

РЕФЕРАТ

Рассчетно-пояснительная записка состоит из 43 страницы, включающих 20 рисунков, 37 листингов, 10 таблиц и 4 приложения.

Перечень ключевых слов: База данных, PostgreSQL, Python, Ruby, PGAdmin4, Актеры театра, Занятость, Репетиции, Сцены, Роли, Инфологическая модель, Датологическая модель, Бизнеспроцессы, СУБД, Реализация базы данных

Целью создания данной базы данных является организация и хранение структурированной информации о занятости актеров.

Для реализации этой базы данных используются следующие инструменты и технологии:

- Система управления базами данных (СУБД) PostgreSQL: мощная объектно-реляционная СУБД, обеспечивающая высокую производительность, надежность и расширяемость для управления данными.
- Языки программирования Python и Ruby: используются для написания скриптов и выполнения операций с базой данных, таких как вставка и обновление данных.
- Среда разработки PGAdmin4: универсальная многофункциональная программа для работы с базами данных, предоставляющая удобный графический интерфейс для разработки, администрирования и управления базами данных.

СОДЕРЖАНИЕ

Введение	5
1 Анализ предметной области	6
1.1 Инфологическая модель базы данных	7
1.2 Датологическая модель базы данных	8
1.3 Таблицы базы данных	8
1.4 Разработка бизнес-процессов предметной области	12
2 Реализация базы данных	12
2.1 Создание базы данных и таблиц	12
2.2 Связывание таблиц	14
2.3 Заполнение базы данных	16
3 Запросы и триггеры	25
3.1 Запрос 1	25
3.2 Запрос 2	27
3.3 Запрос 3	29
3.4 Запрос 4	30
3.5 Запрос 5	31
3.6 Запрос 6	33
3.7 Триггер	34
Заключение	35
Список использованной литературы	36
Приложение А Генерация фотографий	37
Приложение Б Генерация основного объема данных	38
Приложение В Генерация названий пьес и спектаклей	42
Приложение Г Перевод названий пьес и спектаклей	43

ВВЕДЕНИЕ

Цель данной курсовой работы — разработка комплексной базы данных для организации, хранения и управления структурированной информацией о занятости актеров театра. Эта база данных будет предназначена для эффективного учета и отслеживания всех аспектов участия актеров в театральных постановках, включая данные о репетициях, спектаклях, ролях и контрактах.

Основная задача этой базы данных — централизовать и упростить хранение данных, что позволит улучшить доступ к информации, упростить её обработку и анализ, а также повысить прозрачность и эффективность работы театрального коллектива.

Основная проблема, которую необходимо решить с помощью этой базы данных, — эффективное управление большими объемами разнородной информации, связанной с занятостью актеров. В настоящее время данные часто хранятся в разных форматах и местах, что затрудняет их поиск и использование. Кроме того, отсутствие централизованного хранилища данных может привести к потере информации, дублированию записей и увеличению времени на обработку запросов.

Актуальность данной работы обусловлена ростом числа выпускников актерских вузов, которые будут нуждаться в работе в театре. Эффективное управление данными позволит сократить время на поиск и обработку информации, повысить точность и надежность данных о занятости актеров, а также улучшить взаимодействие между различными подразделениями театра.

Для достижения поставленных целей необходимо решить следующие задачи:

- Разработать структуру базы данных для удобного и эффективного хранения информации о репетициях, спектаклях, ролях и контрактах.
- Реализовать функциональность для учета актеров, включая сохранение их данных и фотографий.
- Организовать хранение информации о спектаклях, включая название, бюджет, год постановки и возрастные ограничения.
- Обеспечить учет занятости актеров, включая их участие в репетициях и спектаклях, а также распределение ролей.
- Управлять информацией о контрактах, включая даты приема и увольнения, а также ставки.

1 Анализ предметной области

Для анализа предметной области занятости актеров театра были выявлены несколько ключевых сущностей, которые необходимы для полного и точного отражения всех аспектов функционирования театра. Эти сущности включают информацию о актерах, спектаклях, контрактах, занятости в спектаклях, репетициях, пьесах, сценах и ролях. Каждая сущность играет важную роль в обеспечении целостности и полноты данных, необходимых для эффективного управления занятостью актеров театра.

Ниже приведен анализ каждой из выявленных сущностей:

- **Актеры** - включает информацию о театральных актерах, например ФИО, дата рождения, фото, стаж работы и звания/награды.
- **Спектакли** - содержит информацию о театральных спектаклях, включая название, бюджет, год постановки, возрастное ограничение и код пьесы.
- **Контракт** - охватывает данные о контрактах актеров, включая дату приема, дату увольнения и ставку.
- **Занятость в спектаклях** - включает информацию о занятости актеров в спектаклях.
- **Репетиции** - содержит данные о репетициях спектаклей, такие как дата и время, продолжительность, является ли репетиция читкой и код спектакля.
- **Сцены в репетициях** - охватывает информацию о сценах, репетируемых в конкретных репетициях.
- **Пьесы** - включает данные о пьесах, таких как название, год выпуска и автор.
- **Сцены** - содержит информацию о сценах в пьесах, включая название, продолжительность и код пьесы.
- **Роли** - охватывает данные о ролях, исполняемых актерами в пьесах, включая название роли, является ли роль главной, код пьесы и код актера.
- **Занятость ролей в сценах** - включает информацию о занятости ролей в конкретных сценах.

Этот анализ помогает структурировать данные для эффективного управления занятостью актеров в театре и обеспечивает точное и полное представление всех аспектов театральной деятельности.

1.1 Инфологическая модель базы данных

Исходя из сущностей и их свойств, построим инфологическую модель базы данных.

