области была построена ER диаграмма (рис. 1.7).

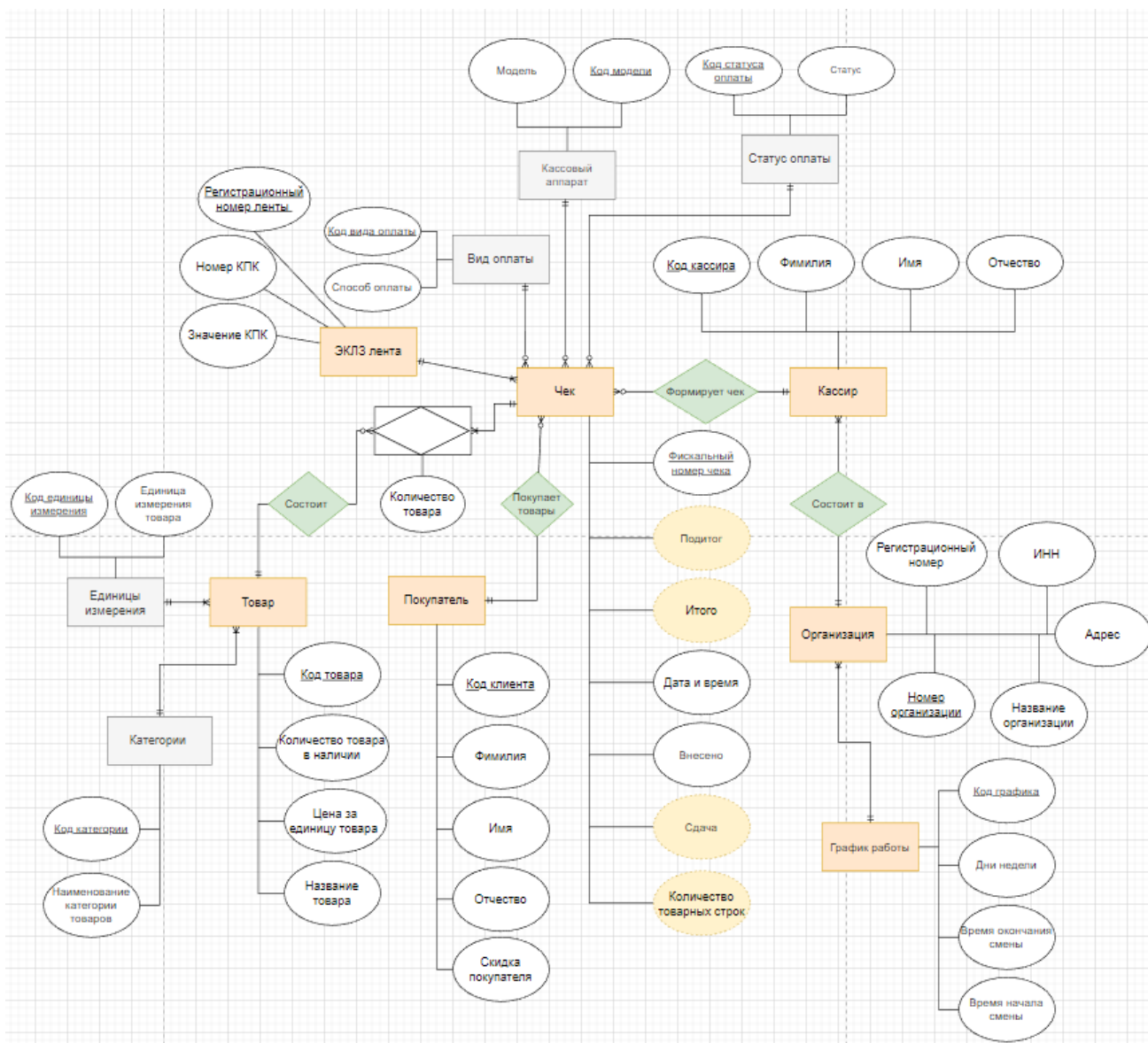


Рис. 1.7. ERD

Результатом данного этапа стало построение ER диаграммы, на основе выполнения анализа предметной области, все цели этапа были достигнуты.

#### 1.4. Реляционная схема базы данных

Цель - создать реляционную схему базы данных.

На основе ERD была составлена реляционная схема базы данных (рис. 1.8).

	Схема БД (реляционная)
1	Организация( <u>НомОрг</u> , НазвОрг, ИНН, ГрафРаб, Адрес)
2	Кассир( <u>КодКас</u> , <b>НомОрг</b> , Фамилия, Имя, Отчество)
3	Категории( <u>КодКат</u> , Наим)
4	Единицы_Измерения( <u>КодЕдИзм</u> , ЕдИзмер)
5	Товар( <u>КодТовара</u> , <b>КодЕдИзм</b> , <b>КодКат</b> , КолТовСклад, ЦенаТов, НазвТов)
6	ЭКЛЗ_Лента( <u>РегНомЛенты</u> , Ном, Знач)
7	Покупатель( <u>КодКлиента</u> , Фамилия, Имя, Отчество, Скидка)
8	Вид_Оплаты( <u>КодВидаОпл</u> , СпособОпл)
9	Кассовый_Аппарат( <u>КодМод</u> , Модель)
10	Статус_Оплаты( <u>КодСтатОпл</u> , статус)
11	Товар_Чек( <b><u>ФискНомЧека</u></b> , <b><u>КодТовара</u></b> , КолТов)
12	Чек( <b><u>ФискНомЧека</u></b> , <b><u>КодСтОпл</u></b> , <b><u>КодВидаОпл</u></b> , <b><u>КодКлиента</u></b> , <b><u>КодКас</u></b> , <b><u>РегНомЛенты</u></b> , <b><u>КодМод</u></b> , НомЧекаЗасмену, Подитог, Итого, КолТовСтрок, ДатаВремя, Внесено, Сдача)

Рис. 1.8. Реляционная схема базы данных

Отношения находятся в 4НФ, так как не имеют многозначных зависимостей.

Таким образом, на данном этапе был проведен анализ предметной области, все цели и задачи данного этапа были достигнуты.

## 2. Реализация базы данных

### 2.1. Реализация БД на сервере

Цель - реализовать спроектированную базу данных на сервере.

Задачи:

1. Выполнить создание таблиц в соответствии с проектом БД;
2. Выбрать правильные типы данных;
3. Реализовать декларативные ограничения целостности;
4. Заполнить таблицы данными

В качестве среды для реализации базы данных на сервере, был выбран MySQL Server.

В первую очередь было выполнено создание таблиц и выбор типов данных для полей в них в соответствии с проектом БД (рис. 2.2 - 2.14). следующим этапом была реализация декларативных ограничений целостности (рис. 2.15 - 2.23). Завершающим этапом было заполнение таблицы данными (рис. 2.24 - 2.36).

Таким образом, на данном этапе была реализована база данных в среде MySQL Server, все цели и задачи данного этапа были достигнуты.

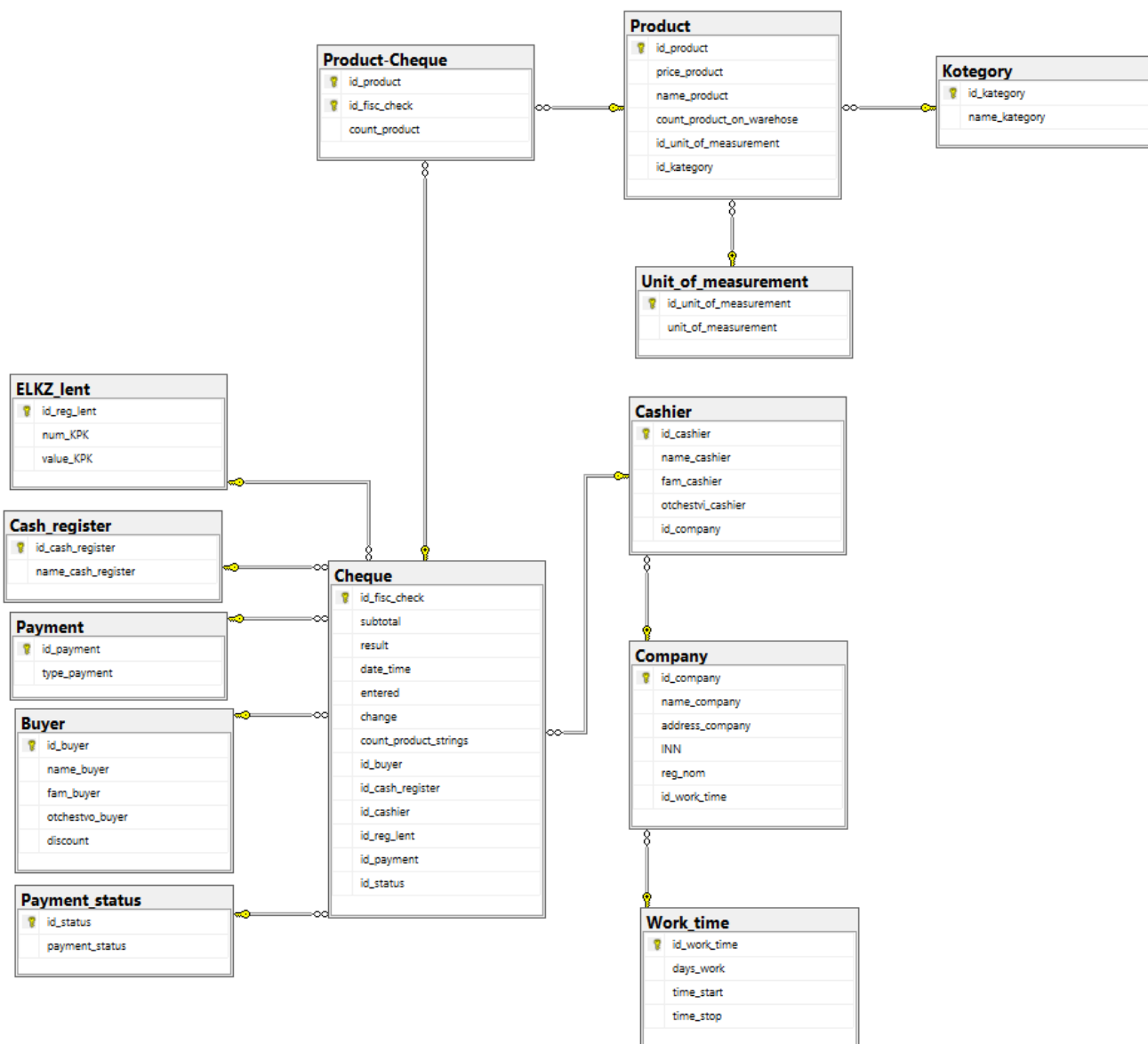


Рис. 2.1 Диаграмма БД

	Column Name	Data Type	Allow Nulls
🔑	id_buyer	int	<input type="checkbox"/>
	name_buyer	nvarchar(50)	<input type="checkbox"/>
	fam_buyer	nvarchar(50)	<input type="checkbox"/>
	otchestvo_buyer	nvarchar(50)	<input checked="" type="checkbox"/>
	discount	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.2. Типы данных и ограничения на значение NULL таблицы Buyer


	Column Name	Data Type	Allow Nulls
	id_cash_register	int	<input type="checkbox"/>
	name_cash_register	nvarchar(50)	<input type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.3. Типы данных и ограничения на значение NULL таблицы Cash\_register


