МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине: «Разработка приложений баз данных для информационных систем»
на тему: «Разработка серверной части информационной системы в СУБД *MS SQL Server*»

Выполнил: студент гр. ИТП-31
Бондарев Е.Ю.
Принял: ректор
Асенчик О.Д.

Гомель 2023

Цель работы: разработать серверную часть клиент-серверной информационной системы, основанной на базе данных в заданной предметной области средствами СУБД *MS SQL Server*.

Задание:

1. Разработать логическую модель реляционной базы данных, моделирующую предметную область согласно своему варианту задания. Структура БД должна быть нормализована – таблицы должны удовлетворять требованиям третьей нормальной формы.

2. Создать базу данных и таблицы в СУБД *MS SQL Server* и заполнить их тестовым набором данных, для этого написать *Transact SQL* скрипт:

2.1. Создания базы данных и ее таблиц.

При создании таблиц должны быть назначены первичные и внешние ключи и установлены необходимые ограничения целостности данных. Наименования таблиц и полей давать в соответствии с соглашением об именовании этих объектов.

2.2. Заполнения не менее чем трех таблиц БД данными (см. пример внутри этого электронного курса).

При выборе таблиц для заполнения тестовыми наборами руководствоваться следующим:

выбранные таблицы должны содержать основную информацию, касающиеся предметной области приложения;

таблицы должны быть связаны непосредственно;

следует воздерживаться от выбора таблиц, характеризующих кадровую подсистему приложения (данные о сотрудниках и их должностях);

не менее, чем одна таблица должна находиться в схеме базы данных на стороне отношения «многие».

При этом заполнение осуществлять в следующем порядке: сначала генерируются данные для таблиц на стороне отношения «один» (таблицы-«справочники»), потом – в таблицы на стороне отношения «многие» («оперативные таблицы»).

БД заполнить записями в количестве, необходимом для отладки и демонстрации возможностей приложения. Таблицы на стороне отношения один должны содержать не менее 500 записей, таблице на стороне отношения многие должны содержать не менее 20000 записей.

3. Создать с использованием средств *Transact SQL* представления, позволяющие отображать данные в удобном для пользователя виде, и заменяющие часто используемые запросы на выборку из двух и более связанных таблиц.

4. Написать не менее трех хранимых процедур с параметрами для вставки и (или) обновление данных в таблицы базы данных.

Вариант задания указан на рисунке 1.

1. Издания (индекс издания, вид издания (газета, журнал), название издания, стоимость подписки на издание на 1 месяц (руб.))
2. Получатели (Ф.И.О. получателя, адрес получателя (улица, дом, квартира), телефон, e-mail).
3. Почтовые отделения города (индекс, адрес, Ф.И.О. руководителя, телефон, e-mail).
4. Подписка (получатель, издание, срок подписки (в месяцах), месяц и год начала доставки издания, отделение).
5. Улицы города (название улицы, почтовое отделение).
6. Сотрудники (Ф.И.О., должность, отделение).

Рисунок 2 – Вариант задания

Ход работы

В начале процесса проектирования базы данных была создана таблица с названием *"PublicationType"*, которая разработана с целью создания перечисления основных типов изданий (газеты и журналы). Это было сделано с учетом необходимости добавления новых типов изданий в будущем. Пример таблицы указан на рисунке 2. Листинг создания этой таблицы указана в приложении А.

	id	type
1	1	газета
2	2	журнал

Рисунок 2 – Пример таблицы *PublicationType*

Далее была создана таблица с названием *"Publication"*, в которой содержится информация о изданиях, такая как цена, название и тип. Эта таблица устанавливает связь таблицей *"PublicationType"*, которая была описана выше. Чтобы заполнить таблицу *"Publication"* с данными, была создана виртуальная таблица, содержащую названия изданий. Эти названия были добавлены в таблицу *"Publication"* в случайном порядке. Пример таблицы указан на рисунке 3. Листинг создания этой таблицы указана в приложении А.

	id	type_id	name	price
1	1	2	Взгляд	82.20
2	2	1	Комсомольская правда	23.74
3	3	1	Наука и техника	73.82
4	4	2	Финансовые новости	97.88
5	5	1	Комсомольская правда	71.09
6	6	1	Спорт-Экспресс	91.75
7	7	2	Искусство и культура	46.60
8	8	1	Искусство и культура	48.47
9	9	2	Искусство и культура	72.86
10	10	1	Асахи симбун	83.85

Рисунок 3 – Пример таблицы *Publication*

После была разработана таблица *"RecipientAddress"*, в которой хранилась информация о адресе получателя, включая улицу, дом и квартиру. Для заполнения этой таблицы были использованы случайные названиями улиц из виртуальной таблицы.