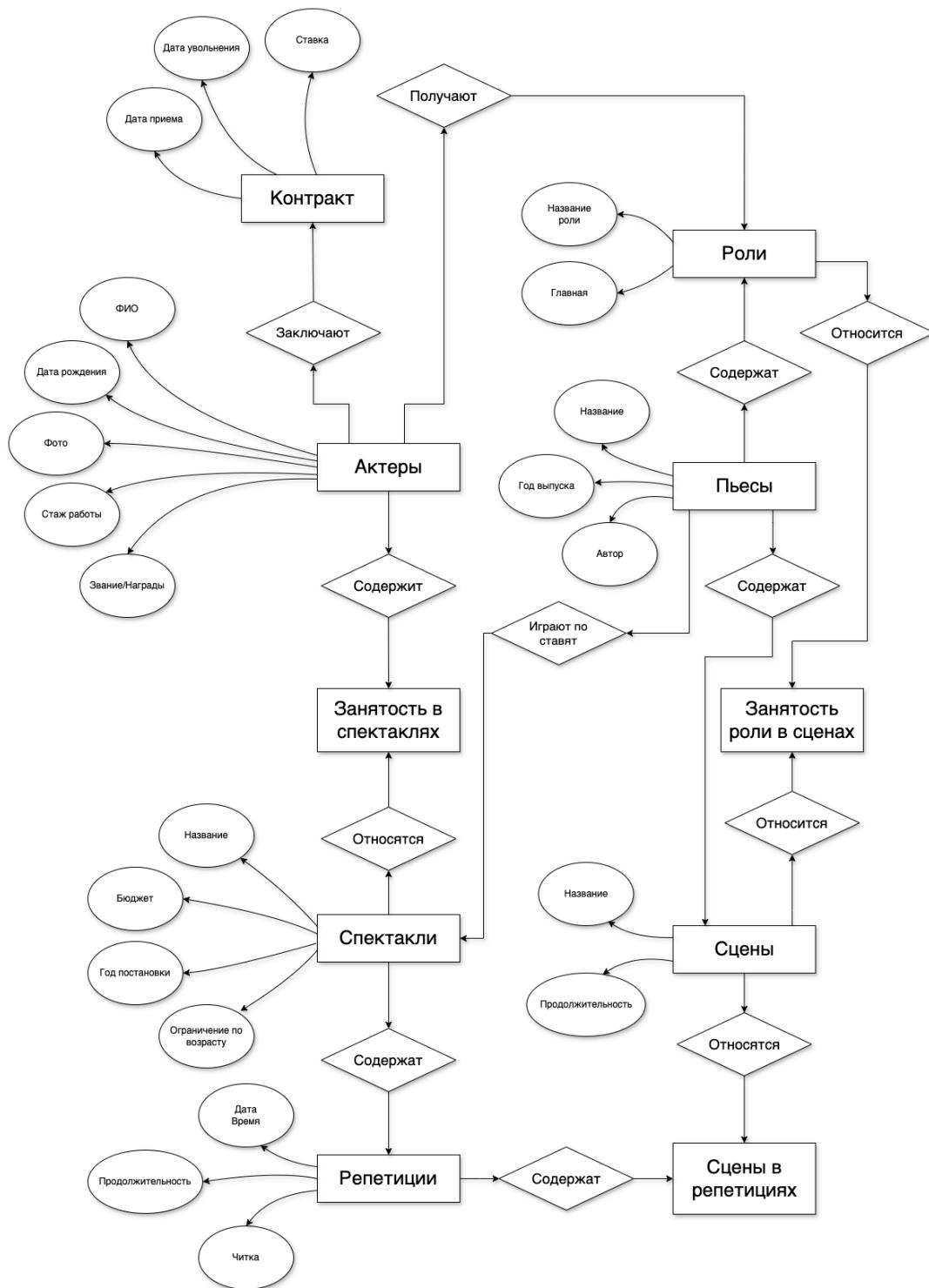


Рисунок 1 — Инфологическая модель базы данных

1.2 Датологическая модель базы данных

На основе анализа предметной области театра и выявленных ключевых сущностей и их взаимосвязей, можно построить ERD-диаграмму (датологическую модель базы данных).

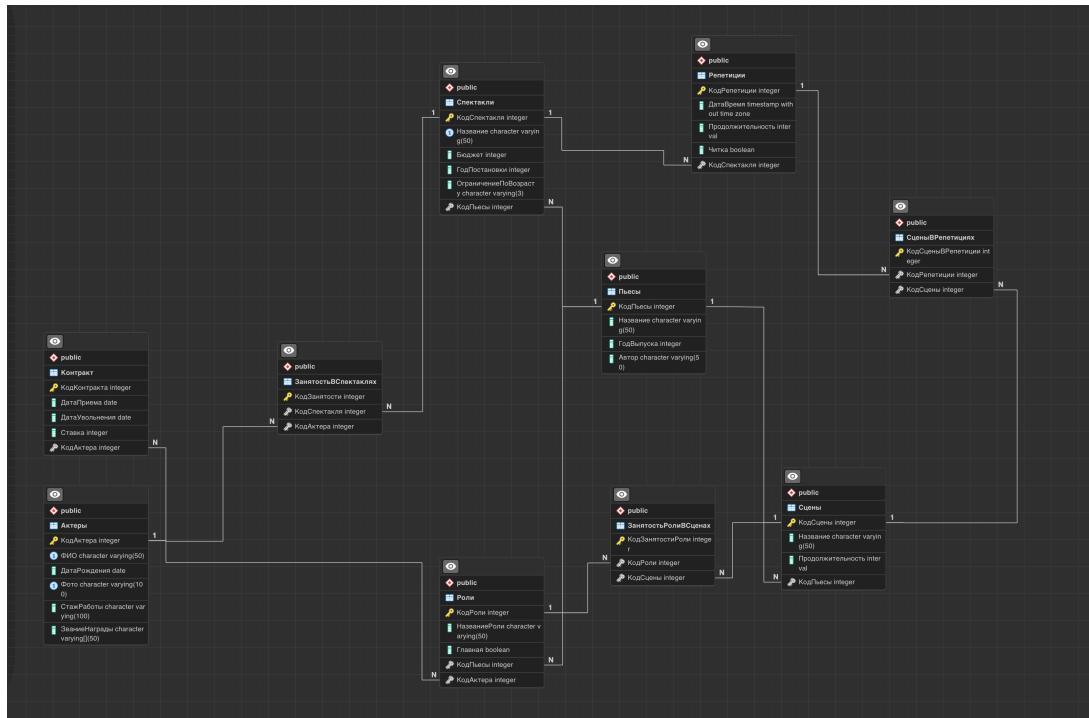


Рисунок 2 — Датологическая модель базы данных

На основе этой диаграммы создадим 10 таблиц.