	Column Name	Data Type	Allow Nulls
	id_cashier	int	<input type="checkbox"/>
	name_cashier	nvarchar(50)	<input type="checkbox"/>
	fam_cashier	nvarchar(50)	<input type="checkbox"/>
	otchestvi_cashier	nvarchar(50)	<input checked="" type="checkbox"/>
	id_company	int	<input type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.4. Типы данных и ограничения на значение NULL таблицы Cashier


	Column Name	Data Type	Allow Nulls
	id_fisc_check	bigint	<input type="checkbox"/>
	subtotal	float	<input checked="" type="checkbox"/>
	result	float	<input checked="" type="checkbox"/>
	date_time	datetime	<input checked="" type="checkbox"/>
	entered	float	<input checked="" type="checkbox"/>
	change	float	<input checked="" type="checkbox"/>
	count_product_strings	smallint	<input checked="" type="checkbox"/>
	id_buyer	int	<input checked="" type="checkbox"/>
	id_cash_register	int	<input type="checkbox"/>
	id_cashier	int	<input type="checkbox"/>
	id_reg_lent	bigint	<input type="checkbox"/>
	id_payment	tinyint	<input checked="" type="checkbox"/>
	id_status	tinyint	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.5. Типы данных и ограничения на значение NULL таблицы Cheque

	Column Name	Data Type	Allow Nulls
▶ 🔑	id_company	int	<input type="checkbox"/>
	name_company	nvarchar(50)	<input type="checkbox"/>
	address_company	nvarchar(100)	<input type="checkbox"/>
	INN	nchar(12)	<input type="checkbox"/>
	reg_nom	nchar(12)	<input type="checkbox"/>
	id_work_time	int	<input type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.6. Типы данных и ограничения на значение NULL таблицы Company

	Column Name	Data Type	Allow Nulls
▶ 🔑	id_reg_lent	bigint	<input type="checkbox"/>
	num_KPK	nchar(8)	<input type="checkbox"/>
	value_KPK	nchar(7)	<input type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.7. Типы данных и ограничения на значение NULL таблицы ELKZ\_lent

	Column Name	Data Type	Allow Nulls
▶ 🔑	id_kategory	smallint	<input type="checkbox"/>
	name_kategory	nvarchar(50)	<input type="checkbox"/>
			<input type="checkbox"/>

Рис. 2.8. Типы данных и ограничения на значение NULL таблицы Kategory

	Column Name	Data Type	Allow Nulls
▶ 🔑	id_payment	tinyint	<input type="checkbox"/>
	type_payment	varchar(11)	<input type="checkbox"/>
▶			<input type="checkbox"/>

Рис. 2.9. Типы данных и ограничения на значение NULL таблицы Payment



	Column Name	Data Type	Allow Nulls
	id_status	tinyint	<input type="checkbox"/>
	payment_status	nvarchar(10)	<input type="checkbox"/>
		<input type="text" value=""/>	<input type="checkbox"/>

Рис. 2.10. Типы данных и ограничения на значение NULL таблицы  
Payment\_status



	Column Name	Data Type	Allow Nulls
	id_product	bigint	<input type="checkbox"/>
	price_product	money	<input type="checkbox"/>
	name_product	varchar(50)	<input type="checkbox"/>
	count_product_on_ware...	float	<input type="checkbox"/>
	id_unit_of_measurement	smallint	<input type="checkbox"/>
	id_kategory	smallint	<input type="checkbox"/>
		<input type="text" value=""/>	<input type="checkbox"/>

Рис. 2.11. Типы данных и ограничения на значение NULL таблицы Product




	Column Name	Data Type	Allow Nulls
	id_product	bigint	<input type="checkbox"/>
	id_fisc_check	bigint	<input type="checkbox"/>
	count_product	float	<input type="checkbox"/>
		<input type="text" value=""/>	<input type="checkbox"/>

Рис. 2.12. Типы данных и ограничения на значение NULL таблицы  
Product-Cheque



	Column Name	Data Type	Allow Nulls
	id_unit_of_measurement	smallint	<input type="checkbox"/>
	unit_of_measurement	nvarchar(50)	<input type="checkbox"/>
		<input type="text" value=""/>	<input type="checkbox"/>

Рис. 2.13. Типы данных и ограничения на значение NULL таблицы  
Unit\_of\_measurement



	Column Name	Data Type	Allow Nulls
?	id_work_time	int	<input type="checkbox"/>
	days_work	nvarchar(50)	<input type="checkbox"/>
	time_start	time(7)	<input type="checkbox"/>
	time_stop	time(7)	<input type="checkbox"/>

