УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчёт по лабораторной работе №2

По предмету

Базы данных

Выполнил:

Гладкий М.Г.

Проверил:

Фадеева Е.Е.

Группа 851005

Минск 2021

1) Какие каскадные операции необходимы в этой базе данных? Опишите их.

Исходная модель:

- Для отношения account установить ON UPDATE CASCADE, ON DELETE NO ACTION, нужно изменять status на «Удален» или «Заблокирован». Это необходимо, так как есть возможность повреждения данных в отношениях payment\_operational и payment\_archive ввиду того, что в них содержатся id на счета пользователей. Например, удалённый пользователь отправлял кому-либо деньги на счет, тогда у этого пользователя пропадёт информация о том, кто перечислял ему деньги в прошлом. Так проявляется аномалия удаления. При изменении id счета необходимо изменять его и в прошедших транзакциях.

- То же самое необходимо делать с отношением account\_owner, чтобы не нарушить целостность данных в account.

- Для отношения site\_pages ON UPDATE CASCADE, ON DELETE SET NULL, так как дочерние страницы нельзя удалять при удалении страницы родителя.

- Для остальных отношений ON UPDATE CASCADE, ON DELETE NO ACTION

Модель после 1-й лабораторной работы:

- Те же операции, что и в исходной модели.

- Для отношений physical\_person, juridical\_person ON UPDATE CASCADE, ON DELETE RESTRICT, так как необходимо сохранять конкретную и актуальную информацию о клиентах банка.

2) Существует ли возможность аномалий операций вставки, обновления, удаления данных? Составьте список таких случаев и внесите в базу данных исправления, позволяющие избежать таких аномалий.

Исходная модель:

- После удаления родителя в отношении site\_pages необходимо указывать в дочерней странице родителя удаленной страницы. Реализовать это можно при помощи триггера

- В отношении statuses необходимо гарантировать уникальность статуса в поле s\_name, что можно сделать при помощи создания уникального индекса. В ином случае будет наблюдаться аномалия вставки. Аналогично для отношения offices в поле of\_name.

- После вставки нового объекта в отношение accounts\_m2m\_statuses возможна утеря данных предыдущего обновления статуса.

Модель после 1-й лабораторной работы:

- То же самое, что и в исходной модели, за исключением аномалий со статусом.

- В отношении account необходимо обеспечить уникальность поля status с помощью уникального индекса.

3) Можно ли использовать схемы «звезда» или «снежинка» с этой базой данных, чтобы избежать некоторых аномалий операций с данными? Переработайте схему, сравните новую с исходной и составьте список аномалий работы с данными, которые были устранены (или, наоборот, появились).

Исходная модель:

- Думаю, в этом нет необходимости и они не устранят найденные аномалии работы с данными.

Модель после 1-й лабораторной работы:

- Думаю, в этом нет необходимости и они не устранят найденные аномалии работы с данными.

4) Составьте список всех функциональных зависимостей в базе данных.

Исходная модель:

- pa\_from, pa\_to, pa\_date\_time -> pa\_money

- po\_from, po\_to, po\_date\_time -> po\_money

- ams\_last\_update -> ams\_s\_id

- a\_is\_system -> a\_owner

Модель после 1-й лабораторной работы:

- pa\_from, pa\_to, pa\_date\_time -> pa\_amount

- po\_from, po\_to, po\_date\_time -> po\_amount

- a\_is\_system -> a\_owner

- Для отношений гарантируется соответствие данных с одинаковыми ключами.

5) Существуют ли отношения, имеющие многозначные зависимости? Если «да», как можно переработать схему, чтобы избежать таких зависимостей?

Исходная модель:

- Таких нет

Модель после 1-й лабораторной работы:

- Таких нет

6) Нарушает ли схема какие-либо «требования нормализации»? Если «да», доработайте схему, чтобы избежать таких нарушений.

Исходная модель:

- Не нарушает

Модель после 1-й лабораторной работы:

- Не нарушает

7) Существуют ли какие-либо потенциальные проблемы с производительностью базы данных? Если «да», запишите их.

Исходная модель:

- Возможно снижение производительности из-за использования древовидной структуры в отношении site\_pages.

