|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)**

**ОТЧЁТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**Ознакомительная практика**

приказ Университета о направлении на практику от «07» февраля 2022 г. № 858-С

|  |  |  |
| --- | --- | --- |
| Отчет представлен к  рассмотрению:  Студент группы ИКБО-27-21 | «\_\_» июня 2022 | (подпись и расшифровка подписи) |
| Отчет утвержден.  Допущен к защите: |  |  |
| Руководитель практики  от кафедры | «\_\_» июня 2022 | (подпись и расшифровка подписи) |

Москва 2022 г.

|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)**

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА УЧЕБНУЮ ПРАКТИКУ**

**Ознакомительная практика**

**Студенту 1 курса учебной группы ИКБО-27-21**

**Осипову Михаилу Алексеевичу**

**Место и время практики:** РТУ МИРЭА кафедра МОСИТ, с 09 февраля 2022 г. по 31 мая 2022 г.

**Должность на практике:**  студент

**1. СОДЕРЖАНИЕ ПРАКТИКИ:**

1.1. Изучить: изучить материал в соответствии с выданной темой (База Данных персонажей игры Brawl Stars).

1.2. Практически выполнить: разработать собственную базу данных и выполнить задания по ней.

1.3. Ознакомиться: с PostgreSQL.

**2. ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ:** нет

**3. ОРГАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ:** в соответствии с методическими указаниями по учебной практике по направлению 09.03.04 «Программная инженерия»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Руководитель практики от кафедры  «09» февраля 2022 г. | | | *Подпись* | | (Чернов Е.А.) |
| Задание получил | |  |  | |  |
| «09» февраля 2022 г. | |  | *Подпись* | | (Родионов А.А.) |
|  | |  |  | |  |
|  | |  |  | |  |
|  | |  |  | |  |
|  | |  |  | |  |
|  | |  |  | |  |
| **СОГЛАСОВАНО:** | |  |  | |  |
| Заведующий кафедрой: | | | | | |
| «09» февраля 2022 г. | | | *Подпись* | | (Головин С.А.) |
| **Проведенные инструктажи:** |  | | |  | |
| Охрана труда: |  | | | «09» февраля 2022 г. | |
| Инструктирующий | *Подпись* | | | Чернов Е.А., Доцент каф.  МОСИТ | |
| Инструктируемый | *Подпись* | | | Родионов А.А. | |
|  |  | | |  | |
| Техника безопасности: |  | | | «09» февраля 2022 г. | |
| Инструктирующий | *Подпись* | | | Чернов Е.А., Доцент каф.  МОСИТ | |
| Инструктируемый | *Подпись* | | | Родионов А.А. | |
|  |  | | |  | |
| Пожарная безопасность: |  | | | «09» февраля 2022 г. | |
| Инструктирующий | *Подпись* | | | Чернов Е.А., Доцент каф.  МОСИТ | |
| Инструктируемый | *Подпись* | | | Родионов А.А. | |
|  | | | |  | |
| С правилами внутреннего распорядка ознакомлен: | | | | «09» февраля 2022 г. | |
|  | *Подпись* | | | Родионов А.А. | |

|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**РАБОЧИЙ ГРАФИК ПРОВЕДЕНИЯ  
ОЗНАКОМИТЕЛЬНОЙ ПРАКТИКИ**

студента Осипова М.А. 1 курса группы ИКБО-27-21 очной формы обучения, обучающегося по направлению подготовки 09.03.04 Программная инженерия

|  |  |  |  |
| --- | --- | --- | --- |
| **Неделя** | **Сроки**  **выполнения** | **Этап** | **Отметка о выполнении** |
| 2 | 14.02.2022 | Подготовительный этап, включающий в себя организационное собрание (Вводная лекция о порядке организации и прохождения производственной практики, инструктаж по технике безопасности, получение задания на практику) |  |
| 6 | 13.03.2022 | Установка PostgreSQL, создание базы данных и блок-схемы её таблиц |  |
| 6 | 14.03.2022 | Представление руководителю базы данных и блок-схемы |  |
| 8 | 27.03.2022 | Создание таблиц и их заполнение в соответствии с выбранной темой (БД персонажей игры Brawl Stars) |  |
| 8 | 28.03.2022 | Представление руководителю заполненных таблиц БД в соответствии с выбранной темой |  |
| 10 | 10.04.2022 | Создание запросов и их выполнение |  |
| 10 | 11.04.2022 | Представление руководителю выполнения созданных запросов |  |
| 12 | 24.04.2022 | Запись результатов запросов в файл result.txt |  |
| 12 | 25.04.2022 | Представление руководителю записи результатов |  |
| 16 | 31.05.2022 | Подготовка окончательной версии отчёта (Оформление материалов отчета в полном соответствии с требованиями на оформление письменных учебных работ студентов) |  |

