МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»

Кафедра информационных систем

Курсовая работа

по дисциплине: «Управление данными»

на тему**:** Проектирование базы данных

“Магазин предметов для многопользовательской игры”.

Выполнил: студент 3 курса бакалавриата направления 09.03.02 – "Информационные системы и технологии"
Першин С.В.

Проверил: доц.каф. ИС
Каширина Е.В.

Санкт-Петербург
2020

Оглавление

Введение

Целью данной курсовой работы является проектирование базы данных для предметной области “Магазин предметов для многопользовательской игры”. Данная предметная область включает в себя деятельность внутриигрового магазина, обеспечивающего продажу и покупку внутриигровых вещей.

Большое количество многопользовательских игр имеют механизм получения пользователями предметов. Зачастую разработчики используют систему внутриигрового магазина, который дает возможность пользователям покупать и продавать внутриигровые предметы, необходимые для игры.

В многопользовательской игре существуют предметы, доступные для покупки и продажи пользователями игры. Некоторые предметы имеют постоянный характер, то есть их можно иметь только в одном экземпляре. Также существуют предметы, имеющие временный характер, то есть пользователь может приобретать и продавать их неограниченное количество раз. Таким образом необходимо обеспечить механизм проверки наличия у пользователя предметов, а также вести учет купленных игроком вещей, для проверки возможности продажи.

При выполнении курсовой работы будут реализованы следующие задачи:

- · Изучение процессов, происходящих в предметной области.
- · Проектирование БД для решения задач пользователей.

Часть 1. Описание предметной области “Магазин предметов для многопользовательской игры”

Магазин предметов решает задачу получения пользователями предметов, а также дает возможность получения внутриигровой валюты путем их продажи.

Для совершения операций во внутриигровом магазине пользователь должен быть зарегистрирован в игре и иметь внутриигрового персонажа. Магазин учитывает количество внутриигровой валюты персонажа и имеющиеся у него предметы.

Предметы в магазине делятся на две категории: временные и постоянные. Временные предметы — это предметы, которые пользователь может купить неограниченное количество раз. Такие предметы являются расходуемыми и после использования пропадают. Временные предметы можно продать. Постоянные предметы — это предметы, которые пользователь может иметь в единственном экземпляре.

Каждый предмет имеет характеристики: “ID предмета” (уникальный идентификатор предмета), “Название”, “Цена” (количество валюты, необходимое для покупки данного предмета, или количество валюты, получаемое при его продаже), “Описание”, “Тип предмета” (временный или постоянный).

Предметы могут добавляться во внутриигровой магазин. После добавления предметы становятся доступны к покупке персонажами. В магазине не может быть несколько предметов с одинаковыми атрибутами.

Для осуществления контроля за покупкой и продажей предметов магазин имеет историю операций. Она содержит порядковый номер операции, тип операции (покупка или продажа), имя персонажа, совершившего операцию, дату операции, уникальный идентификатор предмета, его цену и тип операции. Пользователь, при желании, может просмотреть свои операции, совершенные персонажами.

Механизм работы внутриигрового магазина следующий: выводится список доступных предметов, затем пользователь выбирает нужный предмет, магазин производит проверку возможности покупки или продажи данного предмета пользователем, после чего пользователь, если проверка пройдена успешно, покупает или продает этот предмет, далее с баланса внутриигровой валюты персонажа списывается или поступает цена предмета, в конце в историю записывается информация об операции.

1.1 Описание процессов, происходящих в предметной области, информацию о которых надо хранить в БД

- Добавление пользователя
- Добавление персонажа
- Изменение баланса персонажа
- Проверка баланса персонажа
- Изменение количества наименований и числа предметов персонажа
- Проверка количества наименований и числа предметов персонажа
- Добавление операций в историю операций магазина

1.2 Пользователи информационной системы, для которой проектируется БД

1) Клиенты (игроки многопользовательской игры)
2) Сотрудники управляющие магазином

1.3 Задачи пользователей, которые должны быть решены средствами информационной системы

Система должна решать такие задачи сотрудников: хранение информации об аккаунте пользователя; хранение информации о персонажах на аккаунте пользователя; ведение учета совершенных операций клиентов.

Система должна решать такие задачи клиентов: покупка и продажа внутриигровых предметов; просмотр истории операций; просмотр имеющихся предметов персонажа; просмотр остатка валюты на балансе персонажа; расчет общей стоимости предметов персонажа.

1.4 Методы решения задач пользователей

Клиент, игрок многопользовательской игры проходит процесс регистрации и создания персонажа. Игрок может создавать несколько персонажей, которые используются независимо друг от друга. После создания первого персонажа клиенту открывается доступ в магазин. На заработанную в процессе игры валюту он может приобретать предметы, расположенные в каталоге магазина. При желании игрок может видеть предметы из каталога, которые он может себе позволить на имеющуюся у него валюту. Для этого производится отбор позиций, стоимость которых меньше имеющейся у пользователя внутриигровой валюты. После, для каждого предмета рассчитывается их количество, которое может позволить себе клиент. Также у игрока есть доступ к своему инвентарю, в котором находятся все приобретенные им предметы из каталога. Клиент может продавать предметы из инвентаря. И, кроме того, он может видеть суммарную стоимость предметов, находящихся в инвентаре персонажа.

В магазине присутствует история операций. Игрок может просматривать историю операций любого своего персонажа. Для эффективного использования магазина в историю записываются дата совершенных операций и тип операции.

По желанию клиент может удалять своих персонажей, но обязательно должен оставаться хотя бы один персонаж на аккаунте пользователя. Без наличия персонажей пользоваться магазином нельзя. После удаления последнего персонажа на аккаунте весь аккаунт пользователя удаляется.

Сотрудники магазина — это модераторы, которые следят за работой магазина. Они могут свободно просматривать всю историю операций, а также менять ассортимент, представленный в каталоге.

1.5 Пользовательские ограничения предметной области

- Одним и тем же персонажем не могут владеть несколько пользователей
- Пользователь магазина не может приобрести товар, цена которого больше, чем имеющееся количество валюты у персонажа пользователя.

Часть 2. ER-модель предметной области в стандарте IDEF1X

2.1 Смысловое описание сущностей с указанием идентификаторов каждой сущности

Таблица 1. Описание сущностей.