Рис. 2.14. Типы данных и ограничения на значение NULL таблицы Work\_time

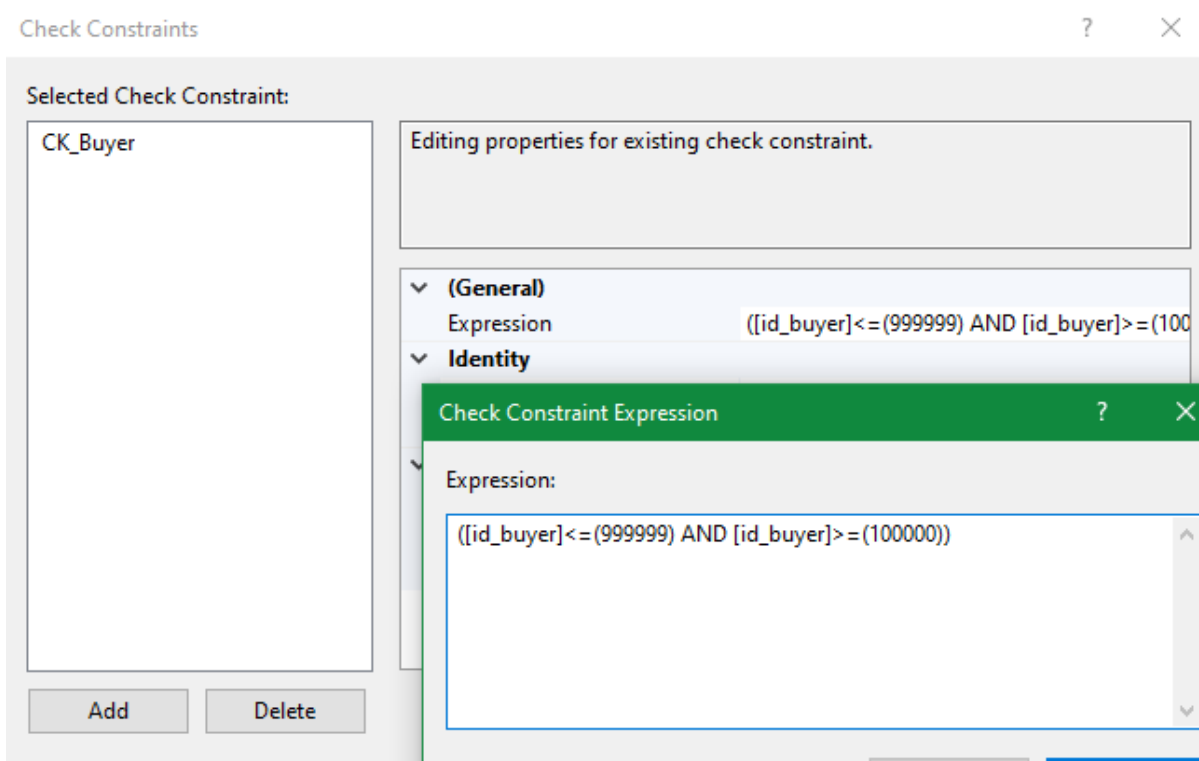


Рис. 2.15. Декларативные ограничения целостности таблицы Buyer

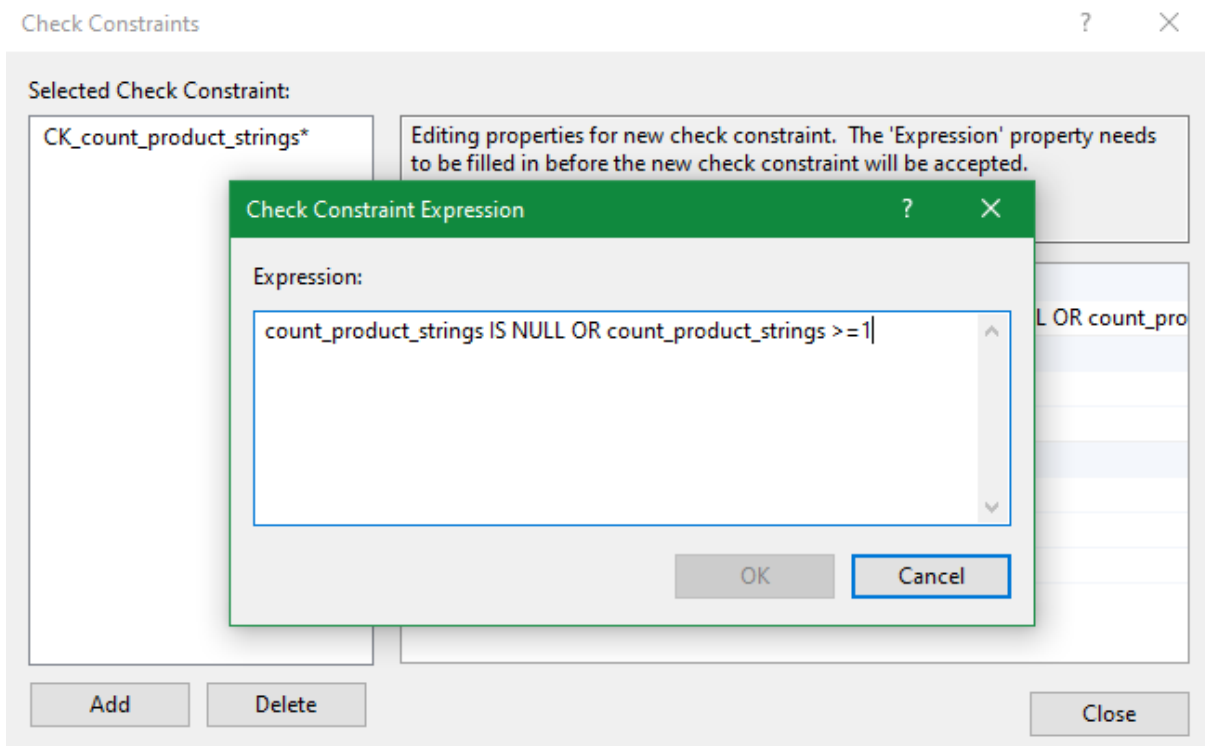


Рис. 2.16. Декларативные ограничения целостности таблицы Cheque

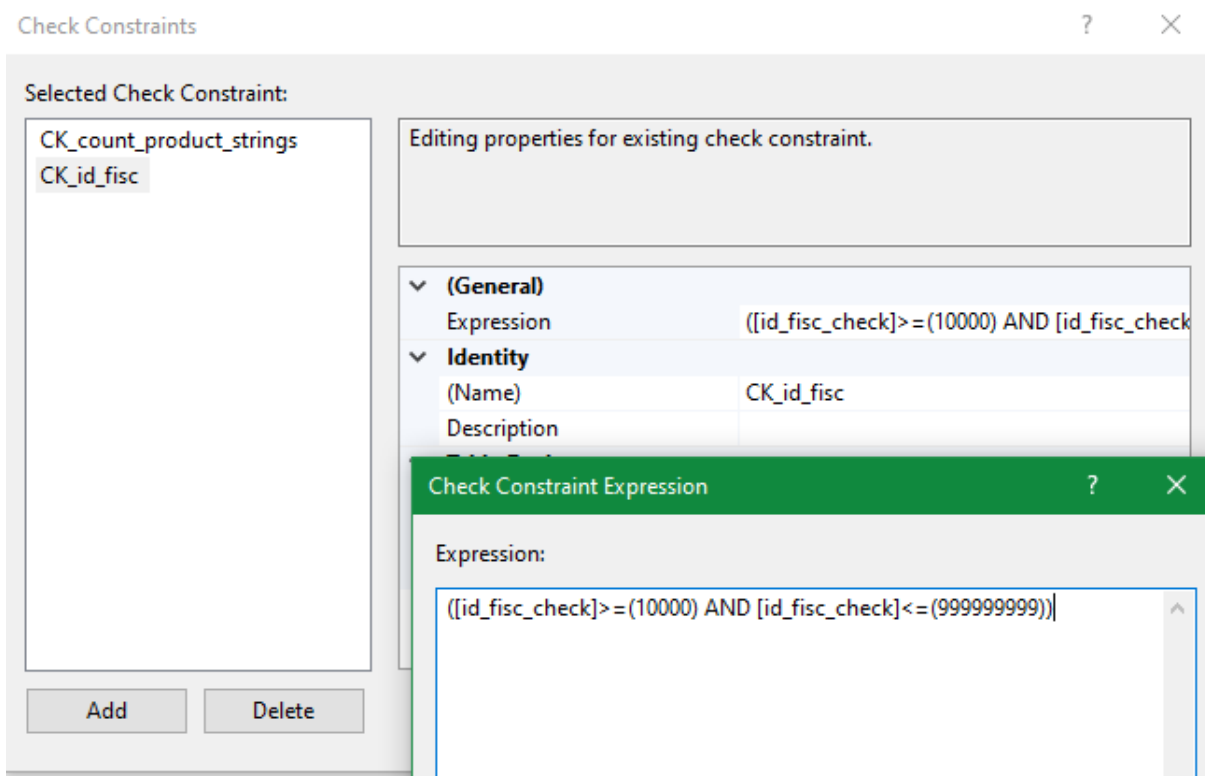


Рис. 2.17. Декларативные ограничения целостности таблицы Cheque

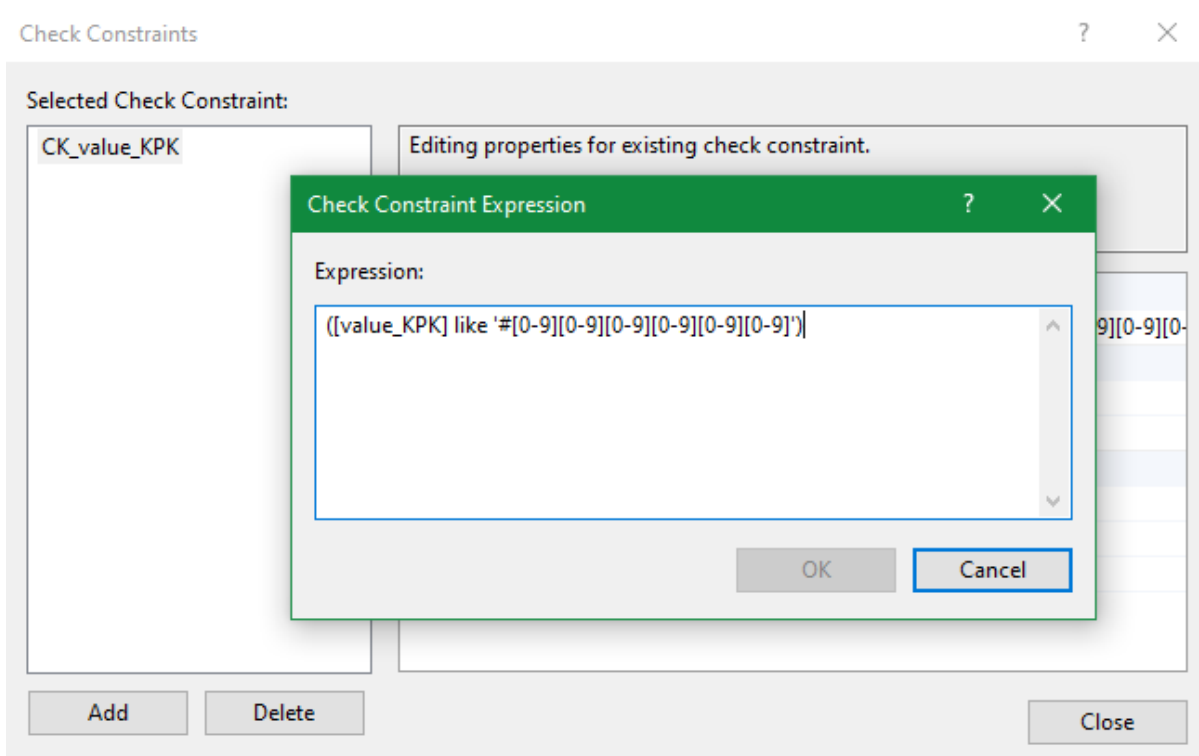


Рис. 2.18. Декларативные ограничения целостности таблицы ELKZ\_lent

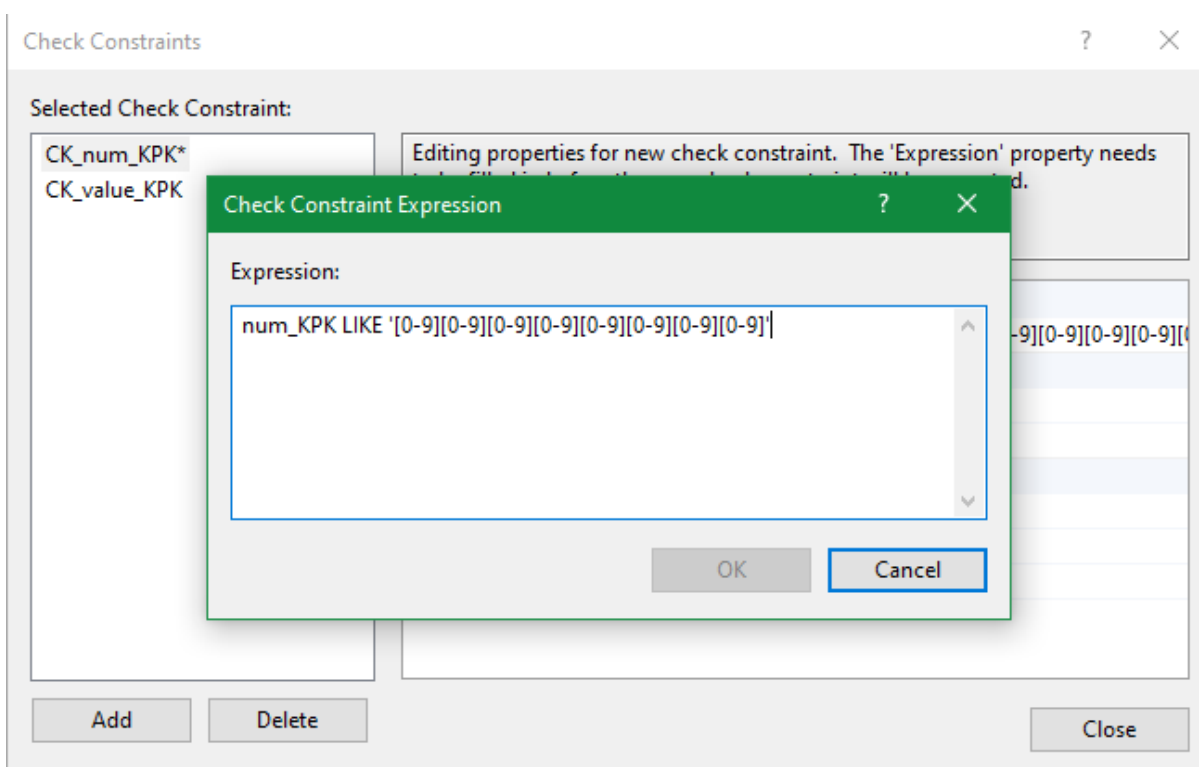


Рис. 2.19. Декларативные ограничения целостности таблицы ELKZ\_lent

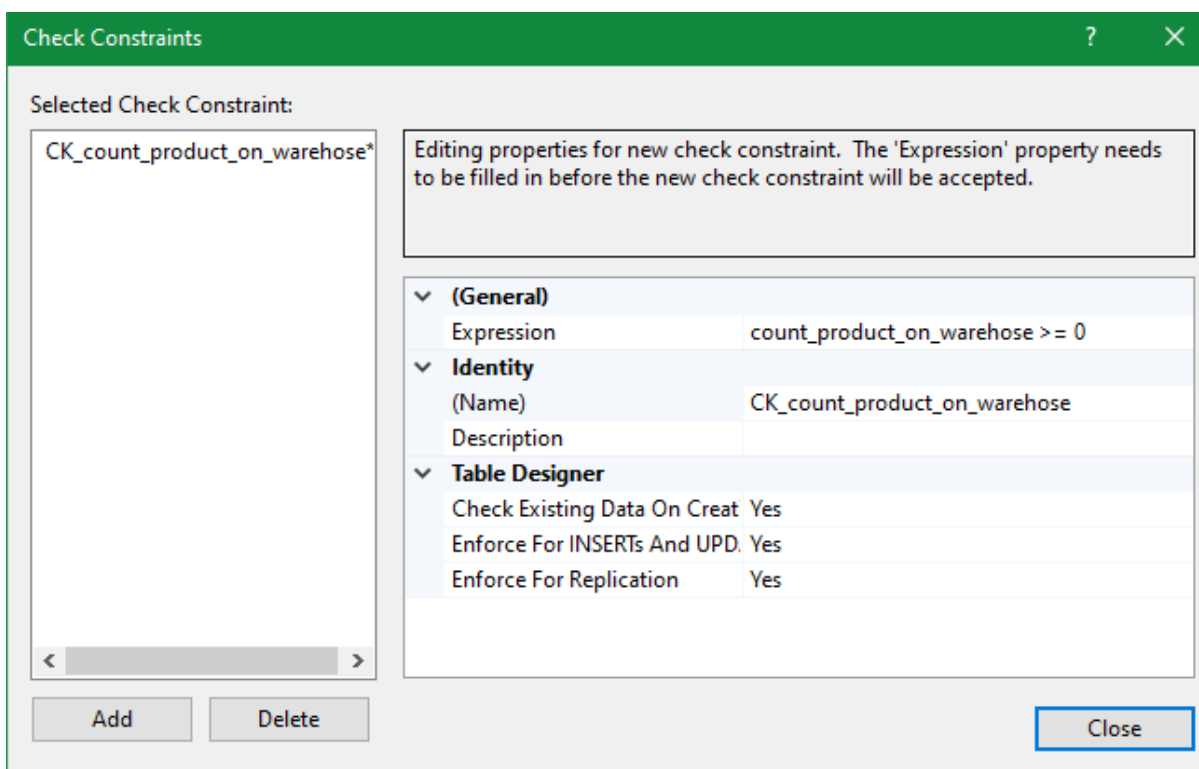


Рис. 2.20. Декларативные ограничения целостности таблицы Product

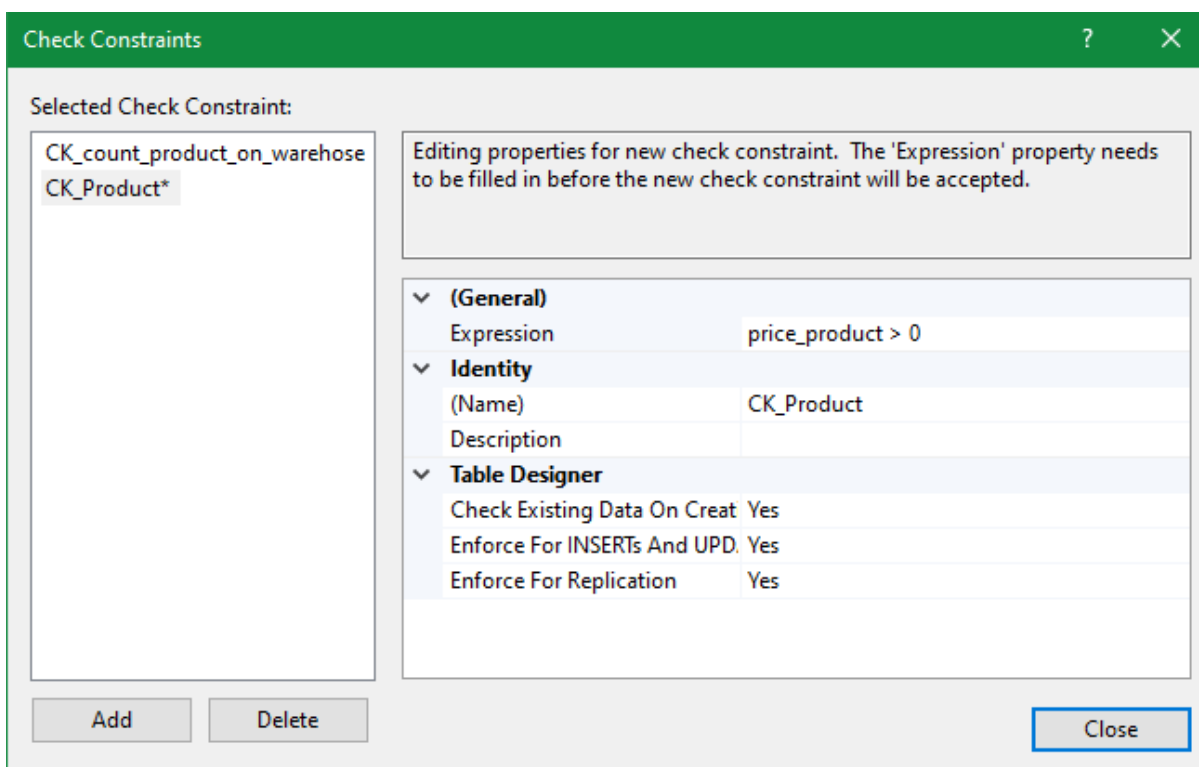


Рис. 2.21. Декларативные ограничения целостности таблицы Product

Check Constraints

Selected Check Constraint:

CK\_count\_product

Editing properties for existing check constraint.

<b>(General)</b> Expression: ([count_product]> (0))	
<b>Identity</b> (Name): CK_count_product Description:	
<b>Table Designer</b> Check Existing Data On Creat: Yes Enforce For INSERTs And UPD: Yes Enforce For Replication: Yes	

Add Delete Close

Рис. 2.22. Декларативные ограничения целостности таблицы Product-Cheque

Check Constraints

Selected Check Constraint:

CK\_Work\_days\*

Editing properties for new check constraint. The 'Expression' property needs to be filled in before the new check constraint will be accepted.

<b>(General)</b> Expression: days_work LIKE '[A-Я][a-я] - [A-Я][a-я]'	
<b>Identity</b> (Name): CK_Work_days Description:	
<b>Table Designer</b> Check Existing Data On Creat: Yes Enforce For INSERTs And UPD: Yes Enforce For Replication: Yes	

Add Delete Close

Рис. 2.23. Декларативные ограничения целостности таблицы Work\_time

	id_buyer	name_buyer	fam_buyer	otchestvo_buyer	discount
►	100001	Георгий	Петров	Львович	NULL
	200001	Сергей	Павлович	NULL	20
	200002	Павел	Васильков	Сергеевич	10
	200003	Виктор	Клопов	Павлович	15
	200004	Даниил	Сметанов	Павлович	10
	200005	Николай	Соловьев	Михайлович	5
	200006	Смирнов	Никита	Павлович	NULL
	200007	Смирнов	Никита	Михайлович	NULL
	200008	Чиколин	Илья	Батькович	20
	200009	Павел	Миронычев	Сергеевич	NULL
	200010	Бухарин	Семен	Николаевич	NULL
	200011	Еся	Гольдбаум	Маркович	2

Рис. 2.24. Значения таблицы Buyer

	id_cash_register	name_cash_re...
►	1	АТОЛ 91Ф
	2	Эвотор 5
	3	Меркурий-115Ф
	4	ПИОНЕР-114Ф...
	5	АТОЛ 25Ф
	6	АТОЛ 11Ф
	7	АТОЛ 27Ф
	8	АТОЛ 50Ф
	9	АТОЛ 22 v2 Ф

Рис. 2.25. Значения таблицы Cash\_register

	id_cashier	name_cashier	fam_cashier	otchestvi_cash...	id_company
►	3	Никита	Николаев	Павлович	1
	5	Дмитрий	Богданов	Глебович	5
	6	Владимир	Лебедев	Сергеевич	5
	7	Юлия	Фролова	Олеговна	7
	8	Анна	Сметанова	Георгиевна	7
	9	Юлия	Шумилова	Дмитрьевна	7
	10	Анастасия	Сараева	NULL	5
	11	Елена	Васильева	Максимовна	1

Рис. 2.26. Значения таблицы Cashier

	id_fisc_check	subtotal	result	date_time	entered	change	count_product...	id_buyer	id_cash_register	id_cashier	id_req_lent	id_payment	id_status
►	10001	54299,9	48869,91	2022-12-27 22:3...	62000	13130,09	4	200001	1	3	1000000000	1	3
	30001	66119	66119	2022-12-27 22:3...	70000	3881	3	200002	2	5	1000000005	1	3
	30003	142880	121448	2022-12-27 22:3...	121448	0	2	200001	3	11	1000000005	2	3
	30004	30760,89	24608,71	2022-12-27 22:3...	24608,71	0	3	200001	3	11	1000000008	2	3
	30005	8110	7299	2022-12-27 22:3...	7300	1	1	200001	5	11	1000000008	1	3
	30006	868,71	868,71	2022-12-27 22:3...	868,71	0	5	200002	1	8	1000000001	2	3
	30008	670,68	603,61	2022-12-27 22:3...	603,61	0	5	200002	1	7	1000000001	2	3
	30009	499,38	449,44	2022-12-27 22:3...	600	150,56	4	200002	1	9	1000000002	1	3
	30011	2731,05	2731,05	2022-12-27 22:3...	3000	268,95	4	200003	4	7	1000000002	1	3
	30012	52990	52990	2022-12-27 22:3...	52990	0	1	200003	9	10	1000000004	2	3
	30013	27990	27990	2022-12-27 22:3...	27990	0	1	200003	6	3	1000000000	2	3
