МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Костромской государственный университет»

(КГУ)

Институт автоматизированных систем и технологий

Информационные системы и технологии

Направление подготовки 09.03.02

Информационные системы и технологии

Пояснительная записка

Тема: “Автосервис”

Выполнил студент:

Куделин Роман

Группа 17-ИСбо-2б

Проверил: Прядкина Н.О.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_

Кострома

2020

**ОГЛАВЛЕНИЕ**

[**1. ЦЕЛЬ**](#_ssobbzr5jto7) **2**

[**2. ВЫБОР ИНСТРУМЕНТАРИЯ**](#_9wlzi8bd1eug) **3**

[2.1 Требования SQL Server 2014 к оборудованию](#_ly4t7po5ifxl) 3

[2.2 Требования SQL Server 2014 к ПО](#_gj29i3tyo98z) 3

[**3. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**](#_f9orfrn5yb36) **4**

[3.1 Категории данных](#_nhjsmni68mus) 4

[3.2 Категории пользователей](#_4r5oxcv2p1y7) 6

[3.3 Бизнес правила и ограничения](#_z50qeujev9qj) 7

[**4. ПОСТРОЕНИЕ ER- ДИАГРАММЫ**](#_7a865vgi2sph) **8**

[4.1 Обозначение](#_ese4j9iorbg) 8

[**5. ПОСТРОЕНИЕ РЕЛЯЦИОННОЙ СХЕМЫ**](#_qxy2wzobownv) **9**

[**5.1 Реляционная схема**](#_wnih6tz0q38a) **10**

[**6. РЕАЛИЗАЦИЯ ОБЪЕКТОВ БАЗЫ ДАННЫХ**](#_vj8z6am49c6f) **10**

[**7. РЕАЛИЗАЦИЯ СИСТЕМЫ БЕЗОПАСНОСТИ**](#_91aq1uxpdngc) **13**

[**8. ЗАКЛЮЧЕНИЕ**](#_1u4xsk9dxzsj) **13**

[**9. СПИСОК ЛИТЕРАТУРЫ**](#_671ztr6rpfhp) **14**

[Приложение](#_ieq343493bfo) 1. [Скрипты создания](#_gjx715hygl07) представлений 15

[Приложение 2](#_a3vlq1p366ue). Скрипты создания хранимых процедур 18

[Приложение 3](#_ps6du8dje7qu). Скрипты создания триггеров 24

# 1. ЦЕЛЬ

Выбрать предметную область и изучить ее. Провести анализ предметной области, выделить категории данных и категории пользователей. Построить ER- диаграмму. На основе ER-диаграммы построить реляционную схему.

Реализовать базу данных, выбранной предметной области на сервере SQL. Реализовать ограничение целостности. Выполнить создание представлений, хранимых процедур и триггеров. Создать пользователей системы и настроить права доступа к объектам.

# 2. **В**ЫБОР ИНСТРУМЕНТАРИЯ

**Microsoft SQL Server** — система управления реляционными базами данных, разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия.

Программа для конфигурирования и управления Microsoft SQL Server: sql server management studio 2014

## 2.1 Требования SQL Server 2014 к оборудованию

Версия SQL Server 2014 предназначена для запуска на разнообразных компьютерных системах от ноутбуков и настольных компьютеров до серверов корпоративного класса. Минимальные требования к процессору — 32-разрядный процессор типа x86 с тактовой частотой 1,0 ГГц, но рекомендуется 64-разрядная модель x64 с тактовой частотой не менее 1,4 ГГц. Специалисты Microsoft рекомендуют быстродействие 2,0 ГГц.

Microsoft рекомендует использовать не менее 4 Гбайт оперативной памяти для SQL Server 2014 Standard и более старших редакций. Требования к пространству на диске зависят от устанавливаемых компонентов. Для SQL Server 2014 необходимо, по крайней мере, 6 Гбайт свободного пространства на диске.

## **2.2 Требования SQL Server 2014 к ПО**

SQL Server 2014 предъявляет ряд требований к программному обеспечению, помимо базовой операционной системы. Перечислим основные из этих требований.