Далее была создана ещё одна таблица с названием *"Recipient"*, в которой была собрана полная информация о получателе, включая его ФИО, адрес, контактный телефон и адрес электронной почты. Эта таблица устанавливает связь между получателями и их адресами. Пример таблиц *RecipientAddress* и *Recipient* указан на рисунках 4 и 5. Листинг создания таблиц указана в приложении А.

	id	street	house	apartment
1	1	Колхозная улица	44	16
2	2	Рабочая улица	62	18
3	3	Луговая улица	86	13
4	4	Пролетарская улица	1	5
5	5	Красивая улица	45	16
6	6	Центральная улица	37	10
7	7	Заводская улица	11	5

Рисунок 4 – Пример таблицы *RecipientAddress*

	id	name	middlename	surname	address_id	mobile_phone	email
1	1	Иван	Петрович	Смирнов	50	+375661035312	CCA528DB-7@gmail.com
2	2	Иван	Иванович	Сидоров	77	+375635249163	B9E6F521-4@gmail.com
3	3	Александр	Александрович	Антонова	50	+375169649821	0CB11227-5@gmail.com
4	4	Евгений	Андреевич	Смирнов	66	+375652379774	F61A6CB4-1@gmail.com
5	5	Иван	Сергеевна	Иванов	50	+375536456322	BB448D9C-5@gmail.com
6	6	Евгений	Андреевич	Антонова	31	+375707283833	AD0EB7D1-F@gmail.com
7	7	Петр	Андреевич	Петров	58	+375892958381	C45997A2-3@gmail.com
8	8	Евгений	Петрович	Петров	14	+375145098802	A17B58AD-1@gmail.com
9	9	Александр	Петрович	Иванов	41	+375374132174	D6479D0B-A@gmail.com
10	10	Кирил	Александрович	Петров	18	+375910414843	02ABA095-C@gmail.com
11	11	Петр	Сергеевна	Сидоров	27	+375899458025	8B66CC7E-C@gmail.com
12	12	Евгений	Александрович	Смирнов	92	+375301389832	1D10CD4C-3@gmail.com

Рисунок 5 – Пример таблицы *Recipient*

В дальнейшей работе была создана таблица *"EmployeePosition"*. Эта таблица содержит информацию о должностях сотрудников в почтовом отделении и так же выступает перечислением как таблица *"PublicationType"*. В таблице *"EmployeePosition"* были перечислены различные должности сотрудников, такие как "почтальон", "кассир" и другие.

Это обеспечивает гибкость в управлении должностями сотрудников в почтовом отделении и позволяет легко добавлять новые должности в

будущем. Пример таблицы *EmployeePosition* указан на рисунке 6. Листинг создания этой таблицы указана в приложении А.

	id	position
1	2	Кассир
2	4	Менеджер по доставке
3	1	Почтальон
4	3	Сортировщик

Рисунок 6 – Пример таблицы *EmployeePosition*

Далее, в ходе проектирования базы данных, была разработана таблица с названием *"Office"*, предназначенная для хранения информации о почтовых отделениях в разных городах. Пример таблицы *Office* указан на рисунке 7. Листинг создания этой таблицы указана в приложении А.

	id	owner_name	owner_middlename	onwner_sumame	street_name	mobile_phone	email
1	1	Иван	Александрович	Смирнов	Космическая улица	+375395728069	C22DE71B-F@gmail.com
2	2	Петр	Петрович	Петров	Пионерская улица	+375169829397	E44ADADE-A@gmail.com
3	3	Петр	Иванович	Сидоров	Космонавтов улица	+375236193338	5EBFC51D-D@gmail.com
4	4	Иван	Александрович	Иванов	Красивая улица	+375649548712	DC95F699-E@gmail.com
5	5	Кирил	Сергеевна	Иванов	Красная улица	+375718172046	E2F8F9B8-9@gmail.com
6	6	Иван	Александрович	Иванов	Новая улица	+375839061268	6B66F3B8-9@gmail.com
7	7	Иван	Петрович	Смирнов	Заводская улица	+375152643086	3200C1B2-A@gmail.com
8	8	Петр	Иванович	Сидоров	Тихая улица	+375634969925	15F30F49-3@gmail.com
9	9	Петр	Александрович	Иванов	Шевченко улица	+375205779911	54B832FF-7@gmail.com
10	10	Иван	Андреевич	Петров	Песчаная улица	+375864020586	1EE90F30-2@gmail.com

Рисунок 7 – Пример таблицы *Office*

Таблица *"Employee"* в базе данных содержит следующую информацию о сотрудниках почтового отделения: Ф.И.О, офис (ссылается на таблицу *"Office"*) и должность сотрудника (ссылается на таблицу *"EmployeePosition"*). Пример таблицы *Employee* указан на рисунке 8. Листинг создания этой таблицы указана в приложении А.

	id	name	middlename	sumame	position_id	office_id
1	15	Кирил	Петрович	Петров	2	58
2	57	Иван	Сергеевна	Петров	2	19
3	82	Кирил	Сергеевна	Антонова	4	60
4	100	Кирил	Петрович	Сидоров	1	56
5	132	Александр	Иванович	Антонова	2	80
6	180	Евгений	Андреевич	Иванов	1	55
7	218	Александр	Андреевич	Антонова	2	15
8	233	Арсений	Сергеевна	Сидоров	4	72
9	255	Евгений	Иванович	Петров	1	65
10	281	Александр	Сергеевна	Смирнов	3	87

Рисунок 8 – Пример таблицы *Employee*

Таблица *"Subscription"* в базе данных содержит следующие поля: Получатель (ссылка на таблицу *"Recipient"*), издание (ссылается на таблицу *"Publication"*), срок подписки, месяц и год начала доставки, отделение (ссылается на таблицу *"Office"*). Пример таблицы *Subscription* указан на рисунке 8. Листинг создания этой таблицы указана в приложении А.