Руководитель практики от  
кафедры \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Чернов Е.А., Доцент/

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Родионов А.А./

**Согласовано:**

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Головин С.А., д.т.н., профессор/

1. **Теория.**

**База данных**

***База данных*** — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД). Данные вместе с СУБД, а также приложения, которые с ними связаны, называются системой баз данных, или, для краткости, просто базой данных.

Данные в наиболее распространенных типах современных баз данных обычно хранятся в виде строк и столбцов формирующих таблицу. Этими данными можно легко управлять, изменять, обновлять, контролировать и упорядочивать. В большинстве баз данных для записи и запросов данных используется язык структурированных запросов (SQL).

**SQL**

***SQL*** — это язык программирования, используемый в большинстве реляционных баз данных для запросов, обработки и определения данных, а также контроля доступа. SQL был разработан в IBM в 1970-х годах. Со временем у стандарта SQL ANSI появились многочисленные расширения, разработанные такими компаниями как IBM, Oracle и Microsoft. Хотя в настоящее время SQL все еще широко используется, начали появляться новые языки программирования запросов.

**Эволюция базы данных**

Базы данных значительно изменились с момента их появления в начале 1960-х годов. Исходными системами, которые использовались для хранения и обработки данных, были навигационные базы данных – например, иерархические базы данных (которые опирались на древовидную модель и допускали только отношение «один-ко-многим») и базы данных с сетевой структурой (более гибкая модель, допускающая множественные отношения). Несмотря на простоту, эти ранние системы были негибкими. В 1980-х годах стали популярными реляционные базы данных, в 1990-х годах за ними последовали объектно-ориентированные базы данных. Совсем недавно вследствие роста Интернета и возникновения необходимости анализа неструктурированных данных появились базы данных NoSQL. В настоящее время облачные базы данных и автономные базы данных открывают новые возможности в отношении способов сбора, хранения, использования данных и управления ими.

**Типы баз данных.**

* *Реляционные базы данных*
* *Объектно-ориентированные базы данных*
* *Распределенные базы данных*
* *Хранилища данных*
* *Базы данных NoSQL*
* *Графовые базы данных*
* *Облачные базы данных*
* *Многомодельные базы данных*
* *Автономные базы данных*

**Что такое программное обеспечение базы данных?**

Что такое программное обеспечение базы данных?

Программное обеспечение базы данных используется для создания, редактирования и обслуживания файлов и записей базы данных, что упрощает создание файлов и записей, ввод данных, редактирование, обновление и отчетность. Программное обеспечение также помогает хранить данных, осуществлять резервное копирование и формировать отчетность, предоставлять управление множественным доступом и поддерживать безопасность. Сегодня надежная безопасность базы данных особенно важна, поскольку случаи кражи данных значительно участились. Программное обеспечение для баз данных иногда называют системой управления базами данных (СУБД).

Программное обеспечение баз данных упрощает управление данными, помогая пользователям хранить данные в структурированной форме, а затем получать к ним доступ. Обычно программа имеет графический интерфейс, помогающий создавать данные и управлять ими, и в некоторых случаях пользователи могут создавать собственные базы данных с помощью такого ПО.

**Что такое система управления базами данных (DBMS)?**

Для базы данных обычно требуется комплексное программное обеспечение, которое называется системой управления базами данных (СУБД). СУБД служит интерфейсом между базой данных и пользователями или программами, предоставляя пользователям возможность получать и обновлять информацию, а также управлять ее упорядочением и оптимизацией. СУБД обеспечивает контроль и управление данными, позволяя выполнять различные административные операции, такие как мониторинг производительности, настройка, а также резервное копирование и восстановление.

В качестве примеров популярного программного обеспечения для управления базами данных, или СУБД, можно назвать MySQL, PostgreSQL, Microsoft Access.

1. **Практическая Часть.**

Для начала работы над задачей, надо установить ПО PostgresSQL. Во время установки ставим пароль суперпользователя «postgres». Работа с системой управления базами данных PostgreSQL осуществляется с помощью программы psql.