* Если выбраны компоненты Database Engine, Reporting Services, Master Data Services, Data Quality Services, Replication или SSMS, то требуется платформа. NET Framework 3.5 SP1. Она не устанавливается в процессе установки SQL Server.
* . NET Framework 4.0. Устанавливается в процессе инсталляции SQL Server.
* PowerShell 2.0 или более новая версия необходима, если установлены компонент Database Engine или SSMS.

# 3. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

На данном этапе были выделены:

1. Категории данных
2. Их сведения
3. Общий объем данных
4. Рост данных
5. Категории пользователей
6. Задачи, которые выполняют категории пользователей
7. Бизнес правила и ограничения целостности

## **3.1 Категории данных**

Таблица 1

Категории данных

|  |  |  |  |
| --- | --- | --- | --- |
| **Категория** | **Сведения** | **Общий объем данных** | **Рост данных** |
| Услуга | **Код услуги**  Наименование  Гарантия  Цена | ~500 | 1% в год |
| Сотрудники | **Код сотрудника**  ФИО  Должность  Адрес  Номер телефона  Паспорт | 7 | +- 1 сотрудник в полгода |
| Клиент | **Код клиента**  ФИО  Номер телефона  Размер скидки  Паспорт | Несколько тысяч | ~5-15 клиентов в день |
| Должность | **Код должности**  Название | 5 | нет |
| Тип оплаты | **Код оплаты**  Наименование | 2 | нет |
| Запчасти | **Код запчасти**  Наименование  Производитель  Дата поставки  Количество  Гарантия  Цена | Несколько тысяч | ~30% в год |
| Производитель | **ID**  Страна производителя  Бренд | Несколько сотен | +-10% в год |
| Марка | **ID**  Наименование  Модель | 100-150 | ~10% в год |
| Модель | **ID**  Наименование | Несколько сотен | ~20% в год |
| Автомобиль | **Код автомобиля**  Марка  Тип двигателя  Объем двигателя  Мощность двигателя  Тип кузова  VIN-код  Гос. номер | Несколько тысяч | ~10% в год |
| Тип двигателя | **ID**  Наименование | 3 | нет |
| Заказ запчастей | **Код заказа**  Запчасти  Количество  Заказы | Несколько тысяч | ~~5-15 в день |
| Заказ услуг | **Код заказа**  Услуга  Сотрудник  Количество  Заказы | Несколько тысяч | ~5-15 в день |
| Заказы | **Код заказа**  Автомобиль  Пробег  Тип оплаты  Дата заказа  Сумма | Несколько тысяч | ~3-10 в день |
| Тип кузова | **Код кузова**  Наименование | 22 | нет |

## **3.2 Категории пользователей**

Таблица 2

Категории пользователей

|  |  |  |  |
| --- | --- | --- | --- |
| **Категории пользователей** | **Задачи** | **Кол-во людей** | **Тенденции роста** |
| Директор | Администрирование. Прием/увольнение сотрудников. | 1 | нет |
| Администратор | Администрирование.  Прием поставок запчастей. Прием заказа услуг. Учет. | 1 | нет |
| Автослесарь | Просмотр заказа услуг | 2 | ~+- 1 в полгода |
| Автомеханик | Просмотр заказа услуг | 2 | ~+- 1 в полгода |
| Стажер | Просмотр заказа услуг | 1 | ~+- 1 в полгода |

## **3.3 Бизнес правила и ограничения**

Декларативные ограничения целостности:

1. Ограничения целостности атрибута:
   1. Размер скидки клиентам устанавливается в индивидуальном порядке (от 0 и до 30%)
   2. Поле “Скидка” принимает значение от 0 до 30
   3. Поля “Гарантия”, “Цена” и “Скидка” содержат только целые числа.
   4. Поле “Телефон” может быть только 11 символьным и содержать только целые цифры.
   5. Поле “Паспорт” может быть только 10-символьным и содержать только целые цифры.
   6. Поле “Паспорт” должно быть уникальным.
   7. Поле “Телефон” должно быть уникальным.
   8. Поле “VIN-код” должно быть уникальным.
   9. Поле “Гос номер” должно быть уникальным.
   10. Поле “VIN-код” равен 17 символам. Основан на стандартах ISO 3779-1983 и ISO 3780.
   11. Гос номер может быть только 8 либо 9 символьным, содержать цифры и буквы “АВЕКМНОРСТУХ” Вид: “А111АА11”