1	80	25	7	22	12.2019
2	23	22	1	21	04.2016
3	96	39	11	52	07.2021
4	77	37	3	42	05.2017
5	4	74	3	33	10.2022
6	25	21	3	39	05.2014
7	41	33	7	38	05.2020
8	49	54	10	3	02.2020
9	10	26	9	17	04.2019
10	15	54	6	65	05.2018

Рисунок 8 – Пример таблицы *Subscription*

После создания базы данных, была создана диаграмма, которая иллюстрирует зависимости между таблицами. Пример диаграммы базы данных указан на рисунке 9.

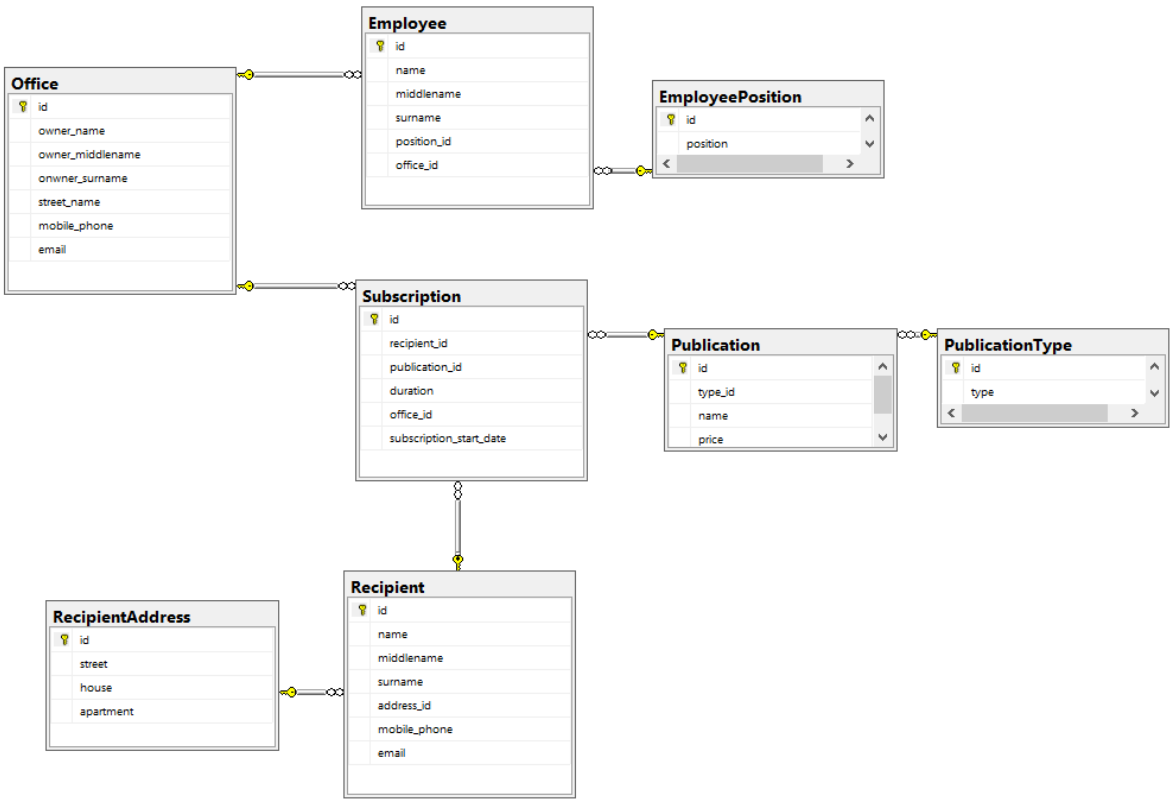

Рисунок 9 – Пример диаграммы базы данных

После написания всех скриптов для создания таблиц был написан скрипт для заполнения таблиц. Пример данного скрипта указан в приложение А. Далее были созданы три представления для удобного получения доступа к данным из таблицы.

Первым было создано представления для получения всех данных из таблицы изданий. Пример работы данного представления указан на рисунке 10. Листинг кода по созданию этого представления указан в приложении А.

	PublicationID	PublicationType	PublicationName	PublicationPrice
1	1	журнал	Взгляд	82.20
2	2	газета	Комсомольская правда	23.74
3	3	газета	Наука и техника	73.82
4	4	журнал	Финансовые новости	97.88
5	5	газета	Комсомольская правда	71.09
6	6	газета	Спорт-Экспресс	91.75
7	7	журнал	Искусство и культура	46.60
8	8	газета	Искусство и культура	48.47
9	9	журнал	Искусство и культура	72.86
10	10	газета	Асахи симбун	83.85

Рисунок 10 – Пример работы представления *PublicationView*

Вторым было создано представления для получения всех данных из таблицы получателей изданий. Пример работы данного представления указан на рисунке 11. Листинг кода по созданию этого представления указан в приложении А.