	30014	22570	22570	2022-12-27 22:3...	22570	0	2	200004	6	3	1000000000	2	3
	30015	52110	52110	2022-12-27 22:3...	55000	2890	2	200004	8	11	1000000008	1	3
	30016	157,63	157,63	2022-12-27 22:3...	157,63	0	2	200004	2	9	1000000002	2	3
	30017	1253,77	1253,77	2022-12-27 22:3...	1500	246,23	2	200005	2	9	1000000002	1	3
	30018	594,3	594,3	2022-12-29 14:2...	1000	405,7	4	200005	7	8	1000000007	1	3
	30024	3690	3505,5	2022-12-27 22:3...	3505,5	0	1	200005	8	11	1000000008	2	3
	30025	9580	9580	2022-12-27 22:3...	9580	0	1	NULL	4	7	1000000002	2	3
	30026	55898,99	55898,99	2022-12-27 22:3...	60000	4101,01	2	NULL	4	7	1000000002	1	3
	30027	683,88	683,88	2022-12-27 22:3...	684	0,12	1	200011	7	8	1000000007	1	3
	30028	134970	107976	2022-12-27 23:0...	110006	2030	1	200008	7	3	1000000001	1	3

Рис. 2.27. Значения таблицы Cheque

	id_company	name_company	address_comp...	INN	req_nom	id_work_time
	1	ООО "Citilink"	Костромская о...	771897930757	114774646142	4
	5	ООО "DNS"	Приморский к...	254016706144	110254000823	4
	7	ООО "Максид...	С-Петербург Д...	007804064663	000000101712	1

Рис. 2.28. Значения таблицы Company

	id_req_lent	num_KPK	value_KPK
	1000000000	94865490	#321156
	1000000001	65168571	#165565
	1000000002	20100225	#065456
	1000000004	15646559	#404555
	1000000005	56452546	#541621
	1000000006	52565565	#124765
	1000000007	21051265	#420455
	1000000008	12155155	#945124
	1000000009	50001216	#598448
	1000000010	64945547	#516465

Рис. 2.29. Значения таблицы ELKZ\_lent

	id_kateqory	name_kateqory
	1	GPU
	2	CPU
	3	SSD
	4	Холодильники
	5	Газовые плиты
	6	Телевизоры
	7	Смартфоны
	8	Материнские ...
	9	Блоки питания
	10	Мониторы
	11	Клавиатуры
	12	Компьютерны...
	13	HDD
	14	Оперативная п...
	15	Крупы
	16	Газировки
	17	Фрукты
	18	Консервы

Рис. 2.30. Значения таблицы Category

	id_payment	type_payment
	1	Наличные
	2	Безналичная

Рис. 2.31. Значения таблицы Payment

	id_status	payment_status
	2	Не оплачен
	3	Оплачен

Рис. 2.32. Значения таблицы Payment\_status



	id_product	price_product	name_product	count_product...	id_unit_of_me...	id_category
	1000000000000	13290,0000	Intel Core i5 11...	618	1	2
	1000000000001	6790,0000	Intel Core i3 10...	65	1	2
	1000000000002	14890,0000	Intel Core i5 12...	9	1	2
	1000000000003	10890,0000	AMD Ryzen 5 3...	79	1	2
	1000000000004	15990,0000	AMD Ryzen 5 5...	12	1	2
	1000000000005	13690,0000	AMD Ryzen 5 5...	6	1	2
	1000000000006	30590,0000	Palit NVIDIA Ge...	152	1	1
	1000000000007	27990,0000	Palit NVIDIA Ge...	560	1	1
	1000000000008	30990,0000	MSI NVIDIA Ge...	492	1	1
	1000000000009	44990,0000	GIGABYTE NVID...	91	1	1
	1000000000010	39990,0000	MSI NVIDIA Ge...	88	1	1
	1000000000011	55590,0000	Palit NVIDIA Ge...	5	1	1
	1000000000012	3290,0000	Kingston A400 ...	49	1	3
	1000000000013	1890,0000	AMD Radeon R...	894	1	3
	1000000000014	3090,0000	Kingston NV1 S...	933	1	3
	1000000000015	3290,0000	KINGSPEC NX-...	88	1	3
	1000000000016	1190,0000	NETAC N600S ...	0	1	3
	1000000000017	4790,0000	Samsung 870 E...	48	1	3
	1000000000018	44990,0000	Beko B5RCNK4...	65	1	4
	1000000000019	38990,0000	Beko RCNK335...	50	1	4
	1000000000021	8110,0000	Бирюса Б-50	64	1	4
	1000000000022	30581,9900	STINOL STS 185 S	14	1	4
	1000000000025	27990,0000	Indesit DS 4200 ...	10	1	4
	1000000000026	15890,0000	GEFEST ПГ 3200...	5	1	5
	1000000000027	13120,0000	GEFEST ПГ 3200...	65	1	5
	1000000000028	37000,0000	GEFEST ПГЭ 55...	56	1	5
	1000000000029	27850,5000	GEFEST ПГЭ 51...	9	1	5
	1000000000030	54990,0000	Samsung UE50...	7	1	6
	1000000000031	52990,0000	Samsung UE55...	98	1	6
	1000000000032	34190,0000	Hisense 55A6B...	12	1	6
	1000000000033	15990,0000	REALME 8 6/12...	78	1	7
	1000000000034	11990,0000	vivo Y35 4/64Gb	57	1	7
	1000000000035	81700,0000	Samsung Galax...	14	1	7
	1000000000036	12990,0000	REALME 8i 4/64...	25	1	7
	1000000000037	6090,0000	MSI H510M-A ...	65	1	8
	1000000000038	7390,0000	ASUS PRIME H5...	22	1	8
	1000000000039	6490,0000	GIGABYTE H510...	25	1	8
►	1000000000040	9890,0000	ASUS PRIME B5...	42	1	8