Процедурные ограничения целостности:

1. Поле “Сумма” в таблице “Заказы” должно автоматически пересчитываться при добавлении или изменении записей в таблицах “Заказ услуг” и “Заказ запчастей”
2. Сотрудник может занимать только одну должность.
3. Поле “Количество” в таблице “Запчасти” должно автоматически уменьшаться при добавлении новой записи “Заказ запчастей”

# 4. ПОСТРОЕНИЕ ER- ДИАГРАММЫ

В соответствии с категориями данных была построена ER-диаграмма. Категории являются сущностями на диаграмме, а сведения - атрибутами.



*Рис.1 ER-диаграмма*

## **4.1 Обозначение**

Таблица 3

Обозначение объектов в ER- диаграмме

|  |  |
| --- | --- |
| Символ | Характеризует сущность, а также отношения между двумя или более элементами. |
| Символ производного атрибута | Атрибут, чье значение можно вычислить. |
| Символ | Отношение между сущностями. |
|  | Сущность |

# 5. ПОСТРОЕНИЕ РЕЛЯЦИОННОЙ СХЕМЫ

На данном этапе была создана реляционная схема базы данных на основе ER-диаграммы.

# 5.1 Реляционная схема

Тип кузова (**Код кузова**, Наименование)

Тип двигателя (**ID**, Наименование)

Марка (**ID**, **ID Модели,** Наименование)

Модель (**ID**, Наименование)

Услуга (**Код услуги**, Наименование, Цена, Гарантия)

Клиент (**Код клиента**, ФИО, Размер скидки, Номер телефона)

Производитель (**Код производителя**, Страна, Бренд)

Запчасти (**Код запчасти, ID Производителя,** Наименование, Количество, Цена, Гарантия, Дата Поставки)

Должность (**Код должности**, Наименование)

Тип оплаты (**Код оплаты**, Наименование)

Автомобиль (**Код авто, Код кузова, ID двигателя**, **ID марки, ID модели,** Объем двигателя, Мощность двигателя, VIN-код, Гос. номер )

Сотрудники (**Код сотрудника, Код должности**, ФИО, Адрес, Номер телефона, Адрес)

Заказы (**Код заказа, Код авто, Код клиента, Код оплаты**, Пробег, Дата оплаты, Сумма)

Заказ услуг (**Код заказа услуг, Код услуги, Код заказа, Код сотрудника,** Количество)

Заказ запчастей (**Код заказа запчастей, Код заказа, Код запчасти**)

# 6. РЕАЛИЗАЦИЯ ОБЪЕКТОВ БАЗЫ ДАННЫХ

Были созданы таблицы, указанные в ER-диаграмме. Так же были созданы представления, триггеры, хранимые процедуры и ограничения целостности. Все созданные объекты можно посмотреть в приложении. Далее приведено описание созданных таблиц (Имя столбца: тип данных).

Таблица Automobils содержит поля:

1. id: int
2. engine\_capitacity: real,
3. engine\_power: int
4. mark: int
5. body: int
6. engine: int
7. VIN\_code: nvarchar
8. State\_number: nvarchar

Таблица Client содержит поля:

1. id:int
2. surname: nvarchar
3. name: nvarchar
4. patronymic: nvarchar
5. discount: int
6. phone\_number: nvarchar
7. passport: nvarchar

Таблица Collaborators содержит поля:

1. id: int
2. surname: nvarchar
3. name: nvarchar
4. patronymic: nvarchar
5. address: nvarchar
6. passport: varchar
7. post: int
8. phone\_number: varchar