Название сущности	Описание
Пользователи	Список пользователей с дополнительной информацией о них

Персонажи	Список персонажей, принадлежащих пользователям
Каталог	Список всех предметов, продающихся на данный момент в магазине
История операций	История всех произведенных пользователями операций в магазине
Приобретенные предметы	Информация о имеющихся у персонажей предметах и их количестве

2.2 Описание атрибутов каждой сущности, её ключей, описание доменов, которым принадлежат атрибуты сущностей, с указанием ограничений на возможные значения

Пользователи:

Таблица 1.1.1 Ключ сущности “Пользователи”

Ключ	Описание	Домен	Ограничение домена	Наличие данных	Пример
ИД_Пользователя (первичный)	Идентификационный номер пользователя	Строка	Строго 9 символов (буквы и цифры)	Обязательное указание	USRABC001

Таблица 1.1.2 Атрибуты сущности “Пользователи”

Атрибуты сущности	Описание атрибута	Домен	Ограничение домена	Наличие данных	Пример
Электронная_почта	Электронная почта пользователя магазина	Строка	Не больше 35 символов (буквы и цифры)	Обязательное указание	abc123@mail.com
Имя_пользователя	Имя пользователя магазина	Строка	Не больше 20 символов (буквы и цифры)	Обязательное указание	abc123

Персонажи:

Таблица 1.2.1 Ключи сущности “Персонажи”

Ключ	Описание	Домен	Ограничение домена	Наличие данных	Пример
ИД_персонажа (первичный)	Идентификационный номер персонажа	Строка	Строго 10 символов (буквы и цифры)	Обязательное указание	CHRABC0011
ИД_пользователя (внешний)	Идентификационный номер пользователя	Строка	Строго 9 символов (буквы и цифры)	Обязательное указание	USRABC001

Таблица 1.2.2 Атрибуты сущности “Персонажи”

Атрибуты сущности	Описание атрибута	Домен	Ограничение домена	Наличие данных	Пример
Имя_персонажа	Название персонажа пользователя	Строка	Не больше 20 символов (буквы и цифры)	Обязательное указание	Abc123
Количество_валюты	Количество валюты, имеющееся у персонажа	Число	Не может быть отрицательным	Обязательное указание	15

Каталог:

Таблица 1.3.1 Ключи сущности “Каталог”

Ключ	Описание	Домен	Ограничение домена	Наличие данных	Пример
ИД_Предмета (первичный)	Идентификационный номер предмета в магазине	Строка	Строго 5 символов (буквы и цифры)	Обязательное указание	ITAB1

Цена (первичный)	Стоимость предмета в магазине	Число	Не может быть отрицательным	Обязательное указание	10

Таблица 1.3.2 Атрибуты сущности “Каталог”

Атрибуты сущности	Описание атрибута	Домен	Ограничение домена	Наличие данных	Пример
Название	Название предмета	Строка	Не больше 20 символов (буквы и цифры)	Обязательное указание	Abc123
Описание	Описание предмета	Строка	Не больше 80 символов (буквы и цифры)	Необязательное указание	15
Тип_предмета	Тип предмета (временный или постоянный)	Строка	Не больше 10 символов (буквы)	Обязательное указание	permanent

История операций:

Таблица 1.4.1 Ключи сущности “История операций”

Ключ	Описание	Домен	Ограничение домена	Наличие данных	Пример
Номер_операции (первичный)	Порядковый номер операции	Число	Не может быть отрицательным	Обязательное указание	11621
ИД_персонажа (внешний)	Идентификационный номер персонажа	Строка	Строго 10 символов (буквы и цифры)	Обязательное указание	CHRABC0011
ИД_пользова	Идентифика	Строка	Строго 9	Обязательное	USRABC001

теля (внешний)	ционный номер пользователя		символов (буквы и цифры)	указание	
ИД_предмета (внешний)	Идентификационный номер предмета в магазине	Идентификационный номер предмета в магазине	Строго 5 символов (буквы и цифры)	Обязательное указание	ITAB1
Цена (внешний)	Стоимость предмета в магазине	Число	Не может быть отрицательным	Обязательное указание	10

Таблица 1.4.2 Атрибуты сущности “История операций”

Атрибуты сущности	Описание атрибута	Домен	Ограничение домена	Наличие данных	Пример
Дата	Дата совершения операции в магазине	ДатаВремя	Дата не может быть ранее открытия магазина	Обязательное указание	Abc123
Тип_операции	Описание типа операции (покупка или продажа)	Строка	Не больше 6 символов (буквы)	Обязательное указание	BUY

Приобретенные предметы:

Таблица 1.5.1 Ключи сущности “Приобретенные предметы”

Ключ	Описание	Домен	Ограничение домена	Наличие данных	Пример
ИД_ячейки	Идентификационный номер ячейки инвентаря персонажа занятого	Строка	Строго 5 символов (буквы и цифры)	Обязательное указание	I0001

	предметом				
ИД_персонажа (внешний)	Идентификационный номер персонажа	Строка	Строго 10 символов (буквы и цифры)	Обязательное указание	CHRABC0011
ИД_пользователя (внешний)	Идентификационный номер пользователя	Строка	Строго 9 символов (буквы и цифры)	Обязательное указание	USRABC001
ИД_предмета (внешний)	Идентификационный номер предмета в магазине	Идентификационный номер предмета в магазине	Строго 5 символов (буквы и цифры)	Обязательное указание	ITAB1
Цена (внешний)	Стоимость предмета в магазине	Число	Не может быть отрицательным	Обязательное указание	10

Таблица 1.5.2 Атрибуты сущности “Приобретенные предметы”

Атрибуты сущности	Описание атрибута	Домен	Ограничение домена	Наличие данных	Пример
Количество	Количество имеющихся у пользователя предметов	Число	Не может быть отрицательным	Обязательное указание	2

2.3 Описание типов связей между сущностями и кардинальности связей между сущностями

Таблица 2. Типы связей и кардинальности между сущностями.

Сущности		Связь	Кардинальность
Родитель	Потомок		
Пользователи	Персонажи	Идентифицирующая принадлежности	Один ко многим

Персонажи	Приобретенные предметы	Идентифицирующая принадлежности	Один ко многим
Персонажи	История операций	Не идентифицирующая принадлежности	
Каталог	История операций	Не идентифицирующая принадлежности	
Каталог	Приобретенные предметы	Не идентифицирующая принадлежности	

2.4 ER-диаграмма предметной области

Приведем логическую диаграмму предметной области, созданную в программном обеспечении Erwin (Рис.1):