Модель после 1-й лабораторной работы:

- Возможно снижение производительности из-за использования древовидной структуры в отношении site\_page.

8) Для каждого отношения в базе данных определите, в какой нормальной форме оно находится. Запишите ответ.

Исходная модель:

- Все отношения, кроме accounts, находятся во 6НФ

Модель после 1-й лабораторной работы:

- Все отношения находятся во 6НФ

9) Есть ли отношения с возможной, но ненужной дальнейшей нормализацией? Составьте список.

Исходная модель:

- Необходимо произвести нормализацию отношения accounts, в которой a\_is\_system должен быть не связан с a\_owner.

Модель после 1-й лабораторной работы:

- Таких нет

10) Можно ли добиться некоторого повышения производительности за счёт денормализации схемы?

Исходная модель:

- Думаю, что в денормализации схемы нет необходимости, ибо это может привести к разрастанию базы данных и использованию большего количества оборудования, взамен небольшого повышения производительности.

Модель после 1-й лабораторной работы:

- Думаю, что в денормализации схемы нет необходимости, ибо это может привести к разрастанию базы данных и использованию большего количества оборудования, взамен небольшого повышения производительности.

11) Можно ли добиться некоторого повышения производительности, добавив в схему кэширующие отношения? Обоснуйте своё мнение.

Исходная модель:

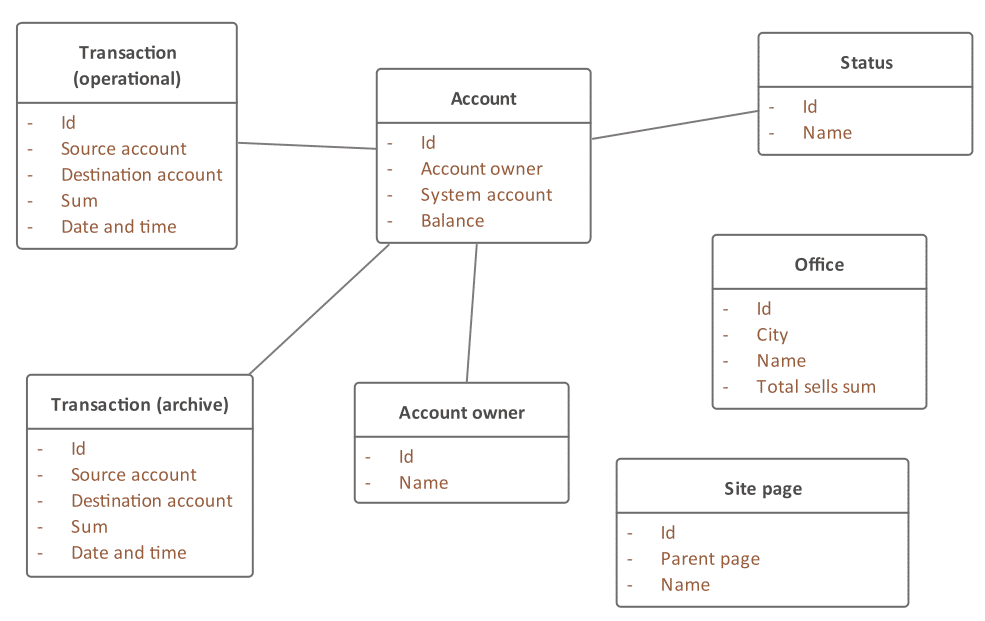
- В схеме присутствует payments\_archive, что является кэширующим отношением для payments\_operational. Для других отношений не вижу в этом смысла, так как данные там не устаревают.

Модель после 1-й лабораторной работы:

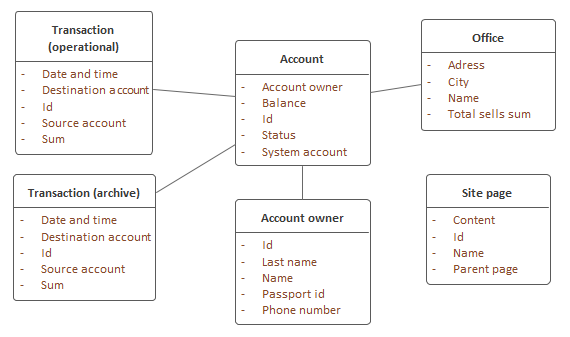
- В схеме присутствует payment\_archive, что является кэширующим отношением для payment\_operational. Для других отношений не вижу в этом смысла, так как данные там не устаревают.