	RecipientID	RecipientName	RecipientMiddlename	RecipientSurname	RecipientStreet	RecipientHouse	RecipientApartment	RecipientMobilePhone	RecipientEmail
1	1	Иван	Петрович	Смирнов	Молодежная улица	33	12	+375661035312	CCA528DB-7@gmail.com
2	2	Иван	Иванович	Сидоров	Юбилейная улица	11	19	+375635249163	B9E6F521-4@gmail.com
3	3	Александр	Александрович	Антонова	Молодежная улица	33	12	+375169649821	0CB11227-5@gmail.com
4	4	Евгений	Андреевич	Смирнов	Юбилейная улица	15	16	+375652379774	F61A6CB4-1@gmail.com
5	5	Иван	Сергеевна	Иванов	Молодежная улица	33	12	+375536456322	BB448D9C-5@gmail.com
6	6	Евгений	Андреевич	Антонова	Щорса улица	76	18	+375707283833	AD0EB7D1-F@gmail.com
7	7	Петр	Андреевич	Петров	Шевченко улица	41	16	+375892958381	C45997A2-3@gmail.com
8	8	Евгений	Петрович	Петров	Набережная улица	47	6	+375145098802	A17B58AD-1@gmail.com
9	9	Александр	Петрович	Иванов	Пушкинская улица	82	4	+375374132174	D6479D0B-A@gmail.com
10	10	Кирил	Александрович	Петров	Свердловская улица	100	7	+375910414843	02ABA095-C@gmail.com

Рисунок 11 – Пример работы представления *RecipientView*

Третьим было создано представления для получения всех данных о подписках. Пример работы данного представления указан на рисунке 12. Листинг кода по созданию этого представления указан в приложении А.

	SubscriptionID	RecipientName	PublicationName	SubscriptionDuration	OfficeOwnerName	SubscriptionStartDate
1	1	Кирил	Наука и техника	7	Иван	12.2019
2	2	Александр	Деловой мир	1	Петр	04.2016
3	3	Петр	Путешествия и приключения	11	Иван	07.2021

Рисунок 12 – Пример работы представления *SubscriptionView*

Далее было создано представления для получения всех данных из таблицы офисов. Пример работы данного представления указан на рисунке 13. Листинг кода по созданию этого представления указан в приложении А.

	OfficeID	OwnerName	OwnerMiddlename	OwnerSurname	StreetName	MobilePhone	Email
1	1	Иван	Александрович	Смирнов	Космическая улица	+375395728069	C22DE71B-F@gmail.com
2	2	Петр	Петрович	Петров	Пионерская улица	+375169829397	E44ADADE-A@gmail.com
3	3	Петр	Иванович	Сидоров	Космонавтов улица	+375236193338	5EBFC51D-D@gmail.com
4	4	Иван	Александрович	Иванов	Красивая улица	+375649548712	DC95F699-E@gmail.com
5	5	Кирил	Сергеевна	Иванов	Красная улица	+375718172046	E2F8F9B8-9@gmail.com
6	6	Иван	Александрович	Иванов	Новая улица	+375839061268	6B66F3B8-9@gmail.com
7	7	Иван	Петрович	Смирнов	Заводская улица	+375152643086	3200C1B2-A@gmail.com
8	8	Петр	Иванович	Сидоров	Тихая улица	+375634969925	15F30F49-3@gmail.com
9	9	Петр	Александрович	Иванов	Шевченко улица	+375205779911	54B832FF-7@gmail.com
10	10	Иван	Андреевич	Петров	Песчаная улица	+375864020586	1EE90F30-2@gmail.com

Рисунок 13 – Пример работы представления *OfficeView*

Далее были реализованы пять хранимых процедур для добавления новых записей в таблицы *Employee*, *Office*, *Publication*, *Recipient* и *Subscription*. Листинг данных хранимых процедур указан в приложении А.

Вывод: в процессе выполнения лабораторной работы была разработана и настроена база данных в среде MS SQL Server для использования в серверной части клиент-серверного приложения. Созданы необходимые таблицы баз данных и заполны данными. Также были разработаны хранимые процедуры, которые обеспечивают возможность вставки новых записей в таблицы. Кроме того, были созданы представления, упрощающие доступ к данным из базы и обеспечивающие более удобное представление информации.