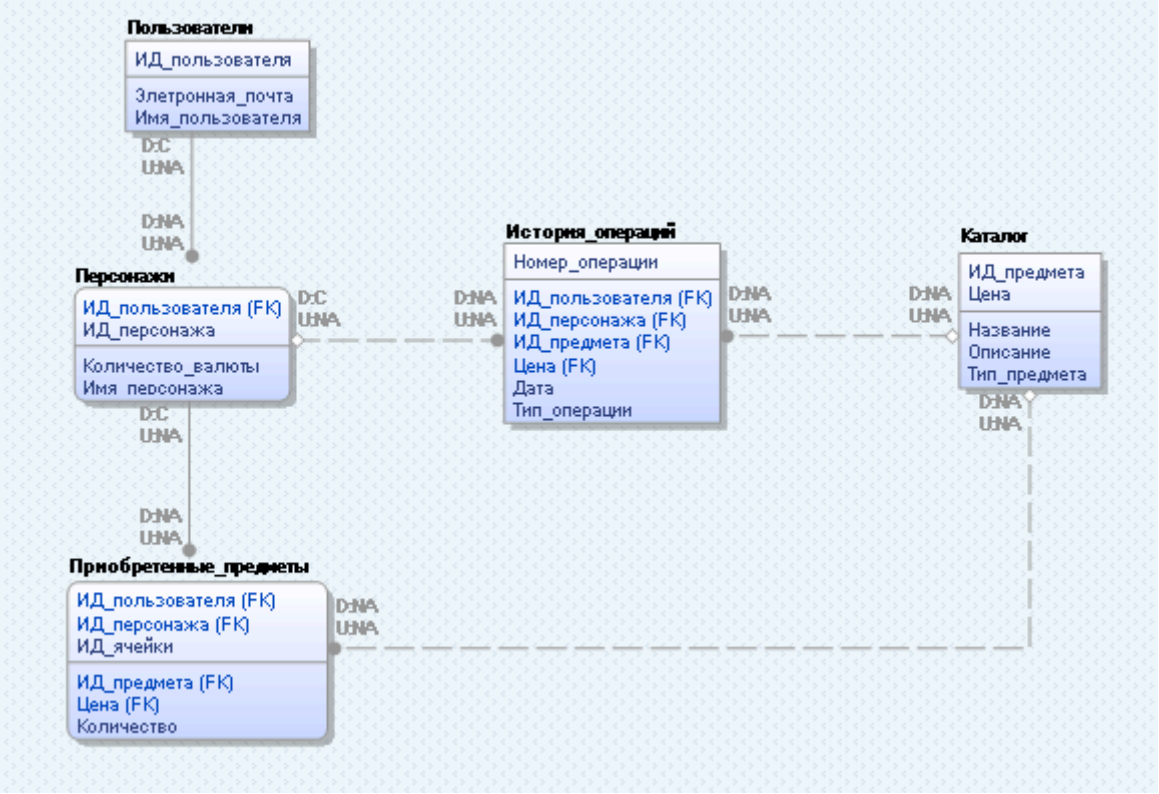

Рисунок 1. Логическая диаграмма предметной области с отображением ссылочной целостности и кардинальности

2.5 Соответствие ER-диаграммы задачам пользователей и ограничениям, заявленным в пунктах 1.1, 1.4, 1.5

Представленная ER-диаграмма отвечает задачам пользователей и ограничениям, описанным в первой части работы.

2.6 Описание дополнительных ограничений сущностей, которые не могут быть выражены через описание доменов, но могут быть выражены через ограничение сущности

- Имя пользователя и Электронная почта сущности Пользователи должны быть уникальными.
- Имя персонажа сущности Персонажи должно быть уникальным.
- Тип предмета сущности каталог должен быть либо “permanent”, либо “temporary”.
- Тип операции сущности История операций должен быть или “BUY”, или “SELL”.

2.7 Описание дополнительных ограничений, которые не могут быть выражены средствами ER-модели предметной области

Дополнительными ограничениями сущностей, которые не могут быть выражены через описание доменов, но могут быть выражены через ограничение сущности, являются в виде такого условия: “Тип операции” в сущности “История операций” не может быть “SELL” для предмета до того, как этот предмет будет приобретен.

2.8 Описание задач, для решения которых необходимы представления

Задачами, для которых нужны представления, являются:

- Перечисление имен всех персонажей, которые приобрели предметы являющиеся постоянными, с указанием этих предметов и их стоимости
- Перечисление всех персонажей и имеющейся у них валюты

2.9 Описание задач, для решения которых необходимы триггеры и хранимые процедуры

Задача, для решения которой необходимы триггеры является:

- После удаления всех персонажей пользователя, информация о пользователе удаляется из базы данных

Задачи, для решения которых необходимы хранимые процедуры, являются:

- Вывод общей стоимости предметов конкретного персонажа
- Вывод всех предметов, доступных для покупки персонажу и их количество, которое может позволить персонаж
- Вывод истории операций совершенных конкретным персонажем

- Приобретение предметов из каталога, с последующим добавлением данной операции в историю операций и изменением остатка валюты персонажа
- Продажа предметов из каталога с последующим добавлением данной операции в историю операций и изменением остатка валюты персонажа

2.10 Описание необходимых транзакций

Необходимые транзакции:

- Проверка, что пользователь имеет достаточное количество валюты, до того, как приобрести желаемый предмет
- Проверка, что пользователю не принадлежит такой предмет типа “Постоянный”, как уже имеющийся у него, до того, как приобрести этот предмет
- Проверка что пользователю принадлежит предмет, до его продажи

Часть 3. Реляционная модель базы данных

3.1 Полное название СУБД, средствами которой создается БД

База данных “InGameShop” была создана в СУБД Microsoft SQL Server 2019.