Результат работы:

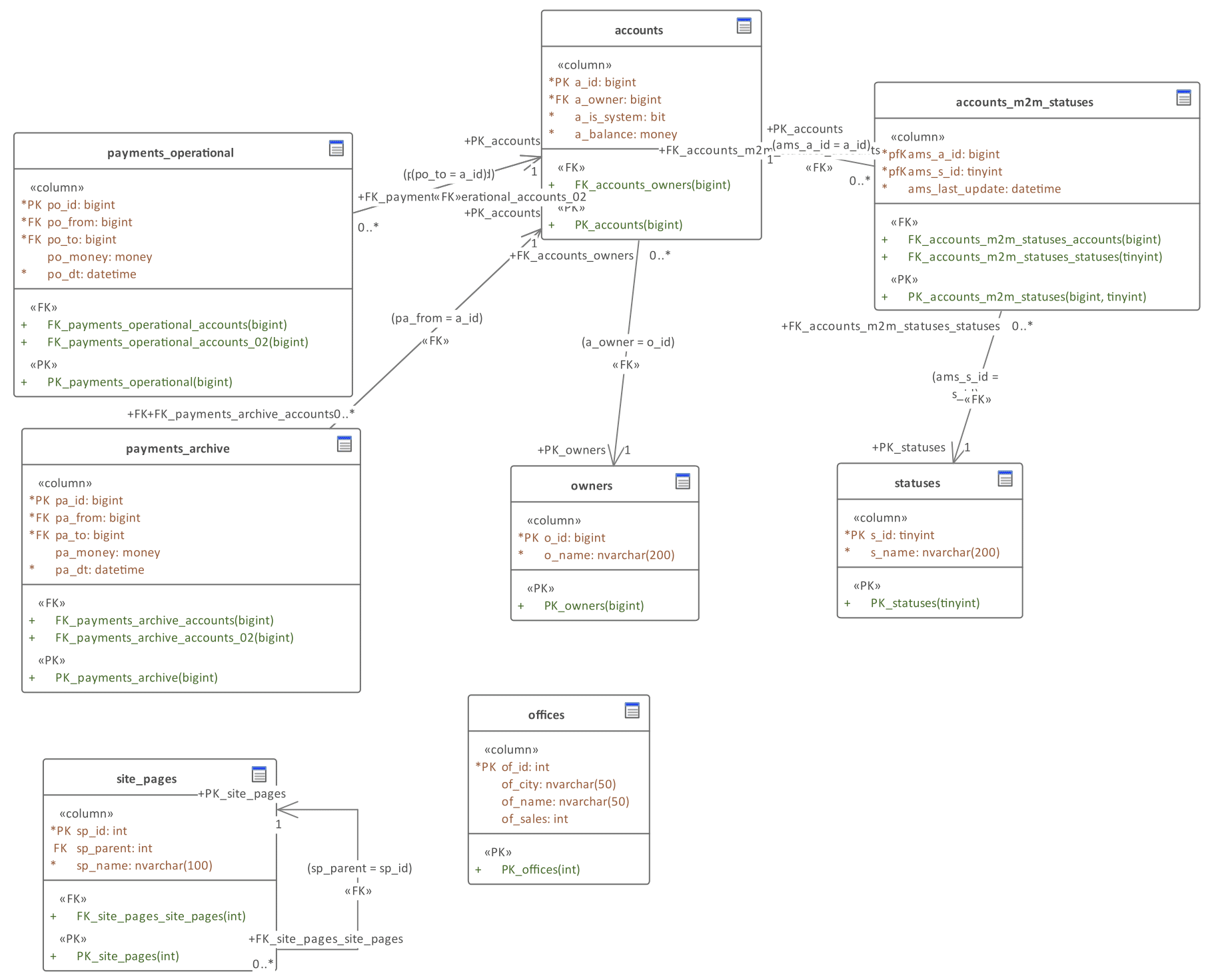
- Исходная инфологическая модель:



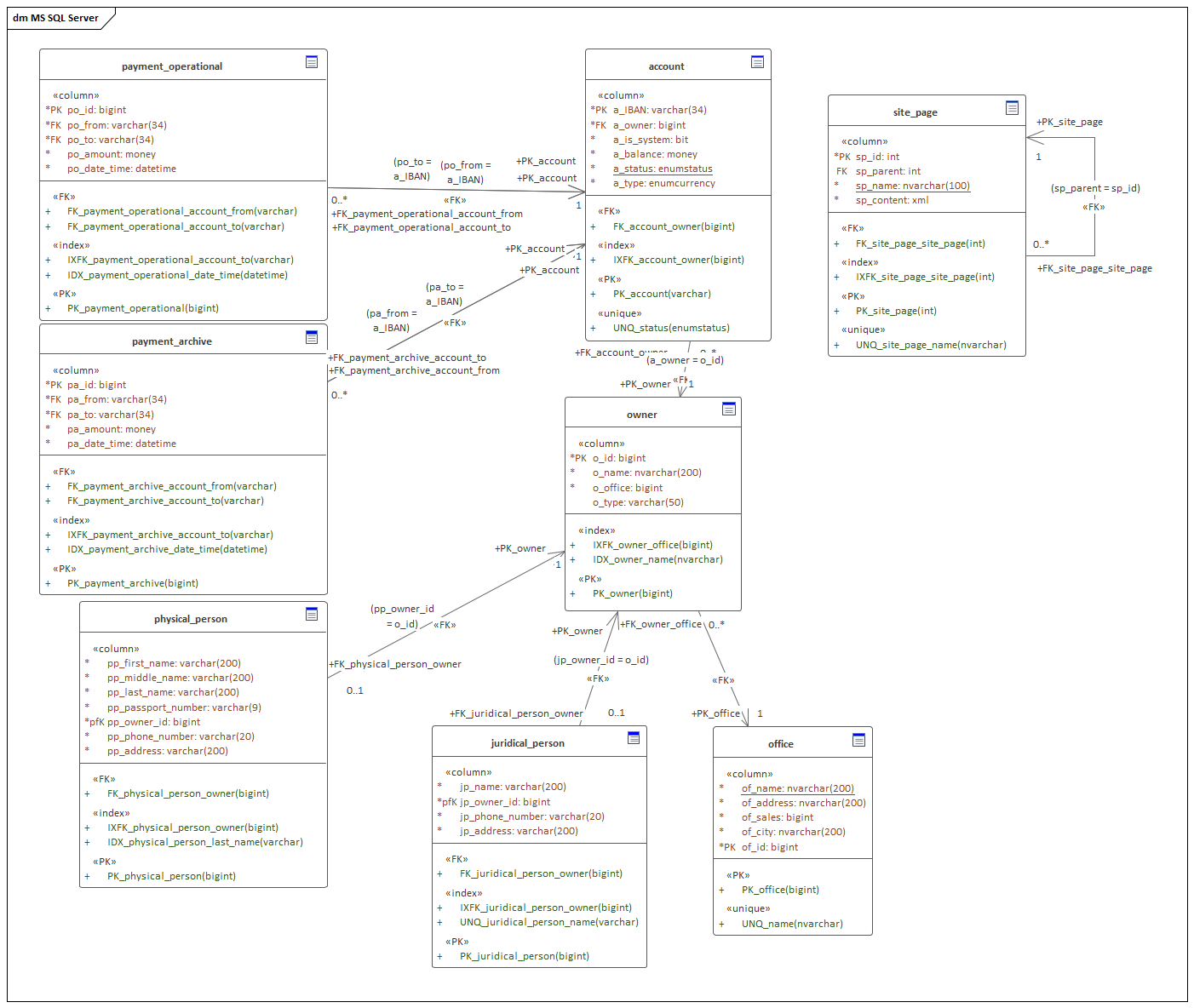
Готовая инфологическая модель:



- Исходная даталогическая модель:



Готовая даталогическая модель:



Сгенерированный SQL код:

/\* ---------------------------------------------------- \*/

/\* Generated by Enterprise Architect Version 15.0 \*/

/\* Created On : 05-март-2021 16:13:59 \*/

/\* DBMS : SQL Server 2012 \*/

/\* ---------------------------------------------------- \*/

/\* Drop Foreign Key Constraints \*/

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_account\_owner]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [account] DROP CONSTRAINT [FK\_account\_owner]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_juridical\_person\_owner]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [juridical\_person] DROP CONSTRAINT [FK\_juridical\_person\_owner]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_payment\_archive\_account\_from]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [payment\_archive] DROP CONSTRAINT [FK\_payment\_archive\_account\_from]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_payment\_archive\_account\_to]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [payment\_archive] DROP CONSTRAINT [FK\_payment\_archive\_account\_to]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_payment\_operational\_account\_from]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [payment\_operational] DROP CONSTRAINT [FK\_payment\_operational\_account\_from]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_payment\_operational\_account\_to]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [payment\_operational] DROP CONSTRAINT [FK\_payment\_operational\_account\_to]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_physical\_person\_owner]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [physical\_person] DROP CONSTRAINT [FK\_physical\_person\_owner]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_site\_page\_site\_page]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [site\_page] DROP CONSTRAINT [FK\_site\_page\_site\_page]

GO

/\* Drop Tables \*/

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[account]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [account]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[juridical\_person]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [juridical\_person]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[office]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [office]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[owner]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [owner]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[payment\_archive]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [payment\_archive]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[payment\_operational]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [payment\_operational]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[physical\_person]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [physical\_person]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[site\_page]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [site\_page]

GO

/\* Create Tables \*/

CREATE TABLE [account]

(

[a\_IBAN] varchar(34) NOT NULL,

[a\_owner] bigint NOT NULL,

[a\_is\_system] bit NOT NULL,

[a\_balance] money NOT NULL,

[a\_status] enumstatus NOT NULL,

[a\_type] enumcurrency NOT NULL

)

GO

CREATE TABLE [juridical\_person]

(

[jp\_name] varchar(200) NOT NULL,

[jp\_owner\_id] bigint NOT NULL,

[jp\_phone\_number] varchar(20) NOT NULL,

[jp\_address] varchar(200) NOT NULL

)

GO

CREATE TABLE [office]

(

[of\_name] nvarchar(200) NOT NULL,

[of\_address] nvarchar(200) NOT NULL,

[of\_sales] bigint NOT NULL,

[of\_city] nvarchar(200) NOT NULL,

[of\_id] bigint NOT NULL IDENTITY (1, 1)

)

GO

CREATE TABLE [owner]

(

[o\_id] bigint NOT NULL IDENTITY (1, 1) NOT FOR REPLICATION,

[o\_name] nvarchar(200) NOT NULL,

[o\_office] bigint NOT NULL,

[o\_type] varchar(50) NULL

)

GO

CREATE TABLE [payment\_archive]

(

[pa\_id] bigint NOT NULL,

[pa\_from] varchar(34) NOT NULL,

[pa\_to] varchar(34) NOT NULL,

[pa\_amount] money NOT NULL,

[pa\_date\_time] datetime NOT NULL

)

GO

CREATE TABLE [payment\_operational]

(

[po\_id] bigint NOT NULL IDENTITY (1, 1),

[po\_from] varchar(34) NOT NULL,

[po\_to] varchar(34) NOT NULL,

[po\_amount] money NOT NULL,

[po\_date\_time] datetime NOT NULL

)

GO

CREATE TABLE [physical\_person]

(

[pp\_first\_name] varchar(200) NOT NULL,

[pp\_middle\_name] varchar(200) NOT NULL,

[pp\_last\_name] varchar(200) NOT NULL,

[pp\_passport\_number] varchar(9) NOT NULL,

[pp\_owner\_id] bigint NOT NULL,

[pp\_phone\_number] varchar(20) NOT NULL,

[pp\_address] varchar(200) NOT NULL

)

GO

CREATE TABLE [site\_page]

(

[sp\_id] int NOT NULL IDENTITY (1, 1),

[sp\_parent] int NULL,

[sp\_name] nvarchar(100) NOT NULL,

[sp\_content] xml NOT NULL

)