Таблица Manufacturer содержит поля

1. id: int
2. country: nvarchar
3. brand: nvarchar

Таблица Marks содержит поля

1. id: int
2. name: nvarchar
3. model: int

Таблица Models содержит поля

1. id: int
2. name: nvarchar

Таблица Oerdering\_spares содержит поля

1. id: int
2. number: int
3. spares: int
4. orderr: int

Таблица Ordering\_servuces содержит поля

1. id: int
2. number: int
3. collaborator: int
4. service: int
5. orderr: int

Таблица Orderr содержит поля

1. id: int
2. mileage: int
3. date\_order: datetime
4. sum: money
5. type\_payment: int
6. automobile: int
7. client: int

Таблица Post содержит поля

1. id: int
2. name: nvarchar

Таблица Service содержит поля

1. id: int
2. name: nvarchar
3. price: money
4. guaratee: int

Таблица Spares содержит поля

1. id: int
2. name: nvarchar
3. manufactur: int
4. number: int
5. date\_of\_delivery: datetime
6. price: money
7. guarantee: int

Таблица Type\_body содержит поля

1. id: int
2. name: nvarchar

Таблица Type\_engines содержит поля

1. id: int
2. name: nvarchar

Таблица Type\_payment содержит поля

1. id: int
2. name: nvarchar

# 7. РЕАЛИЗАЦИЯ СИСТЕМЫ БЕЗОПАСНОСТИ

Для обеспечения безопасности базы данных были созданы следующие 3 пользователя:

1. Hozain - данный пользователь имеет разрешения на добавление, редактирование и удаление записей во всех таблицах а также имеет доступ ко всем представлениям и хранимым процедурам.
2. Administrator - данный пользователь имеет разрешения на добавление, редактирование и удаление записей во многих таблицах. Подробнее можно посмотреть в приложенном документе “Скрипт базы данных”
3. Automehannik - у данного пользователя есть разрешение только на чтение преставления “Заказ услуг Automih”, в котором отображены данные о заказанных услугах содержащие сведения о автомобиле.

# 8. ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта, я применил навыки приобретенные за прошлый семестр обучения в рамках курса “Управление данными”. В курсовом проекте была реализована база данных для выбранной предметной области. Были созданы ограничения целостности некоторых полей таблиц, представления некоторых таблиц, хранимые процедуры, аналитические запросы в виде хранимых процедур с параметрами, а также триггеры.

# 9. СПИСОК ЛИТЕРАТУРЫ

ERD: определение и обзор | Lucidchart [Электронный ресурс] / Lucidchart. – Режим доступа : <https://www.lucidchart.com/pages/ru> (дата обращения: 11.10.2020).

Проектирование реляционной базы данных: Метод. указания к домашнему заданию по курсу "Базы данных" / Московский институт электроники и математики им. А.Н. Тихонова НИУ ВШЭ; Сост.: И.П. Карпова. – Режим доступа : <https://publications.hse.ru/mirror/pubs/share//direct/212747315> (дата обращения: 11.10.2020).

SQL - справочник, примеры запросов, структура базы данных [Электронный ресурс] – Режим доступа : https://unetway.com/tutorials/sql (дата обращения: 28.10.2020).

Права в MS SQL и создание пользователей.[Электронный ресурс] – Режим доступа : https://fixmypc.ru/post/prava-v-ms-sql-i-sozdanie-polzovatelei-create-user-grant-revoke/ (дата обращения: 03.11.2020).

## Приложение 1

## Скрипты создания представлений

**Представление “Автомобили”**

CREATE VIEW [dbo].[Автомобили] AS

SELECT engine\_capitacity as "Объем двигателя л", engine\_power as "Мощьность двигателя лс", Marks.name as "Марка", Models.name as "Модель", Type\_body.name as "Кузов", Type\_engines.name as "Тип двигателя", VIN\_code as "VIN-код", State\_number as "Гос. номер"