3.2 Набор команд SQL для создания структуры БД в указанной СУБД

```
CREATE DATABASE InGameShop
go
USE InGameShop
CREATE TABLE [Пользователи]
(
	[Электронная_почта]  varchar(35)  NOT NULL ,
	[ИД_пользователя]    char(9)  NOT NULL ,
	[Имя_пользователя]   varchar(20)  NOT NULL ,
	CONSTRAINT [AK_Имя_пользователя] UNIQUE([Имя_пользователя]) ,
	CONSTRAINT [AK_Электронная_почта] UNIQUE([Электронная_почта]) ,
	CONSTRAINT [XPKПользователи] PRIMARY KEY ([ИД_пользователя])
)
go

CREATE TABLE [Персонажи]
(
	[ИД_пользователя]    char(9)  NOT NULL ,
```

```
	[ИД_персонажа]	char(10) NOT NULL ,
	[Количество_валюты] integer NOT NULL , CHECK([Количество_валюты]>=0),
	[Имя_персонажа]	varchar(20) NOT NULL ,
	CONSTRAINT [AK_Имя_персонажа] UNIQUE([Имя_персонажа]) ,
	CONSTRAINT [XPKПерсонажи] PRIMARY KEY
([ИД_пользователя],[ИД_персонажа]) ,
	CONSTRAINT [Пользователи_Персонажи] FOREIGN KEY ([ИД_пользователя])
REFERENCES [Пользователи]([ИД_пользователя])
)
go

CREATE TABLE [Каталог]
(
	[ИД_предмета]	char(5) NOT NULL ,
	[Название]	varchar(20) NOT NULL ,
	[Цена]	integer NOT NULL , CHECK([Цена]>=0),
	[Описание]	varchar(80) NULL ,
	[Тип_предмета]	varchar(10) NOT NULL ,
CHECK([Тип_предмета]='permanent' or [Тип_предмета]='temporary'),
	CONSTRAINT [AK_ИД_предмета] UNIQUE([ИД_предмета]) ,
	CONSTRAINT [XPKКаталог] PRIMARY KEY ([ИД_предмета],[Цена])
)
go

CREATE TABLE [История_операций]
(
	[ИД_пользователя] char(9) NOT NULL ,
	[ИД_персонажа]	char(10) NOT NULL ,
	[ИД_предмета]	char(5) NOT NULL ,
	[Цена]	integer NOT NULL , CHECK([Цена]>=0),
	[Номер_операции] integer NOT NULL , CHECK([Номер_операции]>=0),
	[Дата]	datetime NOT NULL ,
	[Тип_операции]	varchar(6) NOT NULL , CHECK([Тип_операции]='BUY' or
[Тип_операции]='SELL'),
```

```
	CONSTRAINT [AK_Номер_операции] UNIQUE([Номер_операции]) ,
	CONSTRAINT [XPKИстория_операций] PRIMARY KEY ([Номер_операции]) ,
	CONSTRAINT [Персонажи__История_операций] FOREIGN KEY
([ИД_пользователя],[ИД_персонажа]) REFERENCES
[Персонажи]([ИД_пользователя],[ИД_персонажа])
	  ON DELETE CASCADE,
	CONSTRAINT [Каталог__История_операций] FOREIGN KEY ([ИД_предмета],[Цена])
REFERENCES [Каталог]([ИД_предмета],[Цена])
)
go

CREATE TABLE [Приобретенные_предметы]
(
	[ИД_ячейки]	varchar(5)  NOT NULL,
	[ИД_пользователя]	char(9)  NOT NULL ,
	[ИД_персонажа]	char(10)  NOT NULL ,
	[ИД_предмета]	char(5)  NOT NULL ,
	[Цена]	integer  NOT NULL , CHECK([Цена]>=0),
	[Количество]	integer  NOT NULL , CHECK([Количество]>=1),
	CONSTRAINT [XPKПриобретенные_предметы] PRIMARY KEY
([ИД_ячейки],[ИД_пользователя],[ИД_персонажа]) ,
	CONSTRAINT [Каталог__Приобретенные_предметы] FOREIGN KEY
([ИД_предмета],[Цена]) REFERENCES [Каталог]([ИД_предмета],[Цена]) ,
	CONSTRAINT [Персонажи__Приобретенные_предметы] FOREIGN KEY
([ИД_пользователя],[ИД_персонажа]) REFERENCES
[Персонажи]([ИД_пользователя],[ИД_персонажа])
	  ON DELETE CASCADE
)
go
```

3.3 Команды создания индексов

```
CREATE INDEX Каталог_Инд ON Каталог([Название], [Тип_предмета])
CREATE INDEX Персонажи_Инд ON Персонажи([Имя_персонажа],[Количество_валюты])
CREATE INDEX История_операций_Инд ON История_операций([Номер_операции])
```

3.4 Выполнение листинга создания базы данных

Листинг выполнен. Результат создания БД представлен в виде схемы на Рис.2.