1.3 Таблицы базы данных

Таблица 1 – Актеры

Название колонки	Описание	Тип
КодАктера	Код актера (первичный ключ)	SERIAL
ФИО	ФИО	VARCHAR(50)
ДатаРождения	Дата рождения	DATE
Фото	Фото	VARCHAR(100)
СтажРаботы	Стаж работы	VARCHAR(100)
ЗваниеНаграды	Звания и награды	VARCHAR(50) []

Таблица 2 – Спектакли

Название колонки	Описание	Тип
КодСпектакля	Код спектакля (первичный ключ)	SERIAL
Название	Название	VARCHAR(50)
Бюджет	Бюджет	INTEGER
ГодПостановки	Год постановки	INTEGER
ОграничениеПоВозрасту	Ограничение по возрасту	VARCHAR(3)
КодПьесы	Код пьесы	SERIAL

Таблица 3 – Контракт

Название колонки	Описание	Тип
КодКонтракта	Код контракта (первичный ключ)	SERIAL
ДатаПриема	Дата приема	DATE
ДатаУвольнения	Дата увольнения	DATE
Ставка	Ставка	INTEGER
КодАктера	Код актера (внешний ключ)	SERIAL

Таблица 4 – Занятость в спектаклях

Название колонки	Описание	Тип
КодЗанятости	Код занятости (первичный ключ)	SERIAL
КодСпектакля	Код спектакля (внешний ключ)	SERIAL
КодАктера	Код актера (внешний ключ)	SERIAL

Таблица 5 – Репетиции

Название колонки	Описание	Тип
КодРепетиции	Код репетиции (первичный ключ)	SERIAL
ДатаВремя	Дата и время	TIMESTAMP
Продолжительность	Продолжительность	INTERVAL
Читка	Читка	BOOLEAN
КодСпектакля	Код спектакля (внешний ключ)	SERIAL

Таблица 6 – Сцены в репетициях

Название колонки	Описание	Тип
КодСценыВРепетиции	Код сцены в репетиции (первичный ключ)	SERIAL
КодРепетиции	Код репетиции (внешний ключ)	SERIAL
КодСцены	Код сцены (внешний ключ)	SERIAL

Таблица 7 – Пьесы

Название колонки	Описание	Тип
КодПьесы	Код пьесы (первичный ключ)	SERIAL
Название	Название	VARCHAR(50)
ГодВыпуска	Год выпуска	INTEGER
Автор	Автор	VARCHAR(50)

Таблица 8 – Сцены

Название колонки	Описание	Тип
КодСцены	Код сцены (первичный ключ)	SERIAL
Название	Название	VARCHAR(50)
Продолжительность	Продолжительность	INTERVAL
КодПьесы	Код пьесы (внешний ключ)	SERIAL

Таблица 9 – Роли

Название колонки	Описание	Тип
КодРоли	Код роли (первичный ключ)	SERIAL
НазваниеРоли	Название роли	VARCHAR(50)
Главная	Главная роль	BOOLEAN
КодПьесы	Код пьесы (внешний ключ)	SERIAL
КодАктера	Код актера (внешний ключ)	SERIAL

Таблица 10 – Занятость роли в сценах

Название колонки	Описание	Тип
КодЗанятостиРоли	Код занятости роли в сценах (первичный ключ)	SERIAL
КодРоли	Код роли (внешний ключ)	SERIAL
КодСцены	Код сцены (внешний ключ)	SERIAL

1.4 Разработка бизнес-процессов предметной области

1. Регистрация нового актера

[Запрос на регистрацию на роль получен] → (Проверка полноты данных) → [Данные полные] → (Регистрация актера) → [Актер зарегистрирован] → (Заключение контракта с актером) → [Контракт заключен]

2. Перенос репетиций и спектакля

[Запрос на перенос получен] → (Изменение времени проведения спектакля) → [Спектакль перенесен] → (Проверка занятости актеров, с учетом занятости перенос репетиций) → [Репетиции перенесены]

3. Постановка спектакля

(Поиск пьесы) → [Пьеса найдена] → (Распределение ролей) → [Роли распределены] → (Читка) → [Читка проведена] → (Распределение сцен и репетиций) → [Сцены и репетиции распределены] → (Репетиции) → [Репетиции завершены] → (Назначение премьеры) → [Спектакль поставлен]

2 Реализация базы данных

2.1 Создание базы данных и таблиц

После анализа предметной области и создания датологической и инфологической моделей базы данных, приступим к ее реализации. Для начала создадим саму базу данных в PostgreSQL (листинг 1).

Листинг 1 — Создаем новую базу данных

```
CREATE DATABASE Занятость_актеров_театра
  WITH
    OWNER = pluttan
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1
    IS_TEMPLATE = False;
```

Разработаем, на основании выделенных сущностей, таблицы в PostgreSQL (листинги 2-11).