FROM Automobils

Inner join Marks on Marks.id = Automobils.mark

Inner join Models on Models.id = Marks.model

Inner join Type\_body on Type\_body.id = Automobils.body

Inner join Type\_engines on Type\_engines.id = Automobils.engine

**Представление “Заказ запчастей”**

CREATE VIEW [dbo].[Заказ запчастей] AS

SELECT orderr as "№ заказа", Spares.name as "Наименование", Oerdering\_spares.number as "Количество", Manufacturer.country as "Страна изготовитель", Manufacturer.brand as "Бренд"

FROM Oerdering\_spares

Inner join Spares on Spares.id = Oerdering\_spares.spares

Inner join Manufacturer on Manufacturer.id = Spares.manufactur

**Представление “Заказ услуг”**

CREATE VIEW [dbo].[Заказ услуг] AS

SELECT orderr as "№ заказа", Service.name as "Услуга", number as "Количество", CONCAT(Collaborators.surname, ' ', Collaborators.name, ' ', Collaborators.patronymic) as "Исполнитель", Post.name as "Должность"

FROM Ordering\_servuces

Inner join Service on Service.id = Ordering\_servuces.service

Inner join Collaborators on Collaborators.id = Ordering\_servuces.collaborator

Inner join Post on Post.id = Collaborators.post

**Представление “Заказ услуг Automih”**

CREATE VIEW [dbo].[Заказ услуг Automih] AS

SELECT orderr as "№ заказа",CONCAT(Marks.name, ' ', Models.name, ' ', Automobils.engine\_capitacity, 'л ', Type\_engines.name, ' ', Automobils.State\_number) as "Автомобиль", Service.name as "Услуга", number as "Количество", CONCAT(Collaborators.surname, ' ', Collaborators.name, ' ', Collaborators.patronymic) as "Исполнитель", Post.name as "Должность"

FROM Ordering\_servuces

Inner join Service on Service.id = Ordering\_servuces.service

Inner join Collaborators on Collaborators.id = Ordering\_servuces.collaborator

Inner join Post on Post.id = Collaborators.post

Inner join Orderr on Orderr.id = Ordering\_servuces.orderr

Inner join Automobils on Automobils.id = Orderr.automobile

Inner join Type\_engines on Type\_engines.id = Automobils.engine

Inner join Marks on Marks.id = Automobils.mark

Inner join Models on Models.id = Marks.model

**Представление “Заказы”**

CREATE VIEW [dbo].[Заказы] AS

SELECT Orderr.id as "№ заказа", CONCAT(Marks.name, ' ', Models.name, ' ', Automobils.engine\_capitacity, 'л ', Type\_engines.name, ' ', Automobils.State\_number) as "Автомобиль", CONCAT(Client.surname, ' ', Client.name, ' ', Client.patronymic) as "Клиент",

Orderr.date\_order as "Дата обращения", Orderr.mileage as "Пробег", Client.discount as "Скидка %", Type\_payment.name as "Тип оплаты", Orderr.sum as "Сумма к оплате"

FROM Orderr

Inner join Automobils on Automobils.id = Orderr.automobile

Inner join Type\_engines on Type\_engines.id = Automobils.engine

Inner join Marks on Marks.id = Automobils.mark

Inner join Models on Models.id = Marks.model

Inner join Client on Client.id = Orderr.client

Inner join Type\_payment on Type\_payment.id = Orderr.type\_payment

**Представление “Марка-Модель”**

CREATE VIEW [dbo].[Марка-Модель] AS

SELECT Marks.name as "Марка", Models.name as "Модель"

FROM Marks

Inner join Models on Models.id = Marks.model

**Представление “Сотрудники”**

CREATE VIEW [dbo].[Сотрудники] AS

SELECT surname as "Фамилия", [Collaborators].name as "Имя", patronymic as "Отчество", phone\_number as "Номер телефона", Post.name as "Должность"