3.5 Схема базы данных с указанием внешних ключей и их свойств

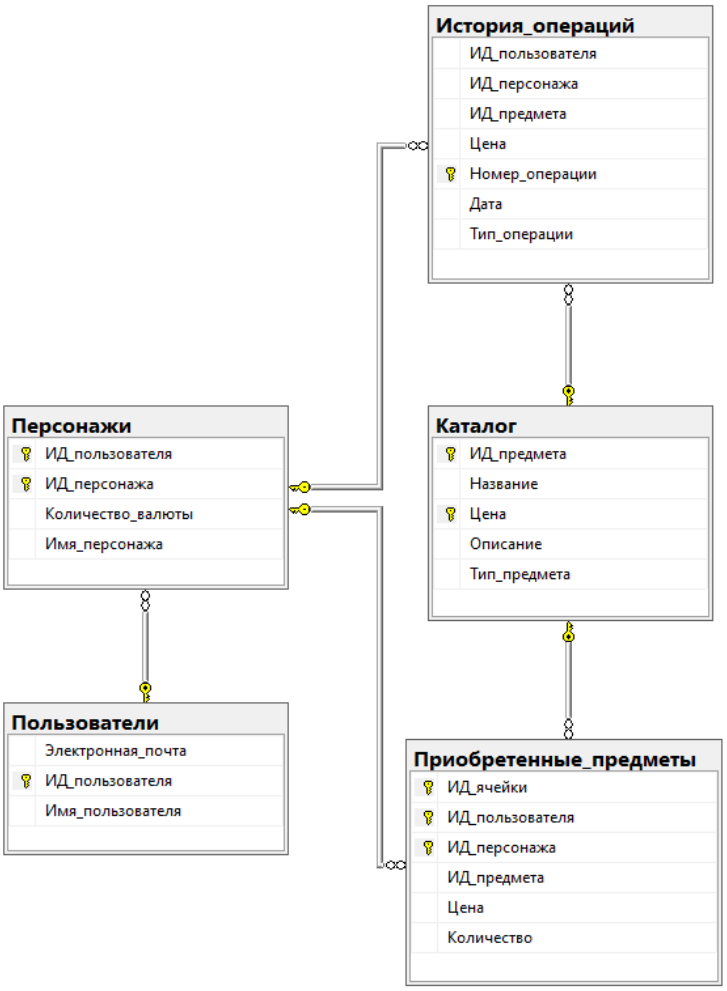

Рисунок 2. Схема базы данных

Часть 4. Ввод данных и выполнение типовых запросов

4.1 Набор команд SQL для ввода данных

```
use InGameShop
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA001', 'crovausseddiva-1840@gmail.com', 'crovausseddiva-1840')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA002', 'rallilenneike-3242@mail.ru', 'rallilenneike-3242')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA003', 'liveuyoinexoi-7956@mail.ru', 'liveuyoinexoi-7956')
```

```
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA004', 'prozecrizoddoi-6456@mail.ru', 'prozecrizoddoi-6456')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA005', 'dalleppummoisa-1776@gmail.com', 'dalleppummoisa-1776')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA006', 'souffauyomminno-1379@gmail.com', 'souffauyomminno-1379')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA007', 'crotteipuqueigu-8275@hotmail.com', 'crotteipuqueigu-8275')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA008', 'baxaxeivagou-6038@gmail.com', 'baxaxeivagou-6038')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA009', 'bettecebicra-9831@mail.ru', 'bettecebicra-9831')
insert into Пользователи
(ИД_пользователя, Электронная_почта, Имя_пользователя)
values('USRAAA010', 'vagounnutrixa-8978@hotmail.com', 'vagounnutrixa-8978')
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA001','CHRAAA0011','Raylee Crutchley', 23)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA001','CHRAAA0012','Rudy Whitley', 105)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA002','CHRAAA0021','Caden Rutland', 230)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA003','CHRAAA0031','Jess Coombs', 2)
```

```
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA003','CHRAAA0032','Jamie Smithies', 133)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA004','CHRAAA0041','Charlie Harley', 76)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA005','CHRAAA0051','Casey Smithy', 54)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA006','CHRAAA0061','Maddox Breeden', 19)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA007','CHRAAA0071','River Leighton', 216)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA008','CHRAAA0081','Cameron Hallewell', 284)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA008','CHRAAA0082','Ashley Peyton', 96)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA009','CHRAAA0091','Brynn Abram', 42)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA009','CHRAAA0092','Billie Mitchell', 11)
insert into Персонажи
(ИД_пользователя, ИД_персонажа, Имя_персонажа, Количество_валюты)
values('USRAAA010','CHRAAA0101','Marley Presley', 182)
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA1','Evil Potion', 25, 'Just an evil potion', 'temporary')
```

```
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA2','Mana Potion', 30, 'Just a mana potion', 'temporary')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA3','Health Potion', 30, 'Just a health potion', 'temporary')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITBA1','Fire Sword', 100, 'Sword with a lighter', 'permanent')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA4','Bottle of water', 5, 'Just bottle with water in it', 'temporary')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITBA2','Boots of speed', 100, 'You must be fast enough to catch them', 'permanent')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA5','Bread', 5, 'Simple bread made from wheat', 'temporary')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA6','Branch', 3, 'From tree', 'temporary')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITAA7','Cookie', 4, 'Om-nom-nom', 'temporary')
insert into Каталог
(ИД_предмета, Название, Цена, Описание, Тип_предмета)
values('ITBA3','Magic Staff', 200, 'It is staff and it has magic in it ', 'permanent')
insert into Приобретенные_предметы
(ИД_ячейки, ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Количество)
values('I0001','USRAAA003','CHRAAA0031','ITBA3',200, 1)
insert into Приобретенные_предметы
(ИД_ячейки, ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Количество)
values('I0001','USRAAA004','CHRAAA0041','ITAA4',5, 2)
```

```
insert into Приобретенные_предметы
(ИД_ячейки, ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Количество)
values('I0001','USRAAA008','CHRAAA0082','ITAA7',4, 1)
insert into Приобретенные_предметы
(ИД_ячейки, ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Количество)
values('I0001','USRAAA010','CHRAAA0101','ITBA1',100, 1)
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA003','CHRAAA0031','ITBA3',200,1,'2000-15-12 00:00:00.000', 'BUY')
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA004','CHRAAA0041','ITAA4',5,2,'2000-16-12 00:00:00.000', 'BUY')
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA004','CHRAAA0041','ITAA4',5,3,'2000-17-12 00:00:00.000', 'BUY')
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA004','CHRAAA0041','ITAA4',5,4,'2000-18-12 00:00:00.000', 'BUY')
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA004','CHRAAA0041','ITAA4',5,5,'2000-18-12 00:00:00.000', 'SELL')
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA008','CHRAAA0082','ITAA7',4,6,'2000-18-12 00:00:00.000', 'BUY')
insert into История_операций
(ИД_пользователя, ИД_персонажа, ИД_предмета, Цена, Номер_операции, Дата,
Тип_операции)
values('USRAAA010','CHRAAA0101','ITBA1',100,7,'2000-19-12 00:00:00.000', 'BUY')
```

4.2 Команды создания представлений

Приведем набор команд для создания представления, решающего следующую задачу: перечислить всех персонажей всех пользователей с количеством их доступной валюты.

```
CREATE VIEW валюта_персонажи
as SELECT Имя_персонажа as "Имя персонажа", Количество_валюты as "Количество валюты"
FROM Персонажи
```

4.3 Команды создания сложных представлений

Приведем набор команд для создания сложных представлений, решающих следующие задачи:

- Перечисление имен всех персонажей, которые приобрели предметы, являющиеся постоянными, с указанием этих предметов и их стоимости

```
CREATE VIEW постоянные_предметы
as SELECT п.Имя_персонажа as "Имя персонажа", к.Название as "Название предмета", к.Цена as "Цена предмета"
FROM Каталог к
inner join История_операций ио on к.ИД_предмета = ио.ИД_предмета
inner join Персонажи п on ио.ИД_персонажа = п.ИД_персонажа
WHERE к.Тип_предмета = 'permanent' AND ио.Тип_операции = 'BUY'
```

4.4 Команды использования представлений

Приведем команды использования представлений “валюта_персонажи” и “постоянные_предметы”

Результат использования представления “валюта_персонажи”:

```
select *
from валюта_персонажи
order by "Количество валюты" DESC
```

	Имя персонажа	Количество валюты
1	Cameron Hallewell	284
2	Caden Rutland	230
3	River Leighton	216
4	Marley Presley	182
5	Jamie Smithies	133
6	Rudy Whitley	105
7	Ashley Peyton	96
8	Charlie Harley	76
9	Casey Smithy	54
10	Brynn Abram	42
11	Raylee Crutchley	23
12	Maddox Breeden	19
13	Billie Mitchell	11
14	Jess Coombs	2

Рисунок 3.1 Результат использования представления “валюта_персонажи”.