Листинг 2 — Таблица Актеры

```
CREATE TABLE Актеры (
    КодАктера      SERIAL PRIMARY KEY,
    ФИО            VARCHAR(50) UNIQUE NOT NULL,
    ДатаРождения   DATE NOT NULL,
    Фото           BYTEA UNIQUE NOT NULL,
    СтажРаботы     VARCHAR(100) NOT NULL,
    ЗваниеНаграды  VARCHAR(50)[] NOT NULL
);
```

Листинг 3 — Таблица Спектакли

```
CREATE TABLE Спектакли (
    КодСпектакля      SERIAL PRIMARY KEY,
    Название          VARCHAR(50) UNIQUE NOT NULL,
    Бюджет            INTEGER NOT NULL,
    ГодПостановки    INTEGER NOT NULL,
    ОграничениеПоВозрасту  VARCHAR(3) NOT NULL,
    КодПьесы          SERIAL NOT NULL
);
```

Листинг 4 — Таблица Контракт

```
CREATE TABLE Контракт (
    КодКонтракта      SERIAL PRIMARY KEY,
    ДатаПриема        DATE NOT NULL,
    ДатаУвольнения    DATE,
    Ставка            INTEGER NOT NULL,
    КодАктера          SERIAL NOT NULL
);
```

Листинг 5 — Таблица ЗанятостьВСпектаклях

```
CREATE TABLE ЗанятостьВСпектаклях (
    КодЗанятости      SERIAL PRIMARY KEY,
    КодСпектакля      SERIAL NOT NULL,
    КодАктера          SERIAL NOT NULL
);
```

Листинг 6 — Таблица Репетиции

```
CREATE TABLE Репетиции (
    КодРепетиции      SERIAL PRIMARY KEY,
    ДатаВремя          TIMESTAMP NOT NULL,
    Продолжительность INTERVAL NOT NULL,
    Читка              BOOLEAN NOT NULL,
    КодСпектакля      SERIAL NOT NULL
);
```

Листинг 7 — Таблица СценыВРепетициях

```
CREATE TABLE СценыВРепетициях (
    КодСценыВРепетиции SERIAL PRIMARY KEY,
    КодРепетиции        SERIAL NOT NULL,
    КодСцены            SERIAL NOT NULL
);
```

Листинг 8 — Таблица Пьесы

```
CREATE TABLE Пьесы (
    КодПьесы      SERIAL PRIMARY KEY,
    Название       VARCHAR(50) NOT NULL,
    ГодВыпуска    INTEGER NOT NULL,
    Автор          VARCHAR(50) NOT NULL
);
```

Листинг 9 — Таблица Сцены

```
CREATE TABLE Сцены (
    КодСцены           SERIAL PRIMARY KEY,
    Название           VARCHAR(50) NOT NULL,
    Продолжительность INTERVAL NOT NULL,
    КодПьесы          SERIAL NOT NULL
);
```

Листинг 10 — Таблица Роли

```
CREATE TABLE Роли (
    КодРоли      SERIAL PRIMARY KEY,
    НазваниеРоли VARCHAR(50) NOT NULL,
    Главная      BOOLEAN NOT NULL,
    КодПьесы    SERIAL NOT NULL,
    КодАктера   SERIAL NOT NULL
);
```

Листинг 11 — Таблица ЗанятостьРолиВСценах

```
CREATE TABLE ЗанятостьРолиВСценах (
    КодЗанятостиРоли SERIAL PRIMARY KEY,
    КодРоли           SERIAL NOT NULL,
    КодСцены          SERIAL NOT NULL
);
```

2.2 Связывание таблиц

После создания таблиц необходимо вторичные ключи связать с первичными, для этого немного изменим таблицы (Листинги 12-24):

Листинг 12 — Добавление внешнего ключа, связывающего актеров и контракты

```
ALTER TABLE Контракт
ADD CONSTRAINT fk_Контракт_Актеры FOREIGN KEY (КодАктера)
REFERENCES Актеры (КодАктера);
```

Листинг 13 — Добавление внешнего ключа, связывающего актеров и их занятость в спектаклях

```
ALTER TABLE ЗанятостьВСпектаклях
ADD CONSTRAINT fk_ЗанятостьВСпектаклях_Актеры FOREIGN KEY (КодАктера)
REFERENCES Актеры (КодАктера);
```

Листинг 14 — Добавление внешнего ключа, связывающего спектакли и занятость актеров в них

```
ALTER TABLE ЗанятостьВСпектаклях
ADD CONSTRAINT fk_ЗанятостьВСпектаклях_Спектакли FOREIGN KEY
(КодСпектакля)
REFERENCES Спектакли (КодСпектакля);
```

Листинг 15 — Добавление внешнего ключа, связывающего спектакли и репетиции

```
ALTER TABLE Репетиции
ADD CONSTRAINT fk_Репетиции_Спектакли FOREIGN KEY (КодСпектакля)
REFERENCES Спектакли (КодСпектакля);
```

Листинг 16 — Добавление внешнего ключа, связывающего сцены в репетициях и репетиции

```
ALTER TABLE СценыВРепетициях
ADD CONSTRAINT fk_СценыВРепетициях_Репетиции FOREIGN KEY
(КодРепетиции)
REFERENCES Репетиции (КодРепетиции);
```

Листинг 17 — Добавление внешнего ключа, связывающего сцены в репетициях и сцены

```
ALTER TABLE СценыВРепетициях
ADD CONSTRAINT fk_СценыВРепетициях_Сцены FOREIGN KEY (КодСцены)
REFERENCES Сцены (КодСцены);
```

Листинг 18 — Добавление внешнего ключа, связывающего пьесы и сцены

```
ALTER TABLE Сцены
ADD CONSTRAINT fk_Сцены_Пьесы FOREIGN KEY (КодПьесы)
REFERENCES Пьесы (КодПьесы);
```

Листинг 19 — Добавление внешнего ключа, связывающего пьесы и роли

```
ALTER TABLE Роли
ADD CONSTRAINT fk_Роли_Пьесы FOREIGN KEY (КодПьесы)
REFERENCES Пьесы (КодПьесы);
```

Листинг 20 — Добавление внешнего ключа, связывающего актеров и роли

```
ALTER TABLE Роли
ADD CONSTRAINT fk_Роли_Актеры FOREIGN KEY (КодАктера)
REFERENCES Актеры (КодАктера);
```

Листинг 21 — Добавление внешнего ключа, связывающего занятость роли в сценах и роли

```
ALTER TABLE ЗанятостьРолиВСценах
ADD CONSTRAINT fk_ЗанятостьРолиВСценах_Роли FOREIGN KEY (КодРоли)
REFERENCES Роли (КодРоли);
```