FROM Collaborators

Inner join Post on Post.id = [Collaborators].post

## Приложение 2

## Скрипты создания хранимых процедур

**Процедура “info\_brand\_spares”**

create procedure [dbo].[info\_brand\_spares](@brand nvarchar(50)) as

begin

select Spares.id, Spares.name, Spares.number, Spares.price, Spares.date\_of\_delivery, Spares.guarantee, Manufacturer.brand

from Spares

Inner join Manufacturer on Manufacturer.id = Spares.manufactur

where Manufacturer.brand = @brand;

end

**Процедура “info\_marks”**

create procedure [dbo].[info\_marks](@marka nvarchar(50)) as

begin

select Marks.id, Marks.name, Models.name

from Marks

Inner join Models on Models.id = Marks.model

where Marks.name = @marka;

end

**Процедура “info\_order\_date”**

create procedure [dbo].[info\_order\_date](@d1 date, @d2 date) as

begin

select Orderr.id, CONCAT(Client.surname, ' ', Client.name, ' ', Client.patronymic) as "Клиент",

CONCAT(Marks.name, ' ', Models.name, ' ', Automobils.engine\_capitacity, 'л ', Type\_engines.name, ' ', Automobils.State\_number) as "Автомобиль",

Orderr.mileage as "Пробег", Orderr.date\_order as "Дата заказа",

Orderr.type\_payment as "Тип платежа", Orderr.sum as "Сумма"

from Orderr

Inner join Automobils on Automobils.id = Orderr.automobile

Inner join Type\_engines on Type\_engines.id = Automobils.engine

Inner join Marks on Marks.id = Automobils.mark

Inner join Models on Models.id = Marks.model

Inner join Client on Client.id = Orderr.client

Inner join Type\_payment on Type\_payment.id = Orderr.type\_payment

where Orderr.date\_order between @d1 and @d2;

end

**Процедура “info\_service\_order\_by\_number”**

CREATE procedure [dbo].[info\_service\_order\_by\_number](@id\_order int) as

begin

select Ordering\_servuces.id as 'Артикул', Service.name as 'Наименование работы',

CONCAT(Collaborators.surname, ' ', Collaborators.name, ' ', Collaborators.patronymic) as 'Исполнитель',

Service.guaratee as 'Гарантия км', Service.price as 'Цена', Ordering\_servuces.number as 'Количество',

Client.discount as 'Скидка %',

(Service.price\* Ordering\_servuces.number\*Client.discount/100) as 'Сумма скидки Р',

(Service.price\* Ordering\_servuces.number\*(100-Client.discount)/100) as 'Сумма'

from Ordering\_servuces

inner join Service on Service.id = Ordering\_servuces.service

inner join Collaborators on Collaborators.id = Ordering\_servuces.collaborator

inner join Orderr on Orderr.id = Ordering\_servuces.orderr

inner join Client on Client.id = Orderr.client

where Ordering\_servuces.orderr = @id\_order

end

**Процедура “info\_spares\_count”**

create procedure [dbo].[info\_spares\_count](@count int) as

begin

select Spares.id, Spares.name as "Наименование", Spares.number as "Количесво", Spares.price as "Цена", Spares.date\_of\_delivery as "Дата поставки",

Spares.guarantee as "Гарантия", Manufacturer.brand as "Бренд", Manufacturer.country as "Страна"

from Spares

Inner join Manufacturer on Manufacturer.id = Spares.manufactur

where Spares.number <= @count;

end

**Процедура “info\_sparesAutiservis\_order\_by\_number”**

create procedure [dbo].[info\_sparesAutiservis\_order\_by\_number](@id\_order int) as

begin

select Spares.id as 'Артикул запчасти (материала)', Spares.name as 'Наименование запчасти (Материала)',

isnull(Spares.guarantee,0) as 'Гарантия, км', Spares.price as 'Цена',

Oerdering\_spares.number as 'Количество', Client.discount as 'Скидка %',

(Spares.price\* Oerdering\_spares.number\*Client.discount/100) as 'Сумма скидки Р',

(Spares.price\* Oerdering\_spares.number\*(100-Client.discount)/100) as 'Сумма Р'

from Oerdering\_spares

inner join Spares on Spares.id = Oerdering\_spares.spares

inner join Orderr on Orderr.id = Oerdering\_spares.orderr

inner join Client on Client.id = Orderr.client

where Oerdering\_spares.orderr = @id\_order and isnull(Spares.price,0) <> 0

end

**Процедура “info\_sparesClient\_order\_by\_number”**

CREATE procedure [dbo].[info\_sparesClient\_order\_by\_number](@id\_order int) as

begin

select Spares.id as 'Артикул запчасти (материала)', Spares.name as 'Наименование запчасти (Материала)',