GO

/\* Create Primary Keys, Indexes, Uniques, Checks \*/

ALTER TABLE [account]

ADD CONSTRAINT [PK\_account]

PRIMARY KEY CLUSTERED ([a\_IBAN] ASC)

GO

ALTER TABLE [account]

ADD CONSTRAINT [UNQ\_status] UNIQUE NONCLUSTERED ([a\_status] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_account\_owner]

ON [account] ([a\_owner] ASC)

GO

ALTER TABLE [juridical\_person]

ADD CONSTRAINT [PK\_juridical\_person]

PRIMARY KEY CLUSTERED ([jp\_owner\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_juridical\_person\_owner]

ON [juridical\_person] ([jp\_owner\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [UNQ\_juridical\_person\_name]

ON [juridical\_person] ([jp\_name] ASC)

GO

ALTER TABLE [office]

ADD CONSTRAINT [PK\_office]

PRIMARY KEY CLUSTERED ([of\_id] ASC)

GO

ALTER TABLE [office]

ADD CONSTRAINT [UNQ\_name] UNIQUE NONCLUSTERED ([of\_name] ASC)

GO

ALTER TABLE [owner]

ADD CONSTRAINT [PK\_owner]

PRIMARY KEY CLUSTERED ([o\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_owner\_office]

ON [owner] ([o\_office] ASC)

GO

CREATE NONCLUSTERED INDEX [IDX\_owner\_name]

ON [owner] ([o\_name] ASC)

GO

ALTER TABLE [payment\_archive]

ADD CONSTRAINT [PK\_payment\_archive]

PRIMARY KEY CLUSTERED ([pa\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_payment\_archive\_account\_to]

ON [payment\_archive] ([pa\_to] ASC)

GO

CREATE NONCLUSTERED INDEX [IDX\_payment\_archive\_date\_time]

ON [payment\_archive] ([pa\_date\_time] ASC)

GO

ALTER TABLE [payment\_operational]

ADD CONSTRAINT [PK\_payment\_operational]

PRIMARY KEY CLUSTERED ([po\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_payment\_operational\_account\_to]

ON [payment\_operational] ([po\_to] ASC)

GO

CREATE NONCLUSTERED INDEX [IDX\_payment\_operational\_date\_time]

ON [payment\_operational] ([po\_date\_time] ASC)

GO

ALTER TABLE [physical\_person]

ADD CONSTRAINT [PK\_physical\_person]

PRIMARY KEY CLUSTERED ([pp\_owner\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_physical\_person\_owner]

ON [physical\_person] ([pp\_owner\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IDX\_physical\_person\_last\_name]

ON [physical\_person] ([pp\_last\_name] ASC)

GO

ALTER TABLE [site\_page]

ADD CONSTRAINT [PK\_site\_page]

PRIMARY KEY CLUSTERED ([sp\_id] ASC)

GO

ALTER TABLE [site\_page]

ADD CONSTRAINT [UNQ\_site\_page\_name] UNIQUE NONCLUSTERED ([sp\_name] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_site\_page\_site\_page]

ON [site\_page] ([sp\_parent] ASC)

GO

/\* Create Foreign Key Constraints \*/

ALTER TABLE [account] ADD CONSTRAINT [FK\_account\_owner]

FOREIGN KEY ([a\_owner]) REFERENCES [owner] ([o\_id]) ON DELETE No Action ON UPDATE Cascade

GO

ALTER TABLE [juridical\_person] ADD CONSTRAINT [FK\_juridical\_person\_owner]

FOREIGN KEY ([jp\_owner\_id]) REFERENCES [owner] ([o\_id]) ON DELETE No Action ON UPDATE Cascade

GO

ALTER TABLE [payment\_archive] ADD CONSTRAINT [FK\_payment\_archive\_account\_from]

FOREIGN KEY ([pa\_from]) REFERENCES [account] ([a\_IBAN]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [payment\_archive] ADD CONSTRAINT [FK\_payment\_archive\_account\_to]

FOREIGN KEY ([pa\_to]) REFERENCES [account] ([a\_IBAN]) ON DELETE No Action ON UPDATE Cascade