Листинг 22 — Добавление внешнего ключа, связывающего занятость роли в сценах и сцены

```
ALTER TABLE ЗанятостьРолиВСценах
ADD CONSTRAINT fk_ЗанятостьРолиВСценах_Сцены FOREIGN KEY (КодСцены)
REFERENCES Сцены (КодСцены);
```

Листинг 23 — Добавление внешнего ключа, связывающего пьесы и спектакли

```
ALTER TABLE Спектакли
ADD CONSTRAINT fk_Спектакли_Пьесы FOREIGN KEY (КодПьесы)
REFERENCES Пьесы (КодПьесы);
```

2.3 Заполнение базы данных

Для начала необходимо продумать заполнение базы данных. Основной таблицей базы данных будет таблица занятости актеров в спектаклях – в ней будет больше всего записей. По условию, данных записей должно быть не меньше миллиона. Количество актеров возьмем меньшее – 1000 человек. Таким образом контрактов тоже должно быть не меньше 1000 (предположим, что у всех

актеров один контракт – контрактов тоже будет ровно 1000). Количество спектаклей возьмем побольше – 2000 записей. Для одного спектакля в среднем требуется 30-40 репетиций, поэтому количество репетиций возьмем равным 80000. Количество сцен возьмем в 2 раза меньше – 40000, а количество пьес пусть будет равно количеству спектаклей – 2000. Количество ролей в спектакле совпадет с количеством занятости актеров, так как на одну роль предусмотрен один актер, то есть ролей тоже будет 1000000.

Для данного набора данных логическая составляющая все еще не очень верная, так как актер не может играть в 1000 спектаклях. Но если увеличивать количество актеров хотя бы до 100000 база данных будет весить около 60 Гб. Поэтому возьмем количество актеров меньшее, чем того требует логика, для тестирования хранения и управления данными этого хватит.

Перед тем, как перейти к заполнению таблиц необходимо разобраться с фото. Основных способов хранения фото в БД всего 2:

1. Хранение фото на внешнем носителе/в интернете, а в базе только ссылки/пути, таким образом сама база будет весить меньше, но это будет требовать дополнительной обработки, постоянный доступ к ресурсам, то есть уменьшаются централизованность данных

2. Хранение фото в виде бинарной строки непосредственно в базе данных. Этот способ позволяет хранить фото непосредственно внутри базы.

Так как фото будут храниться в небольшом разрешении и для повышения централизованности данных я выбрал 2 способ хранения.

Сгенерировать осмысленные текстовые данные с помощью кода достаточно просто, но генерация картинок требует большой вычислительной мощности и хорошего обучения, поэтому для генерации фотографий актеров воспользуемся интернет-сервисом по генерации фото несуществующих людей <https://thispersondoesnotexist.com/>. Сервис не имеет защиты от ботов, поэтому напишем код, который 1000 раз обновит страницу и сохранит все 1000 сгенерированных людей в одну папку (Приложение А).

Добавление больших данных в базу требует небольшой подготовки. Необходимо отключить всю индексацию для столбца. Для этого выполним: `ALTER TABLE "Актеры" DROP CONSTRAINT "Актеры_Фото_key";` Теперь индексация для столбца Фото производиться не будет.

Общее время, которое ушло на генерацию 1000 фото заняло около 15 минут, при учете того, что в коде есть тайм-аут, чтобы не слишком сильно нагружать сайт.

Вот пример сгенерированного фото.

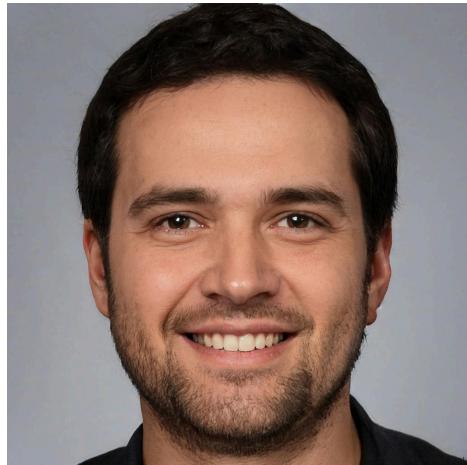


Рисунок 3 — Пример сгенерированного изображения

После генерации всех картинок остается только написать код для генерации остальных данных и вставки всех данных в БД.

Тут необходимо выбрать какой именно язык программирования использовать для наилучшей генерации данных. Я остановился на python, так как в библиотеке faker уже встроен русский язык. При такой генерации была обнаружена одна проблема: нельзя генерировать названия пьес и спектаклей. Для начала, как названия пьес и спектаклей, я взял просто обычные слова. Таким образом удалось за несколько минут сгенерировать и загрузить в базу данных основной набор данных. Так как данные грузились не равномерно во все таблицы, коды, которые представлены в листингах 12-23, я выполнил только после окончания генерации. (Приложение Б)

Но названия пьес и спектаклей все еще не были осмысленными. Поэтому я решил обратиться к библиотеке faker, написанной для ruby. В этой библиотеке нет русского языка, но есть генерация названий книг, поэтому я сгенерировал все названия пьес и спектаклей и внес их в таблицу. (Приложение В)

Так как названия были на английском их оставалось только перевести на русский. Поэтому был написан код, который делает именно это. (Приложение Г)

Итак, вот заполненные таблицы:

1	select * from Актеры				
	Data Output Сообщения Notifications				
	ФИО	датарождения	дата	СправоРолы	Знамениты
1	Денисовна Родион Федоровна	1988-05-19	Бюджетная	25 years	
2	Василий Всеволодович Попов	1993-05-04	Бюджетная	3 years	[Лучшая роль в главной роли];[Лучшая роль второго плана]
3	Феликс Валентин Димитров	2000-11-20	Бюджетная	42 years	[Лучшее исполнение в музыкальном видео];[Лучший актерский ансамбль];[Лучшая женская роль]
4	Мухин Илья Владиславович	1996-09-27	Бюджетная	24 years	0
5	Марина Кузьминична Могилова	1980-11-22	Бюджетная	5 years	[Лучшая женская роль];[Лучший детский актер];[Лучшее исполнение в кинематографии]
6	Ольга Николаевна Кузьмина	2003-01-28	Бюджетная	20 years	[Лучшее исполнение в отечественном фильме]
7	Широбек Айна Турсунова	1993-06-05	Бюджетная	46 years	[Лучший город];[Лучшее голосовое исполнение];[Лучший мужской актор];[Лучшая женская роль]
8	Андрей Евгеньевич Борисов	1973-07-01	Бюджетная	9 years	[Лучшее исполнение в женской роли]
9	Кирилла Мария Болеславовна	2000-06-19	Бюджетная	23 years	[Лучший актер];[Лучшее исполнение в филмографии];[Лучшее исполнение в триллере]
10	Константин Павлович Качалов	1982-10-08	Бюджетная	4 years	
11	Куликова Валерия Вениаминовна	1965-07-15	Бюджетная	40 years	[Лучший актер старшего возраста]
12	Пахомова Альбина Николаевна	1993-12-08	Бюджетная	13 years	0
13	Соловьев Платон Винентиевич	1989-09-05	Бюджетная	44 years	[Лучшее исполнение в мюзикле];[Лучшая мужская роль];[Лучшее исполнение в телевизионном фильме]
14	Азарий Ефимович Егоров	2001-09-03	Бюджетная	1 years	0
15	Вадима Павловна Гоголева	1989-10-18	Бюджетная	31 years	0
16	Эмира Ахметовна Харитонова	2003-04-20	Бюджетная	43 years	0
17	Ерофеев Евгений Николаевич	1981-09-07	Бюджетная	32 years	0
18	Кононенко Карл Артурович	1996-09-11	Бюджетная	16 years	0
19	Любовь Николаевна Смирнова	1979-01-01	Бюджетная	40 years	[Лучшая комедия]
20	Зинаида Геннадьевна Соколова	1981-11-20	Бюджетная	41 years	[Лучшее исполнение в короткометражке];[Лучший продюсер];[Лучший год];[Премия за вклад в искусство]
21	Назарова Нинель Леонидовна	1967-11-29	Бюджетная	29 years	[Лучшее исполнение в музыкальном видео];[Лучший подростковый актер];[Лучшая роль второго плана]
22	Анжела Борисовна Белозерова	1966-08-29	Бюджетная	41 years	0
23	Наталья Борисовна Уварова	1959-02-24	Бюджетная	22 years	[Премия критиков]
24	Костина Наталья Филипповна	1958-09-01	Бюджетная	46 years	
25	Ангелина Георгиевна Ильинова	2000-09-06	Бюджетная	7 years	[Лучший актер озвучивания];[Лучшее исполнение в научно-фантастической роли];[Лучшее исполнение в мюзикле]
26	Терентий Тимофеев Тимофей	1997-04-21	Бюджетная	11 years	
27	Глебников Карл Егорович	1986-07-28	Бюджетная	6 years	[Лучший продюсер];[Лучшее голосовое исполнение];[Лучшее исполнение в мюзикле]
28	Ольга Юриевна Корнилова	1972-05-04	Бюджетная	41 years	0
29	Бекетов Курман Бекетрович	2008-05-08	Бюджетная	40 years	[Лучшая женская роль];[Лучший подростковый актер];[Лучший мужской актер]
30	Павелко Геннадий Геннадьевич	2005-05-28	Бюджетная	5 years	[Премия критиков сценаристов]

Total rows: 1000 of 1000 Data Output Сообщения Notifications

Ln 1, Col 21

Рисунок 4 — Результат заполнения таблицы Актеры

1	select * from ЗанятостьВСпектаклях		
	Data Output Сообщения Notifications		
	КодЗанятости [PK] integer	КодСпектакля integer	КодАктера integer
1	500241	1094	783
2	500242	1609	515
3	500243	265	176
4	500244	673	673
5	500245	1817	804
6	500246	1581	986
7	500247	996	662
8	500248	796	259
9	500249	1409	102
10	500250	577	75
11	500251	1207	617
12	500252	627	71
13	500253	1067	837
14	500254	262	500
15	500255	1425	559
16	500256	653	24
17	500257	331	69
18	500258	1655	148
19	500259	1628	23
20	500260	932	48
21	500261	274	606
22	500262	1981	249
23	500263	1626	489
24	500264	101	47
25	500265	1188	128
26	500266	23	493
27	500267	1176	24
28	500268	1801	601
29	500269	1891	10
30	500270	1767	787
31	500271	640	951

Total rows: 1000 of 1000000 Data Output Сообщения Notifications

Query complete 00:00:00.585

Рисунок 5 — Результат заполнения таблицы ЗанятостьВСпектаклях

1	<code>select * from ЗанятостьРолиВСценах</code>	Data Output	Сообщения	Notifications
	КодЗанятостиРоли [PK] integer	КодРоли integer	КодСцены integer	
1	23681	536880	15168	
2	23682	392336	20530	
3	23683	832104	5967	
4	23684	138742	11677	
5	23685	304556	33228	
6	23686	680767	39966	
7	23687	581908	1699	
8	23688	361873	31097	
9	23689	631627	5898	
10	23690	924556	4720	
11	23691	646204	10291	
12	23692	284139	26718	
13	23693	65056	2159	
14	23694	467568	39531	
15	23695	453849	24867	
16	23696	539938	13906	
17	23697	206575	27631	
18	23698	704990	24392	
19	23699	810186	11876	
20	23700	473898	11588	
21	23701	866599	34242	
22	23702	886303	1345	
23	23703	662811	36769	
24	23704	4347	33170	
25	23705	107928	2116	
26	23706	714093	11386	
27	23707	709320	13881	
28	23708	989240	23073	
29	23709	765258	37012	
30	23710	688977	8732	
31	23711	050560	27260	
Total rows: 1000 of 1000000			Query complete 00:00:00.427	