Рис. 2.33. Значения таблицы Product

	id_product	id_fisc_check	count_product
	1000000000000	10001	1
	1000000000003	30001	1
	1000000000004	30014	1
	1000000000006	30003	2
	1000000000007	30013	1
	1000000000008	10001	1
	1000000000009	30028	3
	1000000000012	30014	2
	1000000000014	10001	2
	1000000000016	30004	1
	1000000000017	30025	2
	1000000000019	30015	1
	1000000000021	30005	1
	1000000000022	30026	1
	1000000000027	30015	1
	1000000000030	30001	1
	1000000000031	30012	1
	1000000000035	30003	1
	1000000000055	30026	3
	1000000000057	10001	1
	1000000000059	30001	1
	1000000000061	30024	1
	1000000000065	30004	2
	1000000000072	30004	1
	1000000000073	30008	1,55
	1000000000073	30017	21,75
	1000000000073	30027	12
	1000000000074	30011	0,5
	1000000000075	30006	0,95
	1000000000075	30008	1,753
	1000000000077	30006	1,5
	1000000000077	30009	0,33
	1000000000077	30018	1
	1000000000078	30008	1
	1000000000078	30016	1,5
	1000000000079	30018	1
	1000000000080	30009	2
►	1000000000080	30017	0,4

Рис. 2.34. Значения таблицы Product-Cheque

	id_unit_of_me...	unit_of_measu...
	1	шт.
	2	кг
	3	л

Рис. 2.35. Значения таблицы Unit\_of\_measurement

	id_work_time	days_work	time_start	time_stop
	1	Пн - Вс	08:00:00	22:00:00
	2	Пн - Пт	08:00:00	20:00:00
	3	Пн - Вс	10:00:00	18:00:00
	4	Пн - Вс	10:00:00	20:00:00
	5	Пн - Вс	00:00:00	23:59:59

Рис. 2.36. Значения таблицы Work\_time

## 2.2. Реализация объектов базы данных

Цель - реализовать объекты БД.

Задачи:

1. Выполнить создание представлений, хранимых процедур, триггеров в соответствии с проектом.
2. Аналитические запросы реализовать в виде хранимых процедур с параметрами.

В соответствии с проектом были созданы триггеры (рис. 2.37 - 2.40), хранимые процедуры (рис. 2.41 - 2.51, 2.53) и представление (рис. 2.52). Далее были созданы аналитические запросы, реализованные в виде хранимых процедур с параметрами (рис. 2.54 - 2.58).

Также для будущей реализации приложения для работы с БД, была создана и использовалась таблица кодов завершения процедур (табл. 1).

Таблица кодов завершения процедур

Код завершения	Что произошло
-3	Ошибка, нельзя менять параметры в уже закрытом чеке
-2	Ошибка, нельзя удалить чек, который имеет статус “оплачен”
-1	Ошибка, в чеке нет продуктов
0	Ошибка, введены некорректные данные
1	Успешное обновление таблицы
2	Успешное удаление записи
3	Успешное добавление записи
4	Запрос(ы) успешно завершён(ы)
5	Курсор выполнил функцию удаления товара со склада

Таким образом, на данном этапе были реализованы объекты базы данных, все цели и задачи данного этапа были достигнуты.

```

USE [Продажа_товаров]
GO
/***** Object: Trigger [dbo].[ChangeStatusTrigger]    Script Date: 29.12.2022 19:41:22 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[ChangeStatusTrigger]
ON [dbo].[Cheque]
AFTER UPDATE
AS
BEGIN
    IF @@ROWCOUNT=1 -- Проверка на вставку одной записи
    BEGIN
        -- Объявляем переменные
        DECLARE @enter as float,
                @check as bigint,
                @payment as tinyint,
                @result as float,
                @err as tinyint

        -- Присваиваем переменным значения из добавляемой записи
        SELECT  @enter = inserted.[entered],
                @check = inserted.[id_fisc_check],
                @result = inserted.[result],
                @payment = inserted.[id_payment]
        FROM inserted

        EXEC @err = [Продажа_товаров].[dbo].[CalculationChange] @enter, @check, @payment, @result
        IF (@err = 1)
        BEGIN
            EXEC @err = [Продажа_товаров].[dbo].[UpdateStatus] @check
        END
    END
END

```

Рис. 2.37. Триггер для таблицы Cheque для события обновления

```

ALTER TRIGGER [dbo].[AddingExisting]
ON [dbo].[Product-Cheque]
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @Prod as bigint,
            @check as bigint,
            @coll as float,
            @wascoll as float,
            @count_on_warehouse as float

    -- Присваиваем переменным значения из добавляемой записи
    SELECT @Prod = inserted.[id_product],
           @check = inserted.[id_fisc_check],
           @coll = inserted.[count_product]
    FROM inserted

    SELECT @count_on_warehouse = (SELECT count_product_on_warehouse FROM Product WHERE id_product = @Prod)--кол тов на складе

    IF ( @Prod in (SELECT id_product FROM [Product-Cheque] WHERE id_fisc_check = @check) )
    BEGIN
        SELECT @wascoll = (SELECT count_product FROM [Product-Cheque] WHERE id_fisc_check = @check AND id_product = @Prod)
        IF ((@wascoll + @coll) <= @count_on_warehouse)
        BEGIN
            UPDATE dbo.[Product-Cheque]
            SET count_product = @coll + @wascoll
            WHERE id_fisc_check = @check AND id_product = @Prod
        END
    END
    ELSE
    BEGIN
        IF (@coll <= @count_on_warehouse)
        BEGIN
            INSERT INTO dbo.[Product-Cheque]
            VALUES (@Prod, @check, @coll)
        END
    END
END
END

```

Рис. 2.38. Триггер для таблицы Product-Cheque вместо события добавления новой записи

```

USE [Продажа_товаров]
GO
/***** Object: Trigger [dbo].[ItogTrigger]    Script Date: 29.12.2022 19:44:00 ****
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[ItogTrigger]
ON [dbo].[Product-Cheque]
FOR INSERT, UPDATE
AS
BEGIN
    IF @@ROWCOUNT=1 -- Проверка на вставку одной записи
    BEGIN
        -- Объявляем переменные
        DECLARE @Prod as bigint,
                @check as bigint,
                @coll as float,
                @err as tinyint

        -- Присваиваем переменным значения из добавляемой записи
        SELECT  @Prod = inserted.[id_product],
                @check = inserted.[id_fisc_check],
                @coll = inserted.[count_product]
                FROM inserted

        EXEC @err = [Продажа_товаров].[dbo].[CalculationResult] @Prod, @check, @coll

        --EXEC [Продажа_товаров].[dbo].[UpdateStatus] @check
    END
END

```

Рис. 2.39. Триггер для таблицы Product-Cheque для событий обновления и добавления записей

```

USE [Продажа_товаров]
GO
/***** Object: Trigger [dbo].[ItogTriggerDel]    Script Date: 29.12.2022 19:44:36 */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[ItogTriggerDel]
    ON [dbo].[Product-Cheque]
    AFTER DELETE
    AS
BEGIN
    IF @@ROWCOUNT=1 -- Проверка на вставку одной записи
    BEGIN
        -- Объявляем переменные
        DECLARE @Prod as bigint,
                @check as bigint,
                @coll as float,
                @err as int

        -- Присваиваем переменным значения из добавляемой записи
        SELECT @Prod = deleted.[id_product],
               @check = deleted.[id_fisc_check],
               @coll = deleted.[count_product]
        FROM deleted

        EXEC @err = [Продажа_товаров].[dbo].[CalculationResult] @Prod, @check, @coll

    END
END

```

Рис. 2.40. Триггер для таблицы Cheque для события обновления



```

ALTER PROCEDURE [dbo].[AddChequeByer]
    @id_cheque as bigint,
    @id_buyer as int
AS
BEGIN
    If(@id_buyer in (SELECT id_buyer FROM Buyer))
    BEGIN
        UPDATE Cheque
        SET id_buyer = @id_buyer
        WHERE id_fisc_check = @id_cheque
        RETURN 1
    END
    ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.41. Процедура добавления в таблицу Cheque покупателя

```

ALTER PROCEDURE [dbo].[AddChequeEnteredCash]
    @id_cheque AS bigint,
    @entered as nvarchar(11)
AS
BEGIN
    If(@id_cheque in (SELECT id_fisc_check FROM Cheque))
    BEGIN
        If((SELECT id_status FROM Cheque WHERE id_fisc_check = @id_cheque) = 2)
        BEGIN
            UPDATE Cheque
            SET entered = @entered
            WHERE id_fisc_check = @id_cheque
            RETURN 1
        END
        ELSE
        BEGIN
            RETURN -3
        END
    END
    ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.42. Процедура добавления в таблицу Cheque суммы принятых  
наличных

```
ALTER PROCEDURE [dbo].[AddChequePayment]
    @id_cheque AS bigint,
    @name_payment as nvarchar(11)
AS
BEGIN
    If(@name_payment in (SELECT type_payment FROM Payment)
    AND @id_cheque in (SELECT id_fisc_check FROM Cheque))
    BEGIN

        DECLARE @id_payment as int

        IF (@name_payment = 'Наличные')
        BEGIN
            SELECT @id_payment = 1
        END
        ELSE
        BEGIN
            SELECT @id_payment = 2
        END
        UPDATE Cheque
        SET id_payment = @id_payment
        WHERE id_fisc_check = @id_cheque
        RETURN 1

    END
    ELSE
    BEGIN
        RETURN 0
    END
END
```

Рис. 2.43. Процедура добавления в таблицу Cheque типа оплаты

```

ALTER PROCEDURE [dbo].[AddChequeProducts]
    @id_cheque as bigint,
    @id_product as bigint,
    @count_prod as float
AS
BEGIN
    If(@id_cheque in (SELECT id_fisc_check FROM Cheque)
        AND @id_product in (SELECT id_product FROM Product))
    BEGIN
        INSERT INTO [Product-Cheque] (id_fisc_check, id_product, count_product)
        VALUES (@id_cheque, @id_product, @count_prod)
        RETURN 3
    END
    ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.44. Процедура добавления в таблицу Product-Cheque продуктов  
для чека