Создадим базу данных с помощью командных запросов CREATE DATABASE, которую назовём rodionov24. Далее создадим роль с помощью команды CREATE ROLE, которая будет использоваться для доступа и работы с создаваемой базой данных. Назовём её server24. Для простоты работы укажем атрибут LOGIN, который позволит выполнять вход от имени этой роли. Также при создании роли задаём пароль, который будет равным имени роли. Для подключения к созданной базе данных отмени созданной роли использует мета-команда \c.

Запросы для создания базы данных и роли:

CREATE DATABASE rodionov24;

CREATE ROLE server21 WITH LOGIN PASSWORD ‘server24;

\c rodionov24 server24

Чтобы начать работу над базой данных, нужно придумать что конкретно будет храниться в нашей базе данных. Проектирование диаграммы базы данных будет выполняться в draw.io. В среде draw.io будем использовать элементы из группы «Entity Relation»: «Table» в качестве основы таблицы и «Row 3» и «Row 6» для добавления первичного ключа и обычного поля соответственно.

Проанализировав поставленную задачу и выделил сущность можно определить, что база данных в поставленной задаче будет состоять и четырёх таблиц: Бравлеры (Brawlers), Скины (skins) и Гаджеты(Gadgets). Также необходима будет вспомогательная таблица Лучшие гаджеты (Best gadgets) для реализации связи “Многие ко Многим”.

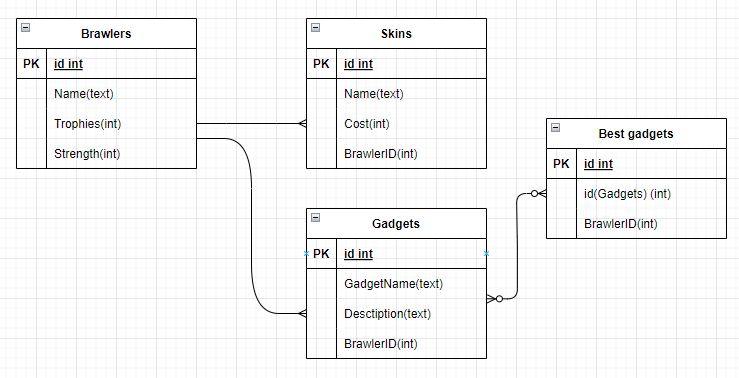


Рис. 1 Проектирование базы данных

База данных спроектирована, теперь нужно написать SQL-запросы для создания необходимых сущностей. В нашей задаче Brawlers, Skins, Gadgets содержат первичный ключ – уникальный идентификационный номер. Для более удобного добавления записей в таблицу, будем использовать последовательность– генератор чисел, которые можно будет использовать лишь для того, чтобы отсчитывать номера за нас, каждый раз выдавая новое число.

Последовательность – самостоятельная сущность, никак не зависящая от какой-либо таблицы. Последовательность может быть использована для любых целей.

Запросы для создания последовательности:

CREATE SEQUENCE seq\_brawlerid;

CREATE SEQUENCE seq\_skinid;

CREATE SEQUENCE seq\_gadgetid;

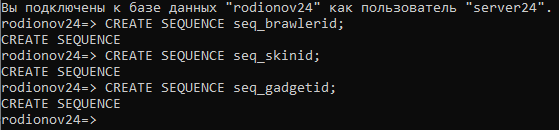


Рис. 2 Успешное создание последовательностей

Далее необходимо создать таблицы, которые будут хранить данные в соответствии с поставленной задачей. После названия таблицы в скобках указываются имена атрибутов, их тип и прочие параметры атрибута. Уникальный ключ обозначается параметром PRIMARY KEY, а с помощью параметра DEFAULT и функции последовательности nextval укажем, что по умолчанию при добавлении новой строки в эту таблицу это поле будет равно следующему значению из созданной ранее последовательности. Остальные поля вводим в соответствии с диаграммой и типами данных, соответствующими задаче.