Рисунок 6 — Результат заполнения таблицы ЗанятостьРолиВСценах

1 select * from Контракт					
Data Output		Сообщения		Notifications	
	КодКонтракта [PK] integer	ДатаПриема date	ДатаУвольнения date	Ставка integer	КодАктера integer
1	1	2020-10-09	2026-01-06	57734	1
2	2	2021-01-26	2025-03-15	69185	2
3	3	2017-01-11	2029-04-21	41804	3
4	4	2007-10-16	2026-11-29	35428	4
5	5	2007-05-09	2025-08-18	55524	5
6	6	2023-09-27	2028-05-01	60059	6
7	7	2021-05-05	2031-04-03	40721	7
8	8	2024-05-11	2024-12-03	80463	8
9	9	2018-11-09	2031-02-03	48470	9
10	10	2007-03-05	2031-01-12	58637	10
11	11	2017-01-04	2026-10-21	65286	11
12	12	2019-02-05	2028-03-04	65290	12
13	13	2017-05-03	2029-01-18	89051	13
14	14	2021-10-08	2024-10-03	95070	14
15	15	2016-05-04	2027-11-13	77683	15
16	16	2014-04-13	2026-11-05	44111	16
17	17	2017-08-26	2029-11-28	58128	17
18	18	2008-11-13	2031-03-13	67638	18
19	19	2023-05-24	2027-06-04	96033	19
20	20	2013-07-07	2025-07-16	33808	20
21	21	2019-10-07	2028-10-18	74324	21
22	22	2012-06-17	2029-12-05	38800	22
23	23	2015-11-27	2030-07-10	31203	23
24	24	2022-03-14	2025-01-07	48915	24
25	25	2024-01-17	2029-07-09	86232	25
26	26	2021-01-03	2027-02-27	33561	26
27	27	2022-02-25	2031-05-25	40452	27
28	28	2012-03-14	2029-07-18	54582	28
29	29	2012-07-18	2027-09-24	34710	29
30	30	2021-04-18	2025-06-24	92301	30

Total rows: 1000 of 1000 | Query complete 00:00:00.142

Рисунок 7 — Результат заполнения таблицы Контракт

1 select * from Пьесы			
Data Output		Сообщения	
	КодПьесы [PK] integer	Название character varying (50)	ГодВыпуска integer
1	1082	К востоку от рая	1991 Макар Бенедиктович Мартынов
2	4	Это поле битвы	1968 Гордеева Фаина Яковлевна
3	5	Таковы были радости	2008 Любовь Святославовна Федотова
4	6	Чудесающий полк женщин	1999 Дроздова Лара Кузьминична
5	7	Я пою тело электричество	1925 Ульян Федорьевич Белоусов
6	8	Некоторые похороны Цезаря	1916 Гусева Марфа Архиповна
7	9	Злой колыбель	1929 Юлиан Давыдович Кошелев
8	10	Сканер мрачно	1984 Руслан Харитонович Самсонов
9	11	Его темные материалы	1908 Игнатьев Тарас Бориславович
10	12	Время подарков	1913 Никифор Маркович Алексеев
11	13	Удивлен радостью	2005 г-н Бобров Виктор Иосилович
12	14	Зеркало взломано из стороны в сторону	1981 Корин Эдгардович Колобов
13	15	Нектар в сите	1962 Абрамова Мария Игоревна
14	16	Один на широком, широком море	1901 Сидор Бенедиктович Савельев
15	17	Кольцо яркой воды	1928 Морозова Елена Юрьевна
16	18	Дорога меньше путешествовала	1951 Степанов Евдокия Тимуровна
17	19	Злой колыбель	1980 Фирс Адрианович Матвеев
18	20	Гордость пыли	1957 Козлова Ксения Богдановна
19	21	Ах, дикая местность!	1950 Григорьев Мир Геннадиевич
20	22	Злой колыбель	1992 Савва Измаклович Шарапов
21	23	Дорога меньше путешествовала	1977 Казаков Никанд Аркадьевич
22	24	Я знаю, почему поет птица в клетке	1910 Горбунова Наталья Макаровна
23	25	Прощай оружие	1934 Парамон Трофимович Даляков
24	26	другая сторона молчания	1913 Князев Пилипарт Богданович
25	27	Quo Vadis	1942 Ермил Демьянович Симонов
26	28	Слава - это шпора	1900 Дроздова Ирина Мироновна
27	29	Желтые лучи эсфодели	1904 Твердышев Илья Красильников
28	30	Человек внутри	2017 Денисов Михаил Ильсович
29	31	За пределами залива Мексики	1961 Емельян Игоревич Кудряшов
30	32	Богу неизвестно	1943 Клавдия Константиновна Овчинникова

Total rows: 1000 of 2000 | Query complete 00:00:00.060

Рисунок 8 — Результат заполнения таблицы Пьесы

1 select * from Репетиции					
	ДатаВремя timestamp without time zone	Продолжительность interval	Читка boolean	КодСпектакля integer	КодРепетиции [PK] integer
1	2022-03-24 20:01:38	01:16:00	true	758	1
2	2020-07-14 10:31:04	01:15:00	false	1123	2
3	2010-11-01 04:04:11	01:47:00	true	1046	3
4	2015-04-09 04:17:46	03:02:00	false	1812	4
5	2013-12-21 00:36:16	01:42:00	true	311	5
6	2007-01-19 18:23:29	03:07:00	false	844	6
7	2006-11-04 11:31:26	02:43:00	true	1191	7
8	2004-11-01 19:21:10	01:43:00	true	889	8
9	2017-12-15 04:15:10	03:07:00	true	1776	9
10	2007-01-29 06:46:23	02:13:00	false	1669	10
11	2004-06-07 07:20:06	04:09:00	false	1512	11
12	2004-12-08 22:51:17	04:54:00	false	1997	12
13	2020-11-05 04:40:08	02:00:00	false	34	13
14	2013-04-15 02:36:54	03:42:00	true	1377	14
15	2004-11-17 12:18:37	04:03:00	true	530	15
16	2007-12-30 14:35:25	02:55:00	false	1067	16
17	2023-09-17 21:08:09	04:54:00	false	1173	17
18	2016-01-08 13:25:44	03:08:00	true	77	18
19	2015-05-27 16:32:15	01:42:00	false	1570	19
20	2016-02-14 19:17:19	01:25:00	true	278	20
21	2020-12-22 04:14:01	01:31:00	false	54	21
22	2018-09-28 01:42:19	03:04:00	true	1830	22
23	2011-10-23 03:57:38	04:03:00	false	1485	23
24	2020-03-18 08:51:13	03:46:00	true	1987	24
25	2017-10-03 02:12:51	02:35:00	false	363	25
26	2013-11-09 19:44:51	03:09:00	false	1534	26
27	2015-05-08 11:04:21	02:07:00	false	1159	27
28	2022-11-22 17:27:00	02:21:00	false	1001	28
29	2016-01-24 21:18:16	02:05:00	false	1832	29
30	2014-01-31 01:52:00	02:03:00	true	74	30
31	2000-11-11 12:04:50	01:10:00	false	1020	31
Total rows: 1000 of 80000		Query complete 00:00:00.144			