GO

ALTER TABLE [payment\_operational] ADD CONSTRAINT [FK\_payment\_operational\_account\_from]

FOREIGN KEY ([po\_from]) REFERENCES [account] ([a\_IBAN]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [payment\_operational] ADD CONSTRAINT [FK\_payment\_operational\_account\_to]

FOREIGN KEY ([po\_to]) REFERENCES [account] ([a\_IBAN]) ON DELETE No Action ON UPDATE Cascade

GO

ALTER TABLE [physical\_person] ADD CONSTRAINT [FK\_physical\_person\_owner]

FOREIGN KEY ([pp\_owner\_id]) REFERENCES [owner] ([o\_id]) ON DELETE No Action ON UPDATE Cascade

GO

ALTER TABLE [site\_page] ADD CONSTRAINT [FK\_site\_page\_site\_page]

FOREIGN KEY ([sp\_parent]) REFERENCES [site\_page] ([sp\_id]) ON DELETE Set Null ON UPDATE Cascade

GO

/\* Create Table Comments \*/

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'account', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Содержит информацию о счете в банке', 'SCHEMA', 'dbo', 'table', 'account'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Содержит информацию о счете в банке', 'SCHEMA', 'dbo', 'table', 'account'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'juridical\_person', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Подробная информация о конкретном юридическом лице в виде названия предприятия, контактного телефона, адреса главного офиса', 'SCHEMA', 'dbo', 'table', 'juridical\_person'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Подробная информация о конкретном юридическом лице в виде названия предприятия, контактного телефона, адреса главного офиса', 'SCHEMA', 'dbo', 'table', 'juridical\_person'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'office', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Информация об офисах банка с их адресами, размерами продаж, городами расположения и названиями отделений.', 'SCHEMA', 'dbo', 'table', 'office'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Информация об офисах банка с их адресами, размерами продаж, городами расположения и названиями отделений.', 'SCHEMA', 'dbo', 'table', 'office'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'owner', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Описание владельца счета, включающее в себя офис, где был открыт счет, тип лица (физическое или юридическое), имя', 'SCHEMA', 'dbo', 'table', 'owner'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Описание владельца счета, включающее в себя офис, где был открыт счет, тип лица (физическое или юридическое), имя', 'SCHEMA', 'dbo', 'table', 'owner'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'payment\_archive', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Транзакции, произошедшие более 1 месяца назад. Редко используются, но должны сохраняться', 'SCHEMA', 'dbo', 'table', 'payment\_archive'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Транзакции, произошедшие более 1 месяца назад. Редко используются, но должны сохраняться', 'SCHEMA', 'dbo', 'table', 'payment\_archive'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'payment\_operational', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Информация о платежах, проведённых в течение 1 месяца с последнего. Необходима для быстрого и частого доступа к последним транзакциям', 'SCHEMA', 'dbo', 'table', 'payment\_operational'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Информация о платежах, проведённых в течение 1 месяца с последнего. Необходима для быстрого и частого доступа к последним транзакциям', 'SCHEMA', 'dbo', 'table', 'payment\_operational'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'physical\_person', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Подробная информация о конкретном физическом лице, включая адрес проживания, телефонный номер, ФИО, номер паспорта', 'SCHEMA', 'dbo', 'table', 'physical\_person'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Подробная информация о конкретном физическом лице, включая адрес проживания, телефонный номер, ФИО, номер паспорта', 'SCHEMA', 'dbo', 'table', 'physical\_person'

end

GO

if exists (select \* from ::fn\_listextendedproperty ('MS\_Description', 'SCHEMA', 'dbo', 'table', 'site\_page', NULL, NULL))

begin

EXEC sys.sp\_updateextendedproperty 'MS\_Description', 'Древовидная структура, содержащая в себе страницы сайта банка с их названиями ии содержанием', 'SCHEMA', 'dbo', 'table', 'site\_page'

end

else

begin

EXEC sys.sp\_addextendedproperty 'MS\_Description', 'Древовидная структура, содержащая в себе страницы сайта банка с их названиями ии содержанием', 'SCHEMA', 'dbo', 'table', 'site\_page'

end

GO