CREATE TABLE brawler

(

id INT PRIMARY KEY DEFAULT nextval('seq\_brawlerid'),

name TEXT,

Trophies INT,

Strength INT

);

CREATE TABLE skin

(

id INT PRIMARY KEY DEFAULT nextval('seq\_skinid'),

name TEXT,

Cost INT,

BrawlerID INT REFERENCES "brawler" ON DELETE CASCADE,

FOREIGN KEY(BrawlerID) REFERENCES brawler(id)

);

CREATE TABLE gadget

(

id INT PRIMARY KEY DEFAULT nextval('seq\_gadgetid'),

name TEXT,

Description TEXT,

BrawlerID INT REFERENCES "brawler",

FOREIGN KEY(BrawlerID) REFERENCES brawler(id)

);

CREATE TABLE best\_gadgets

(

GadgetID INT REFERENCES "gadget" ON DELETE RESTRICT,

BrawlerID INT REFERENCES "brawler" ON DELETE CASCADE,

FOREIGN KEY (GadgetID) REFERENCES "gadget"(id),

FOREIGN KEY (BrawlerID) REFERENCES "brawler"(id)

);

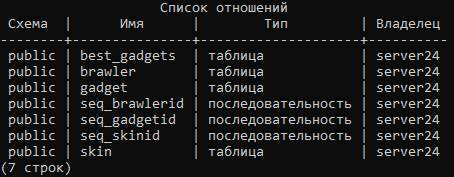


Рис. 3 Успешное создание таблиц

Также необходимо создать вспомогательную таблицу, реализующую связь «Многтм ко Многим». Таблица будет содержать только внешние ключи на другие таблицы, что объявляется с помощью ключевого слова REFERENCES и указания таблицы, на которую этот внешний ключ будет указывать. Также добавим параметр ON DELETE RESTRICT, который будет требовать удалить строку, содержащую внешний ключ, ссылающийся на удаляемую строку, и параметр ON DELTE CASCADE, который сделает удаление за нас.

CREATE TABLE best\_gadgets

(

GadgetID INT REFERENCES "gadget" ON DELETE RESTRICT,

BrawlerID INT REFERENCES "brawler" ON DELETE CASCADE,

FOREIGN KEY (GadgetID) REFERENCES "gadget"(id),

FOREIGN KEY (BrawlerID) REFERENCES "brawler"(id)

);

Для проверки создание сущностей будем использовать мета-команду \d-это команда выводит список таблиц.

Вывод созданных сущностей:

\d

\d brawler

\d skin

\d gadget

\d best\_gadgets

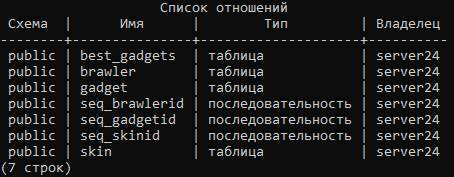


Рис. 4 Успешное создание сущностей

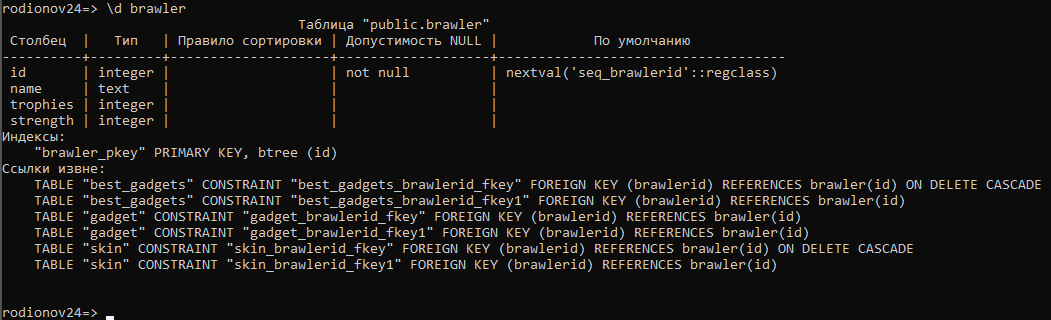
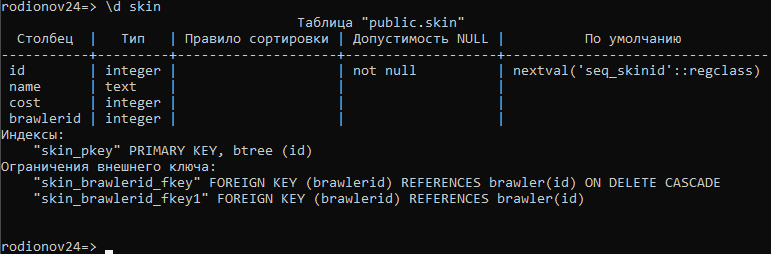
Проверить правильность создания конкретной таблицы можно с помощью той же мета-команды, в таком случае она выведет описание заданного элемента.