Oerdering\_spares.number as 'Количество'

from Oerdering\_spares

inner join Spares on Spares.id = Oerdering\_spares.spares

where Oerdering\_spares.orderr = @id\_order and isnull(Spares.price,0) = 0 and isnull(Spares.guarantee, 0)=0

end

**Процедура “info\_SUM\_order\_date”**

CREATE procedure [dbo].[info\_SUM\_order\_date](@d1 date, @d2 date) as

begin

select SUM(Orderr.sum) as "Итоговая сумма за период"

from Orderr

where Orderr.date\_order between @d1 and @d2;

end

**Процедура “Добавление марки”**

CREATE procedure [dbo].[Добавление марки]( @nameMarks nvarchar(50),

@nameModels nvarchar(50)) as

begin

IF NOT EXISTS ( SELECT 1 FROM Models WHERE name = @nameModels)

BEGIN

INSERT INTO Models (name) VALUES (@nameModels)

END

IF NOT EXISTS ( SELECT 1 FROM Marks WHERE name = @nameMarks

AND model = (Select id from Models Where name = @nameModels))

BEGIN

insert into Marks (name, model)

Values (@nameMarks, (Select id from Models Where name = @nameModels))

END

end

**Процедура “Поставка запчастей”**

CREATE procedure [dbo].[Поставка запчастей]( @name nvarchar(30),

@contry nvarchar(50),

@brand nvarchar(50),

@number int,

@price money,

@guarantee int) as

begin

--Проверяю наличие записи о производителе (Страна, Бренд)

IF NOT EXISTS ( SELECT 1 FROM Manufacturer WHERE country = @contry AND brand = @brand )

BEGIN

--Если запись не найдена то добавляем ее

INSERT INTO Manufacturer (country, brand) VALUES (@contry, @brand)

END

--Проверяем наличие записи о запчасти (название, производитель, прайс и гарантия)

--Если отличается один атрибут то необходима новая запись

IF NOT EXISTS ( SELECT 1 FROM Spares WHERE name = @name

AND manufactur = (Select id From Manufacturer Where country=@contry and brand = @brand)

AND isnull(price,0) = isnull(@price,0) AND isnull(guarantee,0) = isnull(@guarantee,0))

BEGIN

--Добавление записи о запчасти

insert into Spares (name, manufactur,number, date\_of\_delivery, price , guarantee)

Values (@name, (Select id From Manufacturer Where country=@contry and brand = @brand),

@number, GETDATE(), @price, @guarantee)

END

ELSE

BEGIN

--Если запись была найдена то меняем у нее два параметра: количество и дата поставки

update Spares

SET number = number + @number,

date\_of\_delivery = GETDATE()

WHERE id =(SELECT id FROM Spares WHERE name = @name

AND manufactur = (Select id From Manufacturer Where country=@contry and brand = @brand)

AND isnull(price,0) = isnull(@price,0) AND isnull(guarantee,0) = isnull(@guarantee,0) );

END

end

## Приложение 3

## Скрипты создания триггеров

**Триггер “Вычитание количества запчастей”**

CREATE trigger [dbo].[Вычитание количество запчастей]

on [dbo].[Oerdering\_spares]

after insert

as

begin

if ((select Spares.number from Spares where Spares.id = (select spares from inserted)) >= (select number from inserted) )

begin

update Spares

set number = number- (select number from inserted)

where id = (select spares from inserted)

end

else

begin

delete from Oerdering\_spares

where id = (select id from inserted)

end

end

**Триггер “Добавление заказа запчастей”**

CREATE trigger [dbo].[Добавление заказа запчастей]

on [dbo].[Oerdering\_spares]

after insert, update

as

DECLARE @OrderID int

SELECT @OrderID = id FROM Orderr where id = (select orderr from inserted)