```

ALTER Procedure [dbo].[CalculationResult]
    @Prod as bigint,
    @check as bigint,
    @coll as float
AS
BEGIN
    IF (@coll > 0 AND @check in (SELECT id_fisc_check FROM Cheque) AND
        @Prod in (SELECT id_product FROM Product))
    BEGIN
        DECLARE @subtotal as float,
                @result as float,
                @count_prod as smallint,-- объявляем переменную для
подсчета записей
                @bimbim as float,
                @avgz as float,

```

@idBuyer as int

```
SELECT @count_prod = (Select count(*) from [Product-Cheque]
where id_fisc_check = @check) -- Кол товарных строк
SELECT @idBuyer = (SELECT id_buyer FROM Cheque WHERE
id_fisc_check = @check) --id покупателя
```

```
SELECT @bimbim = (SELECT discount FROM Buyer WHERE
id_buyer = @idBuyer ) --Скидка
```

```
SELECT @subtotal = ROUND ( (SELECT
sum(price_product*count_product) FROM (SELECT * FROM [Product-Cheque]
WHERE id_fisc_check = @check) PC join Product P
ON PC.id_product = P.id_product), 2) -- подитог
```

```
SELECT @result = ROUND (((@subtotal - @subtotal * (SELECT
CASE WHEN @bimbim Is NOT NULL THEN @bimbim ELSE 0 END
discount)/100) , 2)--итог
```

```
UPDATE dbo.Cheque
SET subtotal = @subtotal,
    result = @result,
    count_product_strings = @count_prod
WHERE id_fisc_check = @check
```

--обновление скидки пользователя

```
SELECT @avgz = (SELECT avg(result) FROM Cheque
WHERE id_buyer IS NOT NULL AND id_buyer = @idBuyer
```

GROUP By id\_buyer) --ср значение итоговой суммы одного чека с учетом скидок

```
IF (@avgz < 500)
BEGIN
    UPDATE dbo.Buyer
    SET discount = NULL
    WHERE id_buyer = @idBuyer
END
ELSE
BEGIN
    IF (@avgz < 1000)
    BEGIN
        UPDATE dbo.Buyer
        SET discount = 2
        WHERE id_buyer = @idBuyer
    END
    ELSE
    BEGIN
        IF (@avgz < 5000)
        BEGIN
            UPDATE dbo.Buyer
            SET discount = 5
            WHERE id_buyer = @idBuyer
        END
        ELSE
        BEGIN
            IF (@avgz < 25000)
            BEGIN
                UPDATE dbo.Buyer
```

```

        SET discount = 10
        WHERE id_buyer = @idBuyer
    END
    ELSE
    BEGIN
        IF (@avgz < 50000)
        BEGIN
            UPDATE dbo.Buyer
            SET discount = 15
            WHERE id_buyer = @idBuyer
        END
    ELSE
    BEGIN
        UPDATE dbo.Buyer
        SET discount = 20
        WHERE id_buyer = @idBuyer
    END
    END
    END
    END
    END
    RETURN 1
END
ELSE
BEGIN
    RETURN 0
END

```

Рис. 2.45. Процедура расчета итога, записываемого в таблицу Cheque с учетом скидки покупателя (при наличии)

```

ALTER PROCEDURE [dbo].[CalculationChange]
    @enter as float,
    @check as bigint,
    @payment as tinyint,
    @result as float
AS
BEGIN
    IF (@enter > 0
        AND @result > 0
        AND @check in (SELECT id_fisc_check FROM Cheque)
        AND @payment in (SELECT id_payment FROM Cheque))
    BEGIN
        DECLARE @change as float
        IF ( @payment = 1 )
        BEGIN
            SELECT @change = ROUND ((@enter - @result) , 2)
            IF (@change IS NULL)
            BEGIN
                SELECT @change = 0
            END
            IF ( @change > 0 )
            BEGIN
                UPDATE dbo.Cheque
                SET change = @change
                WHERE id_fisc_check = @check

                RETURN 1
            END
        ELSE
        BEGIN
            UPDATE dbo.Cheque
            SET change = NULL
            WHERE id_fisc_check = @check
            RETURN 1
        END
    END
    IF (@payment = 2)
    BEGIN
        SELECT @enter = @result
        SELECT @change = 0
        UPDATE dbo.Cheque
        SET change = @change,
            entered = @enter
        WHERE id_fisc_check = @check

        RETURN 1
    END
END
BEGIN
    RETURN 0
END

```

Рис. 2.46. Процедура расчета подитога, записываемого в таблицу Cheque с учетом выбранного типа оплаты

```

ALTER PROCEDURE [dbo].[UpdateStatus]
    @check as bigint
AS
BEGIN
    IF (@check in (SELECT id_fisc_check FROM Cheque))
    BEGIN
        DECLARE @payment as tinyint,
                @delWarehose as int
        SELECT @payment = (SELECT id_payment FROM Cheque WHERE id_fisc_check = @check )
        IF (@payment = 1)
        BEGIN
            IF ((SELECT change FROM Cheque WHERE id_fisc_check = @check) IS NULL)
            BEGIN
                UPDATE dbo.Cheque
                SET id_status = 2
                WHERE id_fisc_check = @check
                RETURN 1
            END
        ELSE
        BEGIN
            UPDATE dbo.Cheque
            SET id_status = 3,
                date_time = GETDATE()
            WHERE id_fisc_check = @check
            RETURN 1
            EXEC @delWarehose = dbo.DelitedFromWarehose @check
            RETURN @delWarehose
        END
    END
    ELSE
    BEGIN
        IF ((SELECT entered FROM Cheque WHERE id_fisc_check = @check) IS NOT NULL)
        BEGIN
            UPDATE dbo.Cheque
            SET id_status = 3,
                date_time = GETDATE()
            WHERE id_fisc_check = @check
            RETURN 1
            EXEC @delWarehose = dbo.DelitedFromWarehose @check
            RETURN @delWarehose
        END
    ELSE
    BEGIN
        UPDATE dbo.Cheque
        SET id_status = 2
        WHERE id_fisc_check = @check
        RETURN 1
    END
    END
    END
    ELSE
    BEGIN
        RETURN 0
    END
END
END

```

Рис. 2.47. Процедура обновления статуса чека и реализация бизнес-правила по добавлению в БД даты и времени оплаты после оплаты



```

]ALTER PROCEDURE [dbo].[DelProductFromCheque]
    @id_cheque AS bigint,
    @id_product AS bigint,
    @count as float
AS
]BEGIN
]    If(@id_cheque in (SELECT id_fisc_check FROM [Product-Cheque])
        AND @count > 0
        AND @id_product in (SELECT id_product FROM [Product-Cheque]))
]    BEGIN
]        IF (@count < (SELECT count_product FROM [Product-Cheque]
                        WHERE id_fisc_check = @id_cheque AND @id_product = id_product))
]        BEGIN
]            UPDATE [Product-Cheque]
]            SET count_product = count_product - @count
]            WHERE id_fisc_check = @id_cheque AND @id_product = id_product
]            RETURN 1
]        END
]        ELSE
]        BEGIN
]            DELETE FROM [Product-Cheque]
]            WHERE id_fisc_check = @id_cheque AND @id_product = id_product
]            RETURN 2
]        END
]    END
]    ELSE
]    BEGIN
]        RETURN 0
]    END
]END

```

---

Рис. 2.48. Процедура удаления определенного количества продуктов из  
таблицы Product-Cheque

```

ALTER PROCEDURE [dbo].[DelitedFromWarehose]
    @check as bigint
AS
BEGIN
    If(@check in (SELECT id_fisc_check FROM [Product-Cheque]))
    BEGIN
        DECLARE @id_prod AS bigint

        DECLARE MyCursor CURSOR FOR --создание курсора
        SELECT id_product FROM [Product-Cheque] WHERE id_fisc_check = @check
        OPEN MyCursor --открытие курсора
        FETCH next FROM MyCursor INTO @id_prod --переход к следующей записи в крсоре
        WHILE @@FETCH_STATUS = 0
        BEGIN
            EXEC('
                DECLARE @count_on_sklad AS float,
                        @count_prod AS float

                SELECT @count_on_sklad = (SELECT count_product_on_warehose FROM Product P
                JOIN [Product-Cheque] PC ON P.id_product = PC.id_product WHERE id_fisc_check = '+ @check + ' AND P.id_product = '+ @id_prod + ')

                SELECT @count_prod = (SELECT count_product FROM [Product-Cheque] P WHERE id_fisc_check = '+ @check + ' AND id_product = '+ @id_prod + ')

                IF (@count_on_sklad >= @count_prod AND @count_on_sklad > 0)
                BEGIN
                    UPDATE dbo.Product
                    SET count_product_on_warehose = @count_on_sklad - @count_prod
                    WHERE id_product = '+ @id_prod + '
                END
            ')
            FETCH next FROM MyCursor INTO @id_prod --переход к следующей записи в крсоре
        END
        CLOSE MyCursor --заккрытие курсора
        DEALLOCATE MyCursor --удаление курсора
        RETURN 5
    END
ELSE
    BEGIN
        RETURN -1
    END
END

```

Рис. 2.49. Процедура удаления купленного товара со склада в БД в  
таблице Product-Cheque

```

ALTER PROCEDURE [dbo].[DelitedCheque]
    @cheque as bigint
AS
BEGIN
    DECLARE @id_stat as int
    SELECT @id_stat = (SELECT id_status FROM Cheque WHERE id_fisc_check = @cheque)

    IF (@id_stat = 2 OR @id_stat IS NULL)
    BEGIN
        IF ((SELECT id_status FROM Cheque WHERE id_fisc_check = @cheque ) = 2)
        BEGIN
            if (@cheque in (SELECT id_fisc_check FROM [Product-Cheque]))
            BEGIN
                DELETE FROM [Product-Cheque]
                WHERE @cheque = id_fisc_check
            END

            DELETE FROM Cheque
            WHERE @cheque = id_fisc_check
            RETURN 2
        END
    ELSE
        BEGIN
            RETURN -2
        END
    END
ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.50. Процедура удаления чека из база данных (возможно пока чек не имеет статуса «оплачен»)

```

ALTER PROCEDURE [dbo].[CreateNewCheque]
    @id_cashier as int,
    @id_reg_lent as bigint,
    @id_cash_register as int
AS
BEGIN
    If(@id_cashier in (SELECT id_cashier FROM Cashier)
        AND @id_reg_lent in (SELECT id_reg_lent FROM ELKZ_lent)
        AND @id_cash_register in (SELECT id_cash_register FROM Cash_register))
    BEGIN
        INSERT INTO Cheque (id_cashier, id_reg_lent, id_cash_register)
        VALUES (@id_cashier, @id_reg_lent, @id_cash_register)
        RETURN 3
    END
    ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.51. Процедура создания нового чека в таблице Cheque

```

USE Продажа_товаров
GO
CREATE VIEW AllBestCashier
as
WITH First AS
(SELECT DISTINCT id_fisc_check, id_cashier, result, left(convert(varchar, date_time,
120),7) Date FROM Cheque),
Third as(SELECT F.id_cashier,
          name_cashier,
          fam_cashier,
          otchestvi_cashier,
          Date,
          name_company,
          max(sum(result)) over (partition by F.id_cashier, Date) Res
FROM First F Join [Product-Cheque] PC ON F.id_fisc_check =
PC.id_fisc_check
          JOIN Product P ON P.id_product = PC.id_product
          JOIN Cashier Cash ON Cash.id_cashier = F.id_cashier
          JOIN Company Com ON Com.id_company =
Cash.id_company
          GROUP by F.id_cashier, name_cashier, fam_cashier, otchestvi_cashier, Date,
name_company),
Forty AS( SELECT DISTINCT name_company, Date, max(Res)Res FROM Third group by
Date , name_company)