Рис. 5 Проверка правильности таблицы brawlers



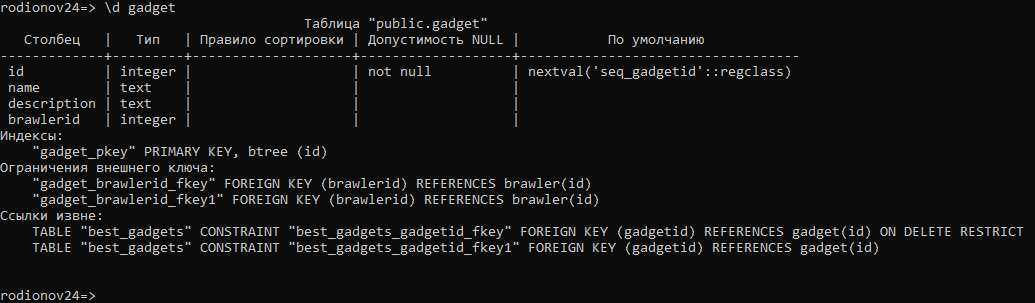
Рис. 6 Проверка правильности таблицы skin

Рис. 7 Проверка правильности таблицы covers

Как было описано ранее, база данных в основном состоит из таблиц, а таблицы хранят в себе данные в виде строк. Наполним созданные таблицы данными. Для этого создадим файл sql, в котором напишем все необходимые запросы, а затем укажем программе psql на исполнение запросов из этого файла, вместо того, чтобы вводить все команды в консоль.

Прежде чем создавать файл для заполнения базы данных, стоит создать файл для ее очистки для быстрого исправления ошибок. Очистку таблицы можно осуществить с помощью простой sql-команды TRUNCATE.

Очистка таблиц производится параллельно с очисткой таблицы связей, так как мы запретили удаление записей, на которые ссылаются записи в этой таблице. Также необходимо сбросить счетчик последовательностей, чтобы номера новых записей начинались с единицы, с помощью команды ALTER SEQUENCE.

Файл clear.sql . Операции очистки таблиц:

ALTER SEQUENCE seq\_brawlerid RESTART WITH 1;

ALTER SEQUENCE seq\_skinid RESTART WITH 1;

ALTER SEQUENCE seq\_gadgetid RESTART WITH 1;

TRUNCATE best\_gadgets, brawler, skin, gadget;

Вставка данных в таблицу производится sql-командой INSERT. Далее в скобках указывается имя таблицы, в которую будут вставлены данные, перечень параметров, а затем в скобках их значения в соответствующем порядке.

Далее требуется написать следующие запросы:

1) Добавление не менее 10 записей в одну таблицу, 2 из с переопределением значений по умолчанию.

2) Добавление не менее 5 записей во вторую таблицу

3) Добавление не менее 5 записей в таблицу, хранящую связи.

Файл insert.sql. Содержание файла:

INSERT INTO brawler (name,Trophies,Strength) VALUES ('dynamike',678,10);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('mr p',583,10);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('barley',574,9);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('jessie',599,9);

INSERT INTO brawler (id, name,Trophies,Strength) VALUES (5, 'shelly',560,9);

SELECT nextval('seq\_brawlerid');

INSERT INTO brawler (name,Trophies,Strength) VALUES ('edgar',568,9);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('nita',599,8);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('piper',574,8);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('spike',574,8);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('poco',494,7);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('bo',536,7);

INSERT INTO brawler (name,Trophies,Strength) VALUES ('fang',206,6);

INSERT INTO skin (id, name, Cost, BrawlerID) VALUES (1,'robomike',399,1);

SELECT nextval('seq\_skinid');

INSERT INTO skin (name, Cost, BrawlerID) VALUES ('classic mike',0,1);

INSERT INTO skin (name, Cost, BrawlerID) VALUES ('golden barley',49,3);

INSERT INTO skin (name, Cost, BrawlerID) VALUES ('wizard barely',49,3);

INSERT INTO skin (name, Cost, BrawlerID) VALUES ('unicorn knight barley',149,3);

INSERT INTO gadget (name, Description, BrawlerID) VALUES ('satchel charge','stun for 1.5s',1);