update Orderr

set Orderr.sum = (select sum(Service.price\*Ordering\_servuces.number\*(100 - Client.discount)/100)

from Orderr

inner join Ordering\_servuces on Ordering\_servuces.orderr = Orderr.id

inner join Service on Service.id = Ordering\_servuces.service

inner join Client on Client.id = Orderr.client

where Orderr.id = @OrderID )+(select sum(Spares.price\*Oerdering\_spares.number\*(100 - Client.discount)/100)

from Orderr

inner join Oerdering\_spares on Oerdering\_spares.orderr = Orderr.id

inner join Spares on Spares.id = Oerdering\_spares.spares

inner join Client on Client.id = Orderr.client

where Orderr.id =@OrderID and isnull(Spares.price,0)<>0 )

**Триггер “Удаление заказа запастей”**

CREATE trigger [dbo].[Удаление заказа запастей]

on [dbo].[Oerdering\_spares]

after delete

as

begin

DECLARE @OrderID int

SELECT @OrderID = id FROM Orderr where id = (select orderr from deleted)

update Orderr

set Orderr.sum = (select sum(Service.price\*Ordering\_servuces.number\*(100 - Client.discount)/100)

from Orderr

inner join Ordering\_servuces on Ordering\_servuces.orderr = Orderr.id

inner join Service on Service.id = Ordering\_servuces.service

inner join Client on Client.id = Orderr.client

where Orderr.id = @OrderID )+(select sum(Spares.price\*Oerdering\_spares.number\*(100 - Client.discount)/100)

from Orderr

inner join Oerdering\_spares on Oerdering\_spares.orderr = Orderr.id

inner join Spares on Spares.id = Oerdering\_spares.spares

inner join Client on Client.id = Orderr.client

where Orderr.id =@OrderID and isnull(Spares.price,0)<>0 )

update Spares

set number = number +(select number from deleted)

where id = (select spares from deleted)

end

**Триггер “Добавление заказа услуг”**

CREATE trigger [dbo].[Добавление заказа услуг]

on [dbo].[Ordering\_servuces]

after insert, update

as

DECLARE @OrderID int

SELECT @OrderID = id FROM Orderr where id = (select orderr from inserted)

update Orderr

set Orderr.sum = (select sum(Service.price\*Ordering\_servuces.number\*(100 - Client.discount)/100)

from Orderr

inner join Ordering\_servuces on Ordering\_servuces.orderr = Orderr.id

inner join Service on Service.id = Ordering\_servuces.service

inner join Client on Client.id = Orderr.client

where Orderr.id = @OrderID )+(select sum(Spares.price\*Oerdering\_spares.number\*(100 - Client.discount)/100)

from Orderr

inner join Oerdering\_spares on Oerdering\_spares.orderr = Orderr.id

inner join Spares on Spares.id = Oerdering\_spares.spares

inner join Client on Client.id = Orderr.client

where Orderr.id =@OrderID and isnull(Spares.price,0)<>0 )

**Триггер “Удаление заказа услуг”**

CREATE trigger [dbo].[Удаление заказа услуг]

on [dbo].[Ordering\_servuces]

after delete

as

DECLARE @OrderID int

SELECT @OrderID = id FROM Orderr where id = (select orderr from deleted)

update Orderr

set Orderr.sum = (select sum(Service.price\*Ordering\_servuces.number\*(100 - Client.discount)/100)

from Orderr

inner join Ordering\_servuces on Ordering\_servuces.orderr = Orderr.id

inner join Service on Service.id = Ordering\_servuces.service

inner join Client on Client.id = Orderr.client

where Orderr.id = @OrderID )+(select sum(Spares.price\*Oerdering\_spares.number\*(100 - Client.discount)/100)

from Orderr

inner join Oerdering\_spares on Oerdering\_spares.orderr = Orderr.id

inner join Spares on Spares.id = Oerdering\_spares.spares

inner join Client on Client.id = Orderr.client

where Orderr.id =@OrderID and isnull(Spares.price,0)<>0 )