Результат использования представления “постоянные_предметы”:

```
select *
from постоянные_предметы
order by "Цена предмета" DESC
```

	Имя персонажа	Название предмета	Цена предмета
1	Jess Coombs	Magic Staff	200
2	Marley Presley	Fire Sword	100

Рисунок 3.2 Результат использования представления “постоянные_предметы”

Часть 5. Реализация ограничений БД с помощью хранимых процедур и триггеров

5.1 Набор команд для создания хранимой процедуры

Приведем набор команд для создания хранимых процедур с параметрами:

- хранимая процедура “worth” с параметрами позволяет узнать по имени персонажа общую стоимость приобретенных им предметов

```
CREATE procedure worth
 @c varchar(20)
AS
 SELECT п.Имя_персонажа as "Имя персонажа", SUM(пп.Цена*пп.Количество) as "Сумма стоимости преметов"
 FROM Приобретенные_предметы пп
 inner join Персонажи п on пп.ИД_персонажа = п.ИД_персонажа
```

```
where @c = п.Имя_персонажа
group by п.Имя_персонажа
```

- хранимая процедура “purch_itms” с параметрами позволяет узнать по имени персонажа доступные для покупки предметы из каталога с указанием их возможного для покупки количества

```
CREATE procedure purch_itms
@c varchar(20)
AS
WITH P_I
AS
(SELECT п.Имя_персонажа as "Имя персонажа",п.Количество_валюты as "Количество валюты",к.Название as "Название предмета",к.Цена as "Стоимость предмета",
CASE WHEN к.Тип_предмета='temporary' then (п.Количество_валюты/к.Цена)
     WHEN (к.Тип_предмета='permanent') and (к.ИД_предмета != пп.ИД_предмета or пп.ИД_предмета is NULL) then 1
END as "Доступное количество"
FROM (select к.* from Каталог к) as к
inner join (select п.* from Персонажи п) as п on к.Цена<п.Количество_валюты
full outer join Приобретенные_предметы as пп on п.ИД_персонажа = пп.ИД_персонажа
where @c=п.Имя_персонажа)
SELECT *
FROM P_I
where "Доступное количество" is not NULL
order by "Стоимость предмета"
```

- хранимая процедура “user_ops” с параметрами позволяет узнать по имени персонажа его историю операций

```
CREATE procedure user_ops
 @c varchar(20)
AS
 SELECT к.Название as 'Название предмета',  ио.Тип_операции, ио.Цена, к.Тип_предмета
 FROM История_операций ио
 inner join Персонажи п ON п.ИД_персонажа = ио.ИД_персонажа
 inner join Каталог к ON к.ИД_предмета = ио.ИД_предмета
 WHERE @c = п.Имя_персонажа
```

- хранимая процедура “buy_itm” с параметрами позволяет выполнять операцию приобретения предмета из каталога

```
CREATE PROC buy_itm @cell_id VARCHAR(8),
                    @char_id CHAR(10),
                    @itm_id  CHAR(5)
AS
 BEGIN
    IF (SELECT к.Тип_предмета
       FROM   Каталог к
       WHERE  к.ИД_предмета = @itm_id) = 'permanent'
      BEGIN
        BEGIN TRANSACTION;
        IF NOT EXISTS((SELECT пп.ИД_персонажа
                       FROM   Приобретенные_предметы пп
                       WHERE  ( пп.ИД_предмета = @itm_id
                              AND пп.ИД_персонажа = @char_id
                            ))
                     )
          AND (SELECT п.Количество_валюты
               FROM   Персонажи п
               WHERE  п.ИД_персонажа = @char_id) >=
                 (SELECT к.Цена
                  FROM   Каталог к
                  WHERE  к.ИД_предмета = @itm_id)
            BEGIN
              UPDATE Персонажи
              SET    Персонажи.Количество_валюты =
                        п.Количество_валюты -
                        (SELECT к.Цена
                         FROM Каталог к
                         WHERE к.ИД_предмета = @itm_id)
              FROM   Персонажи п
              WHERE  п.ИД_персонажа = @char_id
```

```
INSERT INTO Приобретенные_предметы
        (ИД_ячейки,
         ИД_пользователя,
         ИД_персонажа,
         ИД_предмета,
         Цена,
         Количество)
VALUES     (@cell_id,
        (SELECT п.ИД_Пользователя
         FROM   Персонажи п
         WHERE  п.ИД_персонажа = @char_id),
        @char_id,
        @itm_id,
        (SELECT к.Цена
         FROM   Каталог к
         WHERE  к.ИД_предмета = @itm_id),
        1 )

INSERT INTO История_операций
        (ИД_пользователя,
         ИД_персонажа,
         ИД_предмета,
         Цена,
         Номер_операции,
         Дата,
         Тип_операции)
VALUES     ((SELECT п.ИД_Пользователя
         FROM   Персонажи п
         WHERE  п.ИД_персонажа = @char_id),
        @char_id,
        @itm_id,
        (SELECT к.Цена
```

```
                    FROM   Каталог к
                    WHERE  к.ИД_предмета = @itm_id),
                   (SELECT Max(ио.Номер_Операции)
                    FROM   История_операций AS ио)
                   + 1,
                   Getdate(),
                   'BUY' )
            PRINT('Предмет был успешно приобретен')
            COMMIT
          END
         ELSE
          BEGIN
            PRINT(
'Недостаточно валюты для приобретения предмета или этот предмет типа "постоянный" уже присутствует у персонажа'
    )
   ROLLBACK
END
END
ELSE
 BEGIN
    BEGIN TRANSACTION;
    IF (SELECT п.Количество_валюты
       FROM   Персонажи п
       WHERE  п.ИД_персонажа = @char_id) >=
        (SELECT к.Цена
         FROM   Каталог к
         WHERE  к.ИД_предмета = @itm_id)
      BEGIN
        UPDATE Персонажи
        SET    Персонажи.Количество_валюты =
                 п.Количество_валюты -
                 (SELECT к.Цена
```

```
                FROM Каталог к
                WHERE к.ИД_предмета = @itm_id)
    FROM   Персонажи п
    WHERE  п.ИД_персонажа = @char_id
    IF EXISTS(SELECT пп.ИД_предмета
            FROM   Приобретенные_предметы пп
            WHERE  ( пп.ИД_предмета = @itm_id
                    AND пп.ИД_персонажа = @char_id ))
     BEGIN
       UPDATE Приобретенные_предметы
       SET
Приобретенные_предметы.Количество =
      пп.Количество + 1
       FROM   Приобретенные_предметы пп
       WHERE  ( пп.ИД_предмета = @itm_id
              AND пп.ИД_персонажа = @char_id )
     END
    ELSE
     BEGIN
       INSERT INTO Приобретенные_предметы
                  (ИД_ячейки,
                   ИД_пользователя,
                   ИД_персонажа,
                   ИД_предмета,
                   Цена,
                   Количество)
       VALUES      (@cell_id,
                   (SELECT п.ИД_Пользователя
                    FROM   Персонажи п
                    WHERE  п.ИД_персонажа = @char_id),
                   @char_id,
                   @itm_id,
                   (SELECT к.Цена
```

```
                FROM   Каталог к
                WHERE  к.ИД_предмета = @itm_id),
               1 )
        END

    INSERT INTO История_операций
           (ИД_пользователя,
            ИД_персонажа,
            ИД_предмета,
            Цена,
            Номер_операции,
            Дата,
            Тип_операции)
    VALUES     ((SELECT п.ИД_Пользователя
             FROM   Персонажи п
             WHERE  п.ИД_персонажа = @char_id),
            @char_id,
            @itm_id,
            (SELECT к.Цена
             FROM   Каталог к
             WHERE  к.ИД_предмета = @itm_id),
            (SELECT Max(ио.Номер_Операции)
             FROM   История_операций AS ио)
            + 1,
            Getdate(),
            'BUY' )
    PRINT('Предмет был успешно приобретен')
    COMMIT
  END
 ELSE
  BEGIN
    PRINT(
'Недостаточно валюты для приобретения предмета'
```

```
    )


   ROLLBACK
END
END
END
```