SELECT DISTINCT id_cashier, name_cashier, fam_cashier, F.name_company, F.Date,
F.Res
FROM Third T JOIN Forty F ON T.Res = F.Res

```

Рис. 2.52. Представление в котором выводятся все сотрудники месяца в каждом магазине за все время

```

]ALTER PROCEDURE [dbo].[PopularInKategoryInDate]
    @name_company AS nvarchar(50),
    @name_kategory AS nvarchar(50),
    @dateStart AS date,
    @dateStop AS date
AS
]BEGIN
]IF (@name_company in (SELECT name_company FROM Company) AND @dateStart < @dateStop )
]BEGIN
]SELECT TOP 1 name_product,
    sum(count_product) over (partition by PC.id_product) count_prod_kat,
    name_kategory,
    sum(price_product) over (partition by PC.id_product) sum_price_kat,
    name_company
FROM Cashier Cash JOIN Cheque Ch ON Cash.id_cashier = Ch.id_cashier
JOIN [Product-Cheque] PC ON Ch.id_fisc_check = PC.id_fisc_check
JOIN Product Prod ON Prod.id_product = PC.id_product
JOIN Kategory Kat ON Prod.id_kategory = Kat.id_kategory
JOIN Company Com ON Com.id_company = Cash.id_company

WHERE name_company = @name_company AND
name_kategory = @name_kategory AND
CAST(date_time as date) > @dateStart AND
CAST(date_time as date) < @dateStop

ORDER BY count_prod_kat DESC
RETURN 4
END
ELSE
] BEGIN
    RETURN 0
END
END

```

Рис. 2.53. Процедура нахождения самого продаваемого товара в выбранной категории у выбранной компании за определенный период

```

ALTER PROCEDURE [dbo].[AllChequeBuyer]
    @id_buyer AS int
AS
BEGIN
    If(@id_buyer in (SELECT id_buyer FROM Buyer))
    BEGIN
        SELECT Com.name_company,
               Com.address_company,
               Reg.name_cash_register,
               Com.INN,
               Com.reg_nom,
               B.id_buyer,
               Ch.id_fisc_check,
               Ch.date_time,
               Ch.count_product_strings,
               Ch.subtotal,
               cast(B.discount as varchar) + '%' discount,
               Ch.result,
               PS.payment_status,
               Pay.type_payment,
               Ch.entered,
               Ch.change,
               Ch.count_product_strings,
               Work.days_work,
               Work.time_start,
               Work.time_stop,
               E.id_reg_lent,
               E.num_KPK,
               E.value_KPK,
               Ch.date_time
        FROM Cashier Cash JOIN Cheque Ch ON Cash.id_cashier = Ch.id_cashier
        JOIN Company Com ON Com.id_company = Cash.id_company
        JOIN Work_time Work ON Work.id_work_time = Com.id_work_time
        JOIN Buyer B ON B.id_buyer = Ch.id_buyer
        JOIN ELKZ_lent E ON E.id_reg_lent = Ch.id_reg_lent
        JOIN Payment Pay ON Pay.id_payment = Ch.id_payment
        JOIN Payment_status PS ON PS.id_status = Ch.id_status
        JOIN Cash_register Reg ON Reg.id_cash_register = Ch.id_cash_register
        WHERE B.id_buyer = @id_buyer

        SELECT PC.id_fisc_check,
               PC.id_product,
               PC.count_product,
               Un.unit_of_measurement,
               Prod.name_product,
               Prod.count_product_on_warehouse,
               Prod.price_product,
               Kat.name_kategory
        FROM [Product-Cheque] PC JOIN Product Prod ON Prod.id_product = PC.id_product
        JOIN Kategory Kat ON Kat.id_kategory = Prod.id_kategory
        JOIN Unit_of_measurement Un ON Un.id_unit_of_measurement = Prod.id_unit_of_measurement
        WHERE PC.id_fisc_check in (SELECT id_fisc_check FROM Cheque WHERE id_buyer = @id_buyer)

        RETURN 4
    END
    ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.54. Процедура вывода всех чеков для определенного покупателя

```

ALTER PROCEDURE [dbo].[CountProductSell]
    @name_company AS nvarchar(50),
    @date_start AS date
AS
BEGIN
    IF (@name_company in (SELECT name_company FROM Company))
    BEGIN
        DECLARE @date_end as datetime
        SELECT @date_end = (SELECT DateAdd(DAY, 14, @date_start))

        SELECT name_product,
            sum(count_product) over (partition by Prod.id_product) sum_prod,
            sum(price_product) over (partition by Prod.id_product) sum_price_prod
        FROM Cashier Cash JOIN Cheque Ch ON Cash.id_cashier = Ch.id_cashier
        JOIN [Product-Cheque] PC ON Ch.id_fisc_check = PC.id_fisc_check
        JOIN Product Prod ON Prod.id_product = PC.id_product
        JOIN Company Com ON Com.id_company = Cash.id_company
        WHERE name_company = @name_company AND CAST(date_time AS date) > @date_start AND date_time < @date_end
        RETURN 4
    END
    ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.55. Процедура нахождения количества и суммы (на которую продали товара) для каждого товара за 2 недели с выбранной даты у определенной компании



```

ALTER PROCEDURE [dbo].[AllProductsForPeriod]
    @name_company AS nvarchar(50),
    @dateStart AS date,
    @dateStop AS date
AS
BEGIN
IF (@name_company in (SELECT name_company FROM Company))
BEGIN
    SELECT DISTINCT
        name_category,
        name_product,
        sum(count_product) over (partition by name_company, name_product) count_buy,
        unit_of_measurement,
        sum(price_product) over (partition by name_company, name_product) sum_buy
    FROM Cheque Ch
    JOIN [Product-Cheque] PC ON Ch.id_fisc_check = PC.id_fisc_check
    JOIN Product Prod ON Prod.id_product = PC.id_product
    JOIN Unit_of_measurement Unit ON Unit.id_unit_of_measurement = Prod.id_unit_of_measurement
    JOIN Kotegory Kat ON Kat.id_kategory = Prod.id_kategory
    JOIN Cashier Cash ON Cash.id_cashier = Ch.id_cashier
    JOIN Company Com ON Com.id_company = Cash.id_company
    WHERE name_company = @name_company AND CAST(date_time as date) > @dateStart AND CAST(date_time as date) < @dateStop
    RETURN 4
END
ELSE
BEGIN
    RETURN 0
END
END

```

Рис. 2.56. Процедура, которая выводит все продукты с их количеством и суммой, проданных определенной компанией за определенный промежуток времени

```

ALTER PROCEDURE [dbo].[BestCashierOnMounth]
    @name_company AS nvarchar(50),
    @start_dat AS date,
    @stop_dat as date
AS
BEGIN
IF (@name_company in (SELECT name_company FROM Company) AND @start_dat < @stop_dat )
BEGIN
    SELECT * FROM AllBestCashier
    WHERE Date >= left(convert(varchar, @start_dat, 120), 7)
        AND Date <= left(convert(varchar, @stop_dat, 120), 7)
        AND name_company = @name_company

    RETURN 4
END
ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.57. Процедура нахождения работников месяца данной компании, за данный период по месяцам (работник месяца - тот кто продал товара на большую сумму)

```

ALTER PROCEDURE [dbo].[AVGoneCheque]
    @name_company AS nvarchar(50),
    @start_dat AS date,
    @stop_dat as date
AS
BEGIN
IF (@name_company in (SELECT name_company FROM Company) AND @start_dat < @stop_dat )
BEGIN
    SELECT name_company, Date, ROUND(AVG(count_product_strings), 2) avg_count_strings,
        ROUND(AVG(result), 2) avg_result, ROUND(AVG(count_product), 2) avg_count_prod
    FROM Cheque Ch JOIN Cashier Cash ON Ch.id_cashier = Cash.id_cashier
        JOIN [Product-Cheque] PC ON Ch.id_fisc_check = PC.id_fisc_check
        JOIN Company Com ON Com.id_company = Cash.id_company
        JOIN (SELECT DISTINCT id_fisc_check, left(convert(varchar, date_time, 120),7) Date FROM Cheque) CC
        ON CC.id_fisc_check = Ch.id_fisc_check
    WHERE Date >= left(convert(varchar, @start_dat, 120), 7)
        AND Date <= left(convert(varchar, @stop_dat, 120), 7)
        AND name_company = @name_company
    GROUP BY name_company, Date
    RETURN 4
END
ELSE
    BEGIN
        RETURN 0
    END
END

```

Рис. 2.58. Процедура, которая выводит среднее значение вычисляемых полей в чеке для выбранной компании и за определенный период по месяцам

## 2.3. Реализация системы безопасности

Цель - реализовать систему безопасности.

Задача: в соответствии с результатами анализа предметной области настроить права доступа пользователей разных категорий к объектам БД.

Сначала были созданы пользователи с логином и паролем и правами доступа (рис. 2.59, 2.60). Далее было проверено наличие пользователей и их прав (рис. 2.61, 2.62).

Таким образом, на данном этапе была реализована системы безопасности, цель и задача данного этапа были достигнуты.

```
CREATE LOGIN Cashier WITH PASSWORD = '12345', DEFAULT_DATABASE = Продажа_товаров;  
  
CREATE USER Cashier FOR LOGIN Cashier;  
  
GRANT EXECUTE ON CreateNewCheque TO Cashier;  
GRANT EXECUTE ON AddChequeByer TO Cashier;  
GRANT EXECUTE ON AddChequeProducts TO Cashier;  
GRANT EXECUTE ON AddChequePayment TO Cashier;  
GRANT EXECUTE ON AddChequeEnteredCash TO Cashier;  
GRANT EXECUTE ON DelProductFromCheque TO Cashier;  
GRANT EXECUTE ON DelitedCheque TO Cashier;
```

Рис. 2.59. Создание пользователя Cashier

```
CREATE LOGIN Buyer WITH PASSWORD = '12345', DEFAULT_DATABASE = Продажа_товаров;  
  
CREATE USER Buyer FOR LOGIN Buyer;  
  
GRANT EXECUTE ON AllChequeBuyer TO Cashier;
```