ПРИЛОЖНИЕ А
Листинг скрипта для генерации базы данных

```
-- Создание базы с проверкой
USE master
IF NOT EXISTS (SELECT name FROM sys.databases WHERE name = 'SubsCity1')
BEGIN
   CREATE DATABASE SubsCity1
END

GO
ALTER DATABASE SubsCity1 SET RECOVERY SIMPLE
GO

USE [SubsCity1]

-- Создание таблиц с проверкой
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'PublicationType')
BEGIN
   CREATE TABLE [dbo].[PublicationType](
      [id] [int] IDENTITY(1,1) NOT NULL,
      [type] [nvarchar](20) NOT NULL,
      PRIMARY KEY CLUSTERED
      (
         [id] ASC
      )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
   )
END


IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'RecipientAddress')
BEGIN
	CREATE TABLE [dbo].[RecipientAddress](
		[id] [int] IDENTITY(1,1) NOT NULL,
		[street] [nvarchar](50) NULL,
		[house] [int] NULL,
		[apartment] [int] NULL,
	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]
END


IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'EmployeePosition')
BEGIN
	CREATE TABLE [dbo].[EmployeePosition](
		[id] [int] IDENTITY(1,1) NOT NULL,
		[position] [nvarchar](50) NULL,
```

```
	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
	UNIQUE NONCLUSTERED
	(
		[position] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]
END

IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'Publication')
BEGIN
	CREATE TABLE [dbo].[Publication](
		[id] [int] IDENTITY(1,1) NOT NULL,
		[type_id] [int] NOT NULL,
		[name] [nvarchar](70) NOT NULL,
		[price] [decimal](10, 2) NOT NULL,
	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]

	ALTER TABLE [dbo].[Publication]  WITH CHECK ADD FOREIGN KEY([type_id])
	REFERENCES [dbo].[PublicationType] ([id])
END

IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'Recipient')
BEGIN
	CREATE TABLE [dbo].[Recipient](
		[id] [int] IDENTITY(1,1) NOT NULL,
		[name] [nvarchar](20) NOT NULL,
		[middlename] [nvarchar](20) NOT NULL,
		[surname] [nvarchar](20) NOT NULL,
		[address_id] [int] NOT NULL,
		[mobile_phone] [nvarchar](20) NOT NULL,
		[email] [nvarchar](255) NOT NULL,
	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
	UNIQUE NONCLUSTERED
	(
		[mobile_phone] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
	UNIQUE NONCLUSTERED
	(
		[email] ASC
```

```
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]

	ALTER TABLE [dbo].[Recipient]  WITH CHECK ADD FOREIGN KEY([address_id])
	REFERENCES [dbo].[RecipientAddress] ([id])
END


IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME = 'Office')
BEGIN
	CREATE TABLE [dbo].[Office](
		[id] [int] IDENTITY(1,1) NOT NULL,
		[owner_name] [nvarchar](20) NOT NULL,
		[owner_middlename] [nvarchar](20) NOT NULL,
		[onwner_surname] [nvarchar](20) NOT NULL,
		[street_name] [nvarchar](50) NOT NULL,
		[mobile_phone] [nvarchar](20) NOT NULL,
		[email] [nvarchar](255) NOT NULL,
	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
	UNIQUE NONCLUSTERED
	(
		[mobile_phone] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
	UNIQUE NONCLUSTERED
	(
		[email] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]
END


IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'Employee')
BEGIN
	CREATE TABLE [dbo].[Employee](
		[id] [int] IDENTITY(1,1) NOT NULL,
		[name] [nvarchar](20) NOT NULL,
		[middlename] [nvarchar](20) NOT NULL,
		[surname] [nvarchar](20) NOT NULL,
		[position_id] [int] NOT NULL,
		[office_id] [int] NOT NULL,
	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]

	ALTER TABLE [dbo].[Employee]  WITH CHECK ADD FOREIGN KEY([office_id])
	REFERENCES [dbo].[Office] ([id])
```

```
	ALTER TABLE [dbo].[Employee]  WITH CHECK ADD FOREIGN KEY([position_id])
	REFERENCES [dbo].[EmployeePosition] ([id])
END



IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND TABLE_NAME =
'Subscription')
BEGIN
	CREATE TABLE [dbo].[Subscription](
  [id] [int] IDENTITY(1,1) NOT NULL,
  [recipient_id] [int] NOT NULL,
  [publication_id] [int] NOT NULL,
  [duration] [int] NOT NULL,
  [office_id] [int] NOT NULL,
  [subscription_start_date] [nvarchar](7) NOT NULL

	PRIMARY KEY CLUSTERED
	(
		[id] ASC
	)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
	) ON [PRIMARY]


	ALTER TABLE [dbo].[Subscription]  WITH CHECK ADD FOREIGN KEY([office_id])
	REFERENCES [dbo].[Office] ([id])


	ALTER TABLE [dbo].[Subscription]  WITH CHECK ADD FOREIGN KEY([publication_id])
	REFERENCES [dbo].[Publication] ([id])


	ALTER TABLE [dbo].[Subscription]  WITH CHECK ADD FOREIGN KEY([recipient_id])
	REFERENCES [dbo].[Recipient] ([id])
END

-- Заполнение таблиц (если они пусты)
IF NOT EXISTS (SELECT * FROM [dbo].[PublicationType])
BEGIN
  INSERT INTO [dbo].[PublicationType] ([type])
  VALUES ('газета'), ('журнал')
END

DECLARE @StreetNames TABLE (Name NVARCHAR(50))
  INSERT INTO @StreetNames (Name)
  VALUES
    ('Пролетарская улица'),
    ('Ленинская улица'),
    ('Гагарина улица'),
    ('Советская улица'),
    ('Пушкинская улица'),
    ('Московская улица'),
    ('Кировская улица'),
    ('Парковая улица'),
    ('Садовая улица'),
```