- хранимая процедура “sell_itm” с параметрами позволяет выполнять операцию приобретения предмета из каталога

```
CREATE PROC sell_itm @char_id CHAR(10),
                @itm_id  CHAR(5)
AS
 BEGIN
    IF (SELECT к.Тип_предмета
       FROM   Каталог к
       WHERE  к.ИД_предмета = @itm_id) = 'permanent'
      BEGIN
        BEGIN TRANSACTION;

        IF EXISTS((SELECT пп.ИД_персонажа
                FROM   Приобретенные_предметы пп
                WHERE  ( пп.ИД_предмета = @itm_id
                      AND пп.ИД_персонажа = @char_id )))
         BEGIN
           UPDATE Персонажи
           SET    Персонажи.Количество_валюты =
                  п.Количество_валюты
                  +
                  (SELECT к.Цена
                   FROM
                  Каталог к
                  WHERE
                  к.ИД_предмета = @itm_id)
           FROM   Персонажи п
           WHERE  п.ИД_персонажа = @char_id

           DELETE FROM Приобретенные_предметы
           WHERE  (
          Приобретенные_предметы.ИД_предмета
          =
          @itm_id
          AND
Приобретенные_предметы.ИД_персонажа
=
@char_id )

   INSERT INTO История_операций
          (ИД_пользователя,
```

```
                ИД_персонажа,
                ИД_предмета,
                Цена,
                Номер_операции,
                Дата,
                Тип_операции)
    VALUES      ((SELECT п.ИД_Пользователя
                  FROM   Персонажи п
                  WHERE  п.ИД_персонажа = @char_id),
                @char_id,
                @itm_id,
                (SELECT к.Цена
                  FROM   Каталог к
                  WHERE  к.ИД_предмета = @itm_id),
                (SELECT Max(ио.Номер_Операции)
                  FROM   История_операций AS ио)
                + 1,
                Getdate(),
                'SELL' )
    PRINT('Предмет был успешно продан')
    COMMIT
END
ELSE
BEGIN
    PRINT(
'У этого персонажа отсутствует данный предмет'
)

ROLLBACK
END
END
ELSE
BEGIN
BEGIN TRANSACTION;

IF EXISTS((SELECT пп.ИД_персонажа
     FROM   Приобретенные_предметы пп
     WHERE  ( пп.ИД_предмета = @itm_id
           AND пп.ИД_персонажа = @char_id )))
BEGIN
UPDATE Персонажи
SET    Персонажи.Количество_валюты =
           п.Количество_валюты
                                        +
           (SELECT к.Цена
            FROM Каталог к
           WHERE к.ИД_предмета = @itm_id)
FROM   Персонажи п
WHERE  п.ИД_персонажа = @char_id

IF (SELECT пп.Количество
    FROM   Приобретенные_предметы пп
```

```
   WHERE  ( пп.ИД_предмета = @itm_id
         AND пп.ИД_персонажа = @char_id )) > 1
 BEGIN
    UPDATE Приобретенные_предметы
    SET
Приобретенные_предметы.Количество =
  пп.Количество - 1
    FROM   Приобретенные_предметы пп
    WHERE  ( пп.ИД_предмета = @itm_id
          AND пп.ИД_персонажа = @char_id )
 END
ELSE
 BEGIN
    DELETE FROM Приобретенные_предметы
    WHERE  (
 Приобретенные_предметы.ИД_предмета
 =
 @itm_id
 AND
Приобретенные_предметы.ИД_персонажа
=
@char_id )
END

INSERT INTO История_операций
(ИД_пользователя,
ИД_персонажа,
ИД_предмета,
Цена,
Номер_операции,
Дата,
Тип_операции)
VALUES     ((SELECT п.ИД_Пользователя
FROM   Персонажи п
WHERE  п.ИД_персонажа = @char_id),
@char_id,
@itm_id,
(SELECT к.Цена
FROM   Каталог к
WHERE  к.ИД_предмета = @itm_id),
(SELECT Max(ио.Номер_Операции)
FROM   История_операций AS ио)
+ 1,
Getdate(),
'SELL' )
PRINT('Предмет был успешно продан')
COMMIT
END
ELSE
BEGIN
PRINT(
'У этого персонажа отсутствует данный предмет'
```

```
)

ROLLBACK
END
END
END
```

5.2 Команды для вызова хранимой процедуры с параметрами

Приведем команды для вызова хранимой процедуры с параметрами:

- результат вызова хранимой процедуры “worth”

```
EXEC worth 'Charlie Harley'
```

	Имя персонажа	Сумма стоимости преметов
1	Charlie Harley	10

Рисунок 4.1. Результат вызова хранимой процедуры “worth”

- результат вызова хранимой процедуры “purch_itms”

```
EXEC purch_itms 'Marley Presley'
```

	Имя персонажа	Количество валюты	Название предмета	Стоимость предмета	Доступное количество
1	Marley Presley	182	Branch	3	60
2	Marley Presley	182	Cookie	4	45
3	Marley Presley	182	Bread	5	36
4	Marley Presley	182	Bottle of water	5	36
5	Marley Presley	182	Evil Potion	25	7
6	Marley Presley	182	Health Potion	30	6
7	Marley Presley	182	Mana Potion	30	6
8	Marley Presley	182	Boots of speed	100	1

Рисунок 4.2. Результат вызова хранимой процедуры “purch_itms”

- результат вызова хранимой процедуры “user_ops”

```
EXEC user_ops 'Charlie Harley'
```

	Название предмета	Тип_операции	Цена	Тип_предмета
1	Bottle of water	BUY	5	temporary
2	Bottle of water	BUY	5	temporary
3	Bottle of water	BUY	5	temporary
4	Bottle of water	SELL	5	temporary

Рисунок 4.3. Результат вызова хранимой процедуры “user_ops”

- результат вызова хранимой процедуры “buy_itm”

```
EXEC buy_itm 'I0001', 'CHRAAA0021', 'ITBA2'
```

```
(затронута одна строка)

(затронута одна строка)

(затронута одна строка)
Предмет был успешно приобретен
```