Рис. 2.60. Создание пользователя Buyer

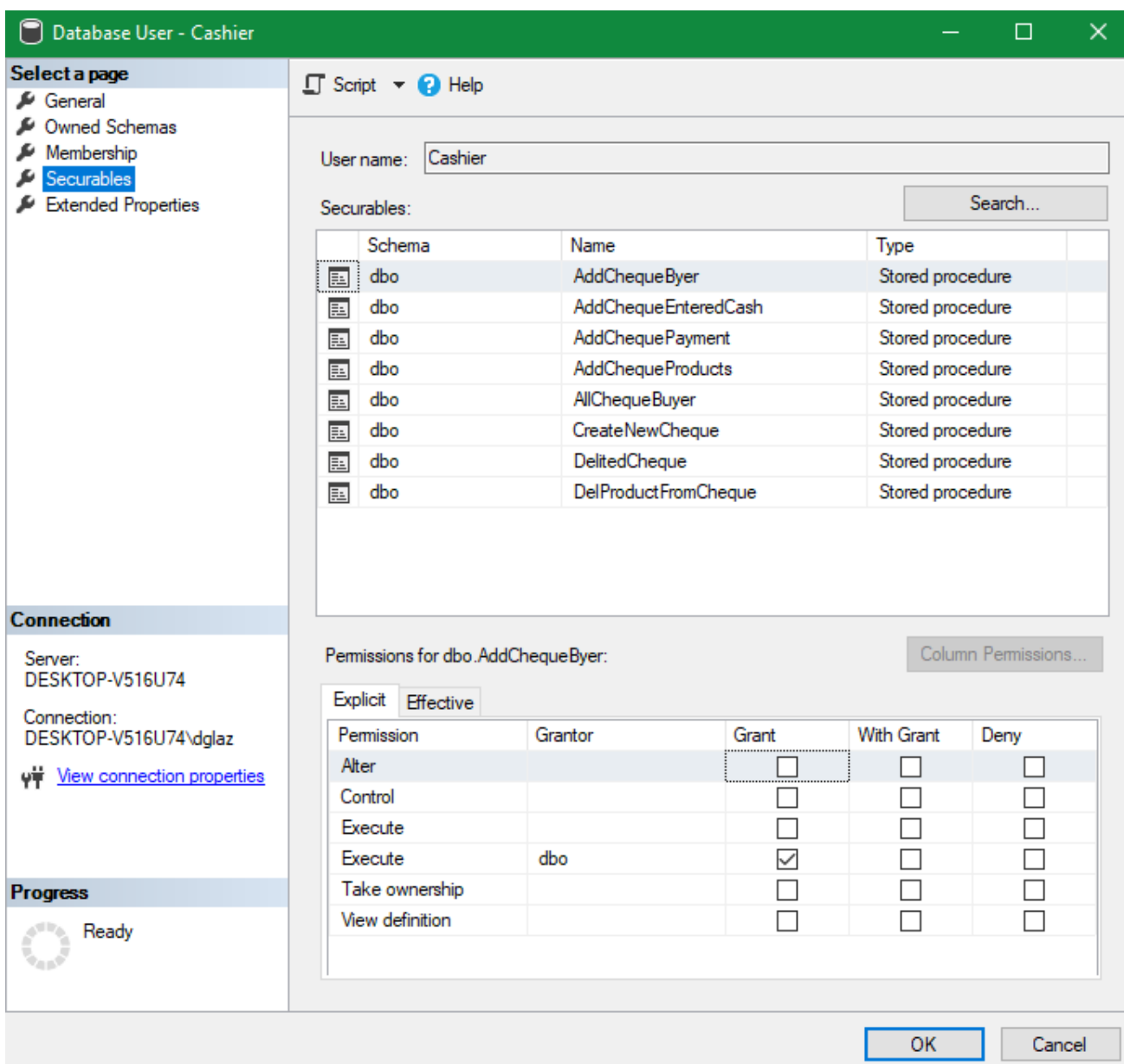


Рис. 2.61. Права пользователя Cashier

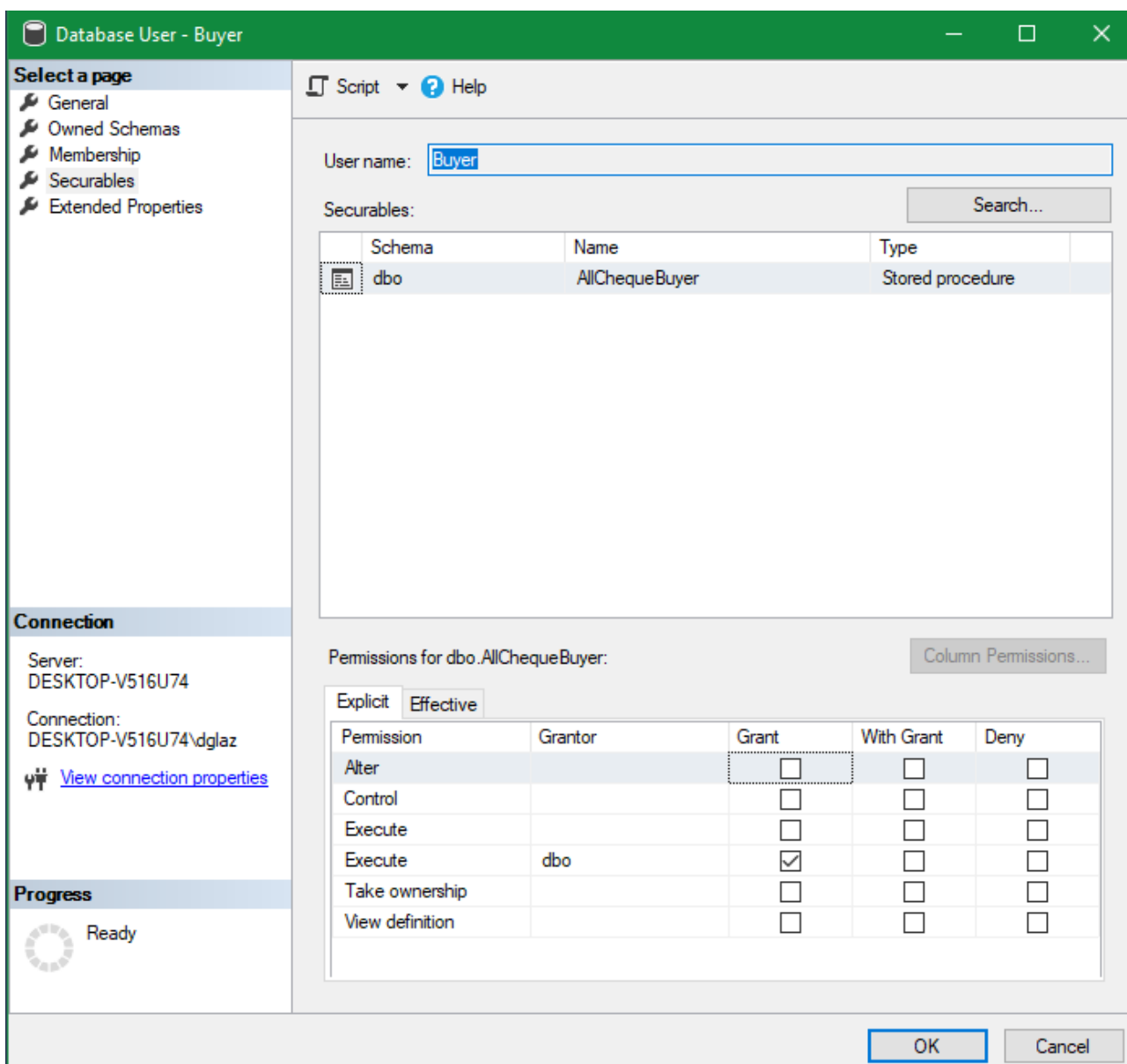


Рис. 2.62. Права пользователя Buyer

### 3. Реализация приложения для базы данных

Цель - реализовать приложения для базы данных.

Было реализовано 2 вкладки:

1. Вкладка с функциями добавления/изменения/удаления (рис. 3.1);
2. Вкладка с вызовом хранимой процедуры с аналитическим запросом (рис. 3.2).

Таким образом, на данном этапе было реализовано приложение для работы с базой данных, цель и задача данного этапа были достигнуты.

id_fisc_check	subtotal	result	date_time	entered	change	count_product_strir	id_buyer	id_cash_register	id_cashier	id_reg_lent
10001	54299,9	48869,91	27.12.2022 22:33	62000	13130,09	4	200011	1	3	1000000000
30001	66119	66119	27.12.2022 22:33	70000	3881	3	200011	2	5	1000000005
30003	142880	121448	27.12.2022 22:33	121448	0	2	200011	3	11	1000000005
30004	30760,89	24608,71	27.12.2022 22:33	24608,71	0	3	200011	3	11	1000000008
30005	8110	7299	27.12.2022 22:34	7300	1	1	200011	5	11	1000000008
30006	868,71	868,71	27.12.2022 22:34	868,71	0	5	200011	1	8	1000000001
30008	670,68	603,61	27.12.2022 22:34	603,61	0	5	200011	1	7	1000000001
30009	499,38	449,44	27.12.2022 22:34	600	150,56	4	200011	1	9	1000000002
30011	2731,05	2731,05	27.12.2022 22:34	3000	268,95	4	200011	4	7	1000000002
30012	52990	52990	27.12.2022 22:34	52990	0	1	200011	9	10	1000000004

Рис. 3.1. Вкладка №1

tabPage1 tabPage2

000 "Citilink" ▾

Вывести средние значения по чеку

First date

1 декабря 2022 г. ▾

Second date

29 декабря 2022 г. ▾

	name_company	Date	avg_count_strings	avg_result	avg_count_prod
▶	000 "Citilink"	2022-12	2	47548,96	1,35
*					

Рис. 3.2. Вкладка №2

# Заключение

Достоинства проекта:

1. Отличная теоретическая часть
2. Хорошая реализация самой базы данных
3. Хорошая реализация объектов базы данных
4. Автоматизированный процесс заполнения большинства полей в чеке

Недостатки:

1. Недоделанность приложения
2. Система безопасности не участвует в работе приложения

В данной курсовой работе использовались такие программные ресурсы как:

1. Visual studio 2019;
2. Microsoft SQL Server 2018;
3. DrawIo;
4. Google Drive.



## Список литературы

1. Самые частые вопросы и ошибки, возникающие при работе с чеками [Электронный ресурс] – Режим доступа:  
[https://www.atol.ru/blog/samye-chastye-voprosy-i-oshibki-voznikayushchie-pri-rabote-s-chekami/?utm\\_source=yandex.ru&utm\\_medium=organic&utm\\_campaign=yandex.ru&utm\\_referrer=yandex.ru](https://www.atol.ru/blog/samye-chastye-voprosy-i-oshibki-voznikayushchie-pri-rabote-s-chekami/?utm_source=yandex.ru&utm_medium=organic&utm_campaign=yandex.ru&utm_referrer=yandex.ru)
2. Статья Типы данных (Transact-SQL) [Электронный ресурс] – Режим доступа:  
[HTTPS://LEARN.MICROSOFT.COM/RU-RU/SQL/T-SQL/DATA-TYPES/DATA-TYPES-TRANSACT-SQL?TOC=/azure%2Fsynapse-analytics%2Fsql-data-warehouse%2Ftoc.json&bc=/azure%2Fsynapse-analytics%2Fsql-data-warehouse%2Fbreadcrumb%2Ftoc.json&view=azure-sqldw-latest&preserve-view=true](https://learn.microsoft.com/ru-ru/sql/t-sql/data-types/data-types-transact-sql?toc=/azure%2Fsynapse-analytics%2Fsql-data-warehouse%2Ftoc.json&bc=/azure%2Fsynapse-analytics%2Fsql-data-warehouse%2Fbreadcrumb%2Ftoc.json&view=azure-sqldw-latest&preserve-view=true)
3. ОБЕСПЕЧЕНИЕ ЦЕЛОСТНОСТИ И БЕЗОПАСНОСТИ БД [Электронный ресурс] – Режим доступа: [HTTPS://STUDFILE.NET/PREVIEW/3827056/PAGE:2/](https://studfile.net/preview/3827056/page:2/)
4. Статья генератор регистрационного номера ккт [Электронный ресурс] – Режим доступа:  
<https://teltaxi.ru/spravochnik/generator-registratsionnogo-nomera-kkt>