('Комсомольская улица'),
('Школьная улица'),
('Жукова улица'),
('Мичурина улица'),
('Свердлова улица'),
('Октябрьская улица'),
('Горького улица'),
('Красноармейская улица'),
('Рабочая улица'),
('Зеленая улица'),
('Трудовая улица'),
('Полярная улица'),
('Красная улица'),
('Строителей улица'),
('Молодежная улица'),
('Центральная улица'),
('Новая улица'),
('Солнечная улица'),
('Заречная улица'),
('Пионерская улица'),
('Речная улица'),
('Восточная улица'),
('Западная улица'),
('Южная улица'),
('Северная улица'),
('Цветочная улица'),
('Лесная улица'),
('Юбилейная улица'),
('Гранитная улица'),
('Маяковского улица'),
('Первомайская улица'),
('Коммунальная улица'),
('Чкалова улица'),
('Горная улица'),
('Сиреневая улица'),
('Сосновая улица'),
('Дружбы улица'),
('Озерная улица'),
('Заводская улица'),
('Вокзальная улица'),
('Партизанская улица'),
('Островская улица'),
('Городская улица'),
('Карла Маркса улица'),
('Железнодорожная улица'),
('Набережная улица'),
('Мирная улица'),
('Севастопольская улица'),
('Колхозная улица'),
('Совхозная улица'),
('Театральная улица'),
('Лермонтова улица'),
('Пушкинская улица'),
('Мичуринская улица'),
('Свердловская улица'),
('Щорса улица'),
('Смирнова улица'),

```
    ('Гусарская улица'),
    ('Петровская улица'),
    ('Космонавтов улица'),
    ('Суворова улица'),
    ('Фрунзе улица'),
    ('Толстого улица'),
    ('Горького улица'),
    ('Шевченко улица'),
    ('Греческая улица'),
    ('Воронцовская улица'),
    ('Скверная улица'),
    ('Сельская улица'),
    ('Холодильная улица'),
    ('Степная улица'),
    ('Заводская улица'),
    ('Космическая улица'),
    ('Речная улица'),
    ('Парковая улица'),
    ('Береговая улица'),
    ('Школьная улица'),
    ('Соседская улица'),
    ('Луговая улица'),
    ('Озерная улица'),
    ('Красивая улица'),
    ('Полевая улица'),
    ('Дачная улица'),
    ('Живописная улица'),
    ('Просторная улица'),
    ('Зеленая улица'),
    ('Чистая улица'),
    ('Тихая улица'),
    ('Песчаная улица')

IF NOT EXISTS (SELECT * FROM [dbo].[RecipientAddress])
BEGIN

  DECLARE @House INT
  DECLARE @Apartment INT
  DECLARE @RandomStreet NVARCHAR(20)

  WHILE (SELECT COUNT(*) FROM [dbo].[RecipientAddress]) < 100
  BEGIN
    SELECT TOP 1 @RandomStreet = Name FROM @StreetNames ORDER BY NEWID()
    SET @House = CAST(RAND() * 100 AS INT) + 1
    SET @Apartment = CAST(RAND() * 20 AS INT) + 1

    INSERT INTO [dbo].[RecipientAddress] ([street], [house], [apartment])
    VALUES (@RandomStreet, @House, @Apartment)
  END
END

IF NOT EXISTS (SELECT * FROM [dbo].[EmployeePosition])
BEGIN
        INSERT INTO [dbo].[EmployeePosition] ([position])
        VALUES
                ('Почтальон'),
                ('Кассир'),
```

```
                    ('Сортировщик'),
                    ('Менеджер по доставке')
END

-- Проверка, пуста ли таблица Publication
IF NOT EXISTS (SELECT * FROM [dbo].[Publication])
BEGIN
  -- Заполнение таблицы Publication случайными данными
  DECLARE @PublicationNames TABLE (Name NVARCHAR(70))
  INSERT INTO @PublicationNames (Name)
  VALUES
    ('Утренние новости'),
    ('Вечерний вестник'),
    ('Здоровье и красота'),
    ('Наука и техника'),
    ('Деловой мир'),
    ('Спортивные новости'),
    ('Искусство и культура'),
    ('Политика и общество'),
    ('Финансовые новости'),
    ('Путешествия и приключения'),
                    ('Асахи симбун'),
                    ('Спорт-Экспресс'),
                    ('Взгляд'),
                    ('Московский комсомолец'),
                    ('Комсомольская правда')


        DECLARE @PublicationType INT
  DECLARE @PublicationName NVARCHAR(50)
  DECLARE @PublicationPrice DECIMAL(10, 2)

  WHILE (SELECT COUNT(*) FROM [dbo].[Publication]) < 100
  BEGIN
    SET @PublicationType = CASE WHEN RAND() > 0.5 THEN 1 ELSE 2 END
    SELECT TOP 1 @PublicationName = Name FROM @PublicationNames ORDER BY NEWID()
    SET @PublicationPrice = CAST(RAND() * 100 AS DECIMAL(10, 2))

    INSERT INTO [dbo].[Publication] ([type_id], [name], [price])