INSERT INTO gadget (name, Description, BrawlerID) VALUES ('fidget spinner','spin and throw dinamite',1);

INSERT INTO gadget (name, Description, BrawlerID) VALUES ('service bell','buff porter',2);

INSERT INTO gadget (name, Description, BrawlerID) VALUES ('porter reinforcment','next attack spawn porter',2);

INSERT INTO best\_gadgets (GadgetID,BrawlerID) VALUES (1,1);

INSERT INTO best\_gadgets (GadgetID,BrawlerID) VALUES (3,2);

INSERT INTO best\_gadgets (GadgetID,BrawlerID) VALUES (4,2);



Рис. 8 Заполнение таблицы brawler



Рис. 9 Заполнение таблицы skin

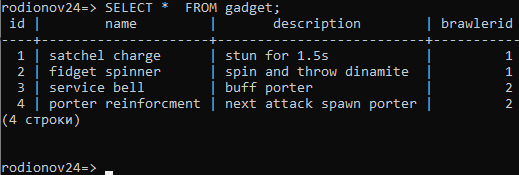


Рис. 10 Заполнение таблицы covers

Данные нужно не только хранить в базе данных, но и извлекать для чтения. Данные извлекаются из таблиц с помощью SQL-запроса SELECT. Далее в скобках указывается что требуется извлечь и в каком порядке. Если нужно извлечь все данные из таблицы, можно поставить символ \*, вместо того, чтобы перечислять все столбцы. Далее пунктом FROM указывается, откуда эти данные требуется извлечь (название таблицы), после чего указываются дополнительные пункты в зависимости от задачи. Например, пунктом ORDER BY указывается столбец, по которому полеченные строки будут отсортированы, а пунктом WHERE определяется условие, по которому полученные строки будут отфильтрованы.

Далее требуется написать следующие запросы:

1. Чтение всех данных из одной таблицы так, как они записаны в таблице;
2. Чтение и сортировка определенных столбцов данных из второй таблицы;
3. Чтение данных с условием из третей таблицы. Условие должно быть составным – содержать несколько логических операторов (аналогично примеру).

Файл select.sql. Содержание файла:

SELECT \* FROM brawler WHERE Strength >= 8 AND Trophies >= 550 ORDER BY Trophies DESC;

SELECT (name,Cost) FROM skin ORDER BY Cost DESC;

SELECT \* FROM gadget;

SELECT \* FROM best\_gadgets;



Рис. 11 Выполнение задания 1 файла select.sql

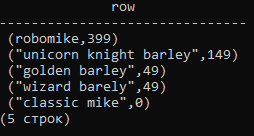


Рис. 12 Выполнение задания 2 файла select.sql

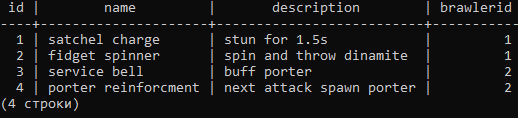


Рис. 13 Выполнение задания 3 файла select.sql

Данные, хранящиеся в таблицах, также можно изменять или удалять.

Изменение данных производится с помощью SQL-запроса UPDATE. В команде указывается таблица, в которой требуется произвести изменения, затем пунктом SET определяются необходимые изменения, после чего пунктом WHERE указываются условия, определяющие какие записи требуется изменить.

Удаление записей производится с помощью SQL-запроса DELETE. Пунктом FROM указывается, в какой таблице требуется удалить записи. затем командой WHERE указываются условия, определяющие какие записи требуется удалить.

Далее требуется написать следующие запросы:

1. Изменение не менее 2 записей одной таблицы так, чтобы изменился результат второго запроса из файла выше (запрос с сортировкой строк);
2. Удаление другой таблицы записи, на которую ссылается какая-либо запись из таблицы связей с параметром ON DELETE RESTRICT. Этот запрос должен провалиться, так как мы запретили подобное удаление;
3. Удаление записи из таблицы связей, которая ссылается на ту, что мы пытались удалить;
4. Повторение второго пункта, теперь он должен выполниться без ошибок.
5. Удаление другой таблицы записи, на которую ссылается какая-либо запись из таблицы связей с параметром ON DELETE CASCADE.