Рис 4.4. Результат вызова хранимой процедуры “buy_itm”

	ИД_ячейки	ИД_пользователя	ИД_персонажа	ИД_предмета	Цена	Количество
1	I0001	USRAAA002	CHRAAA0021	ITBA2	100	1
2	I0001	USRAAA003	CHRAAA0031	ITBA3	200	1
3	I0001	USRAAA004	CHRAAA0041	ITAA4	5	2
4	I0001	USRAAA008	CHRAAA0082	ITAA7	4	1
5	I0001	USRAAA010	CHRAAA0101	ITBA1	100	1

Рисунок 4.5. Результат вызова хранимой процедуры “buy_itm”

- результат вызова хранимой процедуры “sell_itm”

```
EXEC sell_itm 'CHRAAA0021', 'ITBA2'
```

```
(затронута одна строка)

(затронута одна строка)

(затронута одна строка)
Предмет был успешно продан
```

Рисунок 4.6. Результат вызова хранимой процедуры “sell_itm”

	ИД_пользователя	ИД_персонажа	ИД_предмета	Цена	Номер_операции	Дата	Тип_операции
1	USRAAA003	CHRAAA0031	ITBA3	200	1	2000-12-15 00:00:00.000	BUY
2	USRAAA004	CHRAAA0041	ITAA4	5	2	2000-12-16 00:00:00.000	BUY
3	USRAAA004	CHRAAA0041	ITAA4	5	3	2000-12-17 00:00:00.000	BUY
4	USRAAA004	CHRAAA0041	ITAA4	5	4	2000-12-18 00:00:00.000	BUY
5	USRAAA004	CHRAAA0041	ITAA4	5	5	2000-12-18 00:00:00.000	SELL
6	USRAAA008	CHRAAA0082	ITAA7	4	6	2000-12-18 00:00:00.000	BUY
7	USRAAA010	CHRAAA0101	ITBA1	100	7	2000-12-19 00:00:00.000	BUY
8	USRAAA002	CHRAAA0021	ITBA2	100	8	2000-12-20 01:02:33.000	BUY
9	USRAAA002	CHRAAA0021	ITBA2	100	9	2000-12-20 02:03:44.000	SELL

Рисунок 4.7. Результат вызова хранимой процедуры “sell_itm”

5.3 Набор команд для создания триггера

Приведем команды для создания триггера:

- триггер “del_char” срабатывает после удаления данных, в сущности “Персонажи”, рассчитывая количество персонажей пользователя, таким образом после удаления персонажа из базы данных удаляются только данные о конкретном персонаже, либо удаляется вся информация о пользователе

```
CREATE TRIGGER del_char
ON Персонажи
after DELETE
AS
  BEGIN
    DECLARE @del_usr_id  CHAR(9),
        @del_char_id CHAR(10)
```

```
    SELECT @del_char_id = deleted.ИД_персонажа,
        @del_usr_id = deleted.ИД_пользователя
    FROM   deleted

    IF EXISTS(SELECT п.ИД_пользователя
            FROM   Персонажи п
            WHERE  п.ИД_пользователя = @del_usr_id)
      BEGIN
        DECLARE @num_of_chars INTEGER

        SELECT @num_of_chars = Count(*)
        FROM   Персонажи п
        WHERE  п.ИД_пользователя = @del_usr_id

        PRINT( Concat(
'Количество оставшихся персонажей у пользователя: '
, @num_of_chars) )
END
ELSE
  BEGIN
    DELETE FROM Пользователи
    WHERE  Пользователи.ИД_пользователя =
        @del_usr_id

    PRINT(
'Все персонажи этого пользователя были удалены'
    )
END
END
```

5.4 Пример команды, вызывающей триггер

Приведем примеры команд, вызывающих представленный выше триггер:

- команды, вызывающие триггер “del_char”

```
DELETE FROM Персонажи
WHERE Персонажи.ИД_персонажа = 'CHRAAA0091'
```

```
Количество оставшихся персонажей у пользователя: 1

(затронута одна строка)
```

Рисунок 5.1. Вызов триггера “del_char”

```
DELETE FROM Персонажи
WHERE Персонажи.ИД_персонажа = 'CHRAAA0092'
```

```
(затронута одна строка)
Все персонажи этого пользователя были удалены

(затронута одна строка)
```

Рисунок 5.2. Вызов триггера “del_char”

5.5 Описание транзакций

Приведем список транзакций, используемых в хранимых процедурах:

- транзакции в процедуре “buy_itm”

Транзакции в этой хранимой процедуре необходимы для соблюдения условий покупки предметов персонажем, а также для обеспечения корректного выполнения списания средств с баланса валюты пользователя

- транзакции в процедуре “sell_itm”

Транзакции в этой хранимой процедуре необходимы для соблюдения условий продажи предметов персонажем, а также для обеспечения корректного выполнения зачисления средств на баланс валюты пользователя

5.6 Реализация транзакций на языке SQL

Приведем способ реализации транзакций на языке SQL:

- Транзакции в процедуре “buy_itm” сохраняют изменения, при выполнении условий отсутствия такого же предмета типа “постоянный” у персонажа, а также наличия у персонажа достаточного количества валюты для покупки предмета. В противном случае изменения отменяются.
- Транзакции в процедуре “sell_itm” сохраняют изменения, при выполнении условий наличия у персонажа продаваемого предмета. В противном случае изменения отменяются.

Заключение

В ходе выполнения курсовой работы была спроектирована БД, что и являлось ее целью.

Разработанная мной БД включает в себя различные виды связей, ограничений, представлений, триггеров и процедур.

БД и её функционирование были протестированы удачно, таким образом, можно сделать вывод о том, что БД “Магазин предметов для многопользовательской игры” работает правильно и что она готова для дальнейшего практического использования.

Список литературы:

1) Бьюли А. Изучаем SQL: Пер. с англ. – СПб: Символ-Плюс, 2007. – 312 с.
2) Дейт, К.Дж. Введение в системы баз данных : [пер. с англ.] / Дейт К.Дж. – 8-е изд. – Москва ; СПб. ; Киев : Издательский дом «Вильямс», 2005. – 1327 с.
3) Маркин, А. В. Построение запросов и программирование на SQL. Учебное пособие / А.В. Маркин. - М.: Диалог-Мифи, 2014. - 384 с.
4) Малыхина, М. Базы данных: основы, проектирование, использование: Учебное пособие / М. Малыхина. – СПб. : БХВ-Петербург, 2004.
5) Роб, Питер Системы баз данных: проектирование, реализация и управление : Пер. с англ. / П. Роб, К. Коронелл. – 5-е изд. – СПб. : БХВПетербург, 2004. – 1024 с. : ил.