    VALUES (@PublicationType, @PublicationName, @PublicationPrice)
  END
END

DECLARE @Names TABLE (Name NVARCHAR(20))
DECLARE @MiddleNames TABLE (MiddleName NVARCHAR(20))
DECLARE @Surnames TABLE (Surname NVARCHAR(20))

INSERT INTO @Names (Name) VALUES ('Евгений'), ('Иван'), ('Петр'), ('Кирил'), ('Александр'), ('Арсений')
INSERT INTO @MiddleNames (MiddleName) VALUES ('Иванович'), ('Андреевич'), ('Петрович'), ('Сергеевна'),
('Александрович')
INSERT INTO @Surnames (Surname) VALUES ('Иванов'), ('Антонова'), ('Петров'), ('Смирнов'), ('Сидоров')

DECLARE @AddressID INT
DECLARE @MobilePhone NVARCHAR(20)
DECLARE @Email NVARCHAR(255)

IF NOT EXISTS (SELECT * FROM [dbo].[Recipient])
```

```
BEGIN
	-- Заполнение таблицы Recipient
	WHILE (SELECT COUNT(*) FROM [dbo].[Recipient]) < 100
	BEGIN
		SET @AddressID = CAST(RAND() * 100 AS INT)
		SET @MobilePhone = '+375' + CAST(100000000 + CAST(RAND() * 899999999 AS INT) AS NVARCHAR(20))
		SET @Email = LEFT(NEWID(), 10) + '@gmail.com'

		INSERT INTO [dbo].[Recipient] ([name], [middlename], [surname], [address_id], [mobile_phone], [email])
		SELECT TOP 1
			(SELECT TOP 1 Name FROM @Names ORDER BY NEWID()),
			(SELECT TOP 1 MiddleName FROM @MiddleNames ORDER BY NEWID()),
			(SELECT TOP 1 Surname FROM @Surnames ORDER BY NEWID()),
			@AddressID,
			@MobilePhone,
			@Email
	END
END


IF NOT EXISTS (SELECT * FROM [dbo].[Office])
BEGIN
	WHILE (SELECT COUNT(*) FROM [dbo].[Office]) < 100
	BEGIN
		SET @MobilePhone = '+375' + CAST(100000000 + CAST(RAND() * 899999999 AS INT) AS NVARCHAR(20))
		SET @Email = LEFT(NEWID(), 10) + '@gmail.com'

		INSERT INTO [dbo].[Office] ([owner_name], [owner_middlename], [onwner_surname], [street_name],
[mobile_phone], [email])
		SELECT TOP 1
			(SELECT TOP 1 Name FROM @Names ORDER BY NEWID()),
			(SELECT TOP 1 MiddleName FROM @MiddleNames ORDER BY NEWID()),
			(SELECT TOP 1 Surname FROM @Surnames ORDER BY NEWID()),
			(SELECT TOP 1 Name FROM @StreetNames ORDER BY NEWID()),
			@MobilePhone,
			@Email
	END
END

IF NOT EXISTS (SELECT * FROM [dbo].[Employee])
BEGIN
	DECLARE @PositionID INT
	DECLARE @OfficeID INT
	WHILE (SELECT COUNT(*) FROM [dbo].[Employee]) < 100
	BEGIN
		SET @PositionID = CAST(RAND() * 100 + 1 AS INT)
		SET @OfficeID = CAST(RAND() * 100 + 1 AS INT)

		INSERT INTO [dbo].[Employee] ([name], [middlename], [surname], [position_id], [office_id])
		SELECT TOP 1
			(SELECT TOP 1 Name FROM @Names ORDER BY NEWID()),
			(SELECT TOP 1 MiddleName FROM @MiddleNames ORDER BY NEWID()),
			(SELECT TOP 1 Surname FROM @Surnames ORDER BY NEWID()),
			@PositionID,
			@OfficeID
	END
END
```

```
IF NOT EXISTS (SELECT * FROM [dbo].[Subscription])
BEGIN
	DECLARE @RecipientID INT
	DECLARE @PublicationID INT
	DECLARE @Duration INT
	DECLARE @StartDate NVARCHAR(7)

	WHILE (SELECT COUNT(*) FROM [dbo].[Subscription]) < 100
	BEGIN
		SET @RecipientID = CAST(RAND() * 100 + 1 AS INT)
		SET @PublicationID = CAST(RAND() * 100 + 1 AS INT)
		SET @Duration = CAST(RAND() * 12 + 1 AS INT)
		SET @OfficeID = CAST(RAND() * 100 + 1 AS INT)

		SET @StartDate = RIGHT('0' + CAST(ROUND(RAND() * 11 + 1, 0) AS NVARCHAR(2)), 2) + '.' +
CAST(ROUND(RAND() * 9 + 2014, 0) AS NVARCHAR(4))

		INSERT INTO [dbo].[Subscription] ([recipient_id], [publication_id], [duration], [office_id],
[subscription_start_date])
		VALUES (@RecipientID, @PublicationID, @Duration, @OfficeID, @StartDate)
	END
END


-- Создание хранимых процедур
IF OBJECT_ID ( 'dbo.InsertPublication', 'P' ) IS NOT NULL
  DROP PROCEDURE dbo.InsertPublication;
GO
	CREATE PROCEDURE InsertPublication
		@type_id INT,
		@name NVARCHAR(70),
		@price DECIMAL(10, 2)
	AS
	BEGIN
		INSERT INTO [dbo].[Publication] ([type_id], [name], [price])
		VALUES (@type_id, @name, @price)
	END
GO

IF OBJECT_ID ( 'dbo.InsertRecipient', 'P' ) IS NOT NULL
  DROP PROCEDURE dbo.InsertRecipient;
GO
	CREATE PROCEDURE InsertRecipient
		@name NVARCHAR(20),
		@middlename NVARCHAR(20),
		@surname NVARCHAR(20),
		@street NVARCHAR(50),
		@house INT,
		@apartment INT,
		@mobile_phone NVARCHAR(20),
		@email NVARCHAR(255)
	AS