Файл update.sql. Содержание файла:

UPDATE skin SET Cost = 499 WHERE name = 'robomike';

UPDATE skin SET Cost = 79 WHERE name = 'golden barley';

DELETE FROM brawler WHERE name = 'mr p';

DELETE FROM best\_gadgets WHERE BrawlerID = 2;

DELETE FROM gadget WHERE BrawlerID = 2;

DELETE FROM brawler WHERE name = 'mr p';

DELETE FROM brawler WHERE name = 'barley';

Таким образом были написаны 4 файла:

1. Очистка базы данных – clear.sql;
2. Добавление данных – insert.sql;
3. Получение данных – select.sql;
4. Изменение данных – update.sql;

Теперь необходимо выполнить написанные запросы. Для этого воспользуемся мета-командой psql \i, которая считывает и исполняет весь файл. После команды должен быть указан путь к файлу в одинарных кавычках.

Надо исполнить файлы в следующем порядке: добавление данных, получение данных, изменение данных, получение данных, очистка базы данных.

Вызов SQL запросов через файлы и запись в отдельный файл result.txt:

\o 'D:/ВУЗ/СУБД/result.txt'

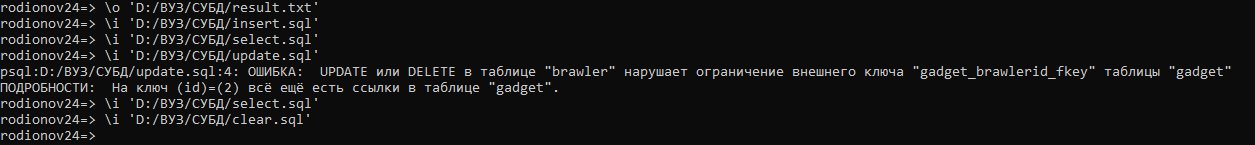
\i 'D:/ВУЗ/СУБД/insert.sql'

\i 'D:/ВУЗ/СУБД/select.sql'

\i 'D:/ВУЗ/СУБД/update.sql'

\i 'D:/ВУЗ/СУБД/select.sql'

\i 'D:/ВУЗ/СУБД/clear.sql'



Так как мы ранее пытались удалить запись другой таблицы, на которую ссылается какая-либо запись из таблицы связей с параметром ON DELETE RESTRICT, вызывается ошибка, так как мы запретили подобное удаление.

Результаты выполнение программы были записаны в отдельный файл result.txt с помощью мета-команды \o.

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

nextval

---------

5

(1 строка)

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

nextval

---------

1

(1 строка)

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

INSERT 0 1

id | name | trophies | strength

----+----------+----------+----------

1 | dynamike | 678 | 10

7 | nita | 599 | 8

4 | jessie | 599 | 9

2 | mr p | 583 | 10

9 | spike | 574 | 8

3 | barley | 574 | 9

8 | piper | 574 | 8

6 | edgar | 568 | 9

5 | shelly | 560 | 9

(9 строк)

row

-------------------------------

(robomike,399)

("unicorn knight barley",149)

("golden barley",49)

("wizard barely",49)

("classic mike",0)

(5 строк)

id | name | description | brawlerid

----+---------------------+--------------------------+-----------

1 | satchel charge | stun for 1.5s | 1

2 | fidget spinner | spin and throw dinamite | 1

3 | service bell | buff porter | 2

4 | porter reinforcment | next attack spawn porter | 2

(4 строки)

gadgetid | brawlerid

----------+-----------

1 | 1

3 | 2

4 | 2

(3 строки)

UPDATE 1

UPDATE 1

DELETE 2

DELETE 2

DELETE 1

DELETE 1

id | name | trophies | strength

----+----------+----------+----------

1 | dynamike | 678 | 10

4 | jessie | 599 | 9

7 | nita | 599 | 8

8 | piper | 574 | 8

9 | spike | 574 | 8

6 | edgar | 568 | 9

5 | shelly | 560 | 9

(7 строк)

row

--------------------

(robomike,499)

("classic mike",0)

(2 строки)

id | name | description | brawlerid

----+----------------+-------------------------+-----------

1 | satchel charge | stun for 1.5s | 1

2 | fidget spinner | spin and throw dinamite | 1

(2 строки)

gadgetid | brawlerid

----------+-----------

1 | 1

(1 строка)

ALTER SEQUENCE

ALTER SEQUENCE

ALTER SEQUENCE

TRUNCATE TABLE

**III. Вывод.**

В результате выполнения работы я получил навыки по работе с базой данных. Научился работать с запросами, создавать таблицы, выводить, добавлять и работать с данными в таблице.