	BEGIN
		DECLARE @AddressID INT
```

```
		INSERT INTO [dbo].[RecipientAddress] ([street], [house], [apartment])
		VALUES (@street, @house, @apartment)

		SET @AddressID = SCOPE_IDENTITY()

		INSERT INTO [dbo].[Recipient] ([name], [middlename], [surname], [address_id], [mobile_phone], [email])
		VALUES (@name, @middlename, @surname, @AddressID, @mobile_phone, @email)
	END
GO

IF OBJECT_ID ( 'dbo.InsertOffice', 'P' ) IS NOT NULL
  DROP PROCEDURE dbo.InsertOffice;
GO
	CREATE PROCEDURE InsertOffice
		@owner_name NVARCHAR(20),
		@owner_middlename NVARCHAR(20),
		@owner_surname NVARCHAR(20),
		@street_name NVARCHAR(50),
		@mobile_phone NVARCHAR(20),
		@email NVARCHAR(255)
	AS
	BEGIN
		INSERT INTO [dbo].[Office] ([owner_name], [owner_middlename], [onwner_surname], [street_name],
[mobile_phone], [email])
		VALUES (@owner_name, @owner_middlename, @owner_surname, @street_name, @mobile_phone,
@email)
	END
GO

IF OBJECT_ID ( 'dbo.InsertSubscription', 'P' ) IS NOT NULL
  DROP PROCEDURE dbo.InsertSubscription;
GO
	CREATE PROCEDURE InsertSubscription
		@recipient_id INT,
		@publication_id INT,
		@duration INT,
		@office_id INT,
		@subscription_start_date NVARCHAR(7)
	AS
	BEGIN
		INSERT INTO [dbo].[Subscription] ([recipient_id], [publication_id], [duration], [office_id],
[subscription_start_date])
		VALUES (@recipient_id, @publication_id, @duration, @office_id, @subscription_start_date)
	END
GO

IF OBJECT_ID ( 'dbo.InsertEmployee', 'P' ) IS NOT NULL
  DROP PROCEDURE dbo.InsertEmployee;
GO
	CREATE PROCEDURE InsertEmployee
		@name NVARCHAR(20),
		@middlename NVARCHAR(20),
		@surname NVARCHAR(20),
		@position_id INT,
		@office_id INT
	AS
	BEGIN
```

```
                INSERT INTO [dbo].[Employee] ([name], [middlename], [surname], [position_id], [office_id])
                VALUES (@name, @middlename, @surname, @position_id, @office_id)
        END
GO

-- Создание представлений
IF NOT EXISTS (SELECT * FROM sys.views WHERE [name] = 'PublicationView')
BEGIN
  EXEC('
    CREATE VIEW PublicationView AS
    SELECT
      p.id AS PublicationID,
      pt.type AS PublicationType,
      p.name AS PublicationName,
      p.price AS PublicationPrice
    FROM
      [dbo].[Publication] p
    JOIN
      [dbo].[PublicationType] pt ON p.type_id = pt.id;
  ');
END

IF NOT EXISTS (SELECT * FROM sys.views WHERE [name] = 'RecipientView')
BEGIN
  EXEC('
    CREATE VIEW RecipientView AS
  SELECT
    r.id AS RecipientID,
    r.name AS RecipientName,
    r.middlename AS RecipientMiddlename,
    r.surname AS RecipientSurname,
    ra.street AS RecipientStreet,
    ra.house AS RecipientHouse,
    ra.apartment AS RecipientApartment,
    r.mobile_phone AS RecipientMobilePhone,
    r.email AS RecipientEmail
  FROM
    [dbo].[Recipient] r
  JOIN
    [dbo].[RecipientAddress] ra ON r.address_id = ra.id;
  ');
END

IF NOT EXISTS (SELECT * FROM sys.views WHERE [name] = 'SubscriptionView')
BEGIN
  EXEC('
    CREATE VIEW SubscriptionView AS
        SELECT
                s.id AS SubscriptionID,
                r.name AS RecipientName,
                p.name AS PublicationName,
                s.duration AS SubscriptionDuration,
                o.owner_name AS OfficeOwnerName,
                s.subscription_start_date AS SubscriptionStartDate
        FROM
                [dbo].[Subscription] s
        JOIN
```

```
                [dbo].[Recipient] r ON s.recipient_id = r.id
        JOIN
                [dbo].[Publication] p ON s.publication_id = p.id
        JOIN
                [dbo].[Office] o ON s.office_id = o.id;
  ');
END

IF NOT EXISTS (SELECT * FROM sys.views WHERE [name] = 'OfficeView')
BEGIN
  EXEC('
    CREATE VIEW OfficeView AS
        SELECT
                id AS OfficeID,
                owner_name AS OwnerName,
                owner_middlename AS OwnerMiddlename,
                onwner_surname AS OwnerSurname,
                street_name AS StreetName,
                mobile_phone AS MobilePhone,
                email AS Email
        FROM
                [dbo].[Office];
  ');
END
```