Рисунок 9 — Результат заполнения таблицы Репетиции

1 select * from Роли					
	КодРоли [PK] integer	НазваниеРоли character varying (50)	Главная boolean	КодПьесы integer	КодАктера integer
1	507748	Ладислав Суханова	false	1563	331
2	507749	Олег Блинов	false	943	279
3	507750	Христофор Мамонтова	false	1018	448
4	507751	Софон Рогов	false	938	991
5	507752	Елизавета Константинов	true	288	773
6	507753	Леон Мартынов	true	1439	573
7	507754	Андрей Капустина	false	818	480
8	507755	Поликарп Шарова	false	54	314
9	507756	Егор Котов	true	767	116
10	507757	Дмитрий Голубев	true	1193	502
11	507758	Владислав Матвеева	true	1961	205
12	507759	тov. Измаил Григорьев	true	1486	552
13	507760	Ратмир Макарова	false	462	256
14	507761	Евпраксия Якушева	false	1172	519
15	507762	Андрей Лаврентьев	false	143	791
16	507763	г-н Федосий Орехова	true	888	650
17	507764	г-н Мефодий Нестерова	false	1927	524
18	507765	Рубен Ершов	true	952	312
19	507766	Варвара Самойлов	true	706	927
20	507767	Сибирский Третьяков	false	1142	191
21	507768	Аким Цветков	true	989	86
22	507769	тov. Зоя Кулакова	false	781	374
23	507770	Феоктист Воронцов	true	778	328
24	507771	Марianne Белозеров	false	564	530
25	507772	Изяслав Симонов	true	184	534
26	507773	тov. Савва Большакова	false	731	636
27	507774	Софон Гуляева	true	636	890
28	507775	Юлий Романов	false	1283	144
29	507776	Лазарь Соболев	true	658	50
30	507777	Татьяна Калинина	false	46	236
31	507778	Елена Каша	false	1200	170
Total rows: 1000 of 1000000		Query complete 00:00:00.708			

Рисунок 10 — Результат заполнения таблицы Роли

	Название character varying (50)	Бюджет integer	ГодПостановки integer	ОграничениеПоЗвраст character varying (3)	КодПьесы integer	КодСпектакля [PK] integer
1	В сухой сезон	2376	1940	17+	101	101
2	Движущийся палец	3207	2013	14+	102	102
3	Приимлемое время	3612	1927	5+	103	103
4	Бледные короли и принцы	3464	1990	17+	104	104
5	Эта спокойная ночь	6560	1975	5+	105	105
6	Француз Саган	2233	2012	14+	106	106
7	Я знаю, почему поет птица в камере	8085	2019	10+	107	107
8	Плыть за закатом	6130	1911	13+	108	108
9	Маленькие ручки хлопают в ладоши	1452	1992	5+	1093	1093
10	Какой-то похороненный Цезарь	6907	1936	14+	1094	1094
11	Все люди короли	4135	2015	17+	1096	1096
12	Облаха свидетелей	2852	1920	8+	1097	1097
13	Драгоценное проклятие	6957	1931	8+	1098	1098
14	По ком звонит колокол	7886	1943	0+	1099	1099
15	Его темные материалы	5688	1923	2+	1100	1100
16	Драгоценное проклятие	3675	1961	18+	1101	1101
17	Золотые альбомы Солнца	3758	1945	15+	1102	1102
18	Вот человек	3790	2008	7+	1103	1103
19	Его темные материалы	9288	1928	6+	1104	1104
20	О мышах и людях	4332	1935	7+	1105	1105
21	Мать Ночь	6580	1989	0+	1106	1106
22	Капуста и короли	5919	1914	6+	1107	1107
23	Античное сено	9422	1985	4+	1108	1108
24	Конфедерация туниц	9443	1972	14+	1109	1109
25	Эта сторона рай	7507	2020	10+	1110	1110
26	Под кровоточением	3663	1913	16+	1111	1111
27	Удивлен радостью	2555	1967	16+	1112	1112
28	Когда я умирал	6179	1924	18+	1113	1113
29	Инесенные ветром	8453	1933	14+	1114	1114
30	Не говори уже о собаке	2518	1974	0+	1115	1115
31	Собачиной языком	1820	1929	5+	1116	1116

Рисунок 11 — Результат заполнения таблицы Спектакли

	КодСцены [PK] integer	Название character varying (50)	Продолжительность interval	КодПьесы integer
1	1	Деньги.	00:26:00	744
2	2	Монета художественный волк.	00:29:00	602
3	3	Армейский заявление.	00:24:00	1132
4	4	Бригада.	00:05:00	70
5	5	Деньги опасность.	00:26:00	296
6	6	Птиро команда.	00:21:00	1369
7	7	Материя спичка тревога.	00:27:00	1000
8	8	Горкий засунуть.	00:10:00	48
9	9	Вскакивать деньги.	00:03:00	1800
10	10	Подробность функция увеличиваться.	00:11:00	828
11	11	Разнообразный юный.	00:25:00	289
12	12	Поколение конструкция.	00:09:00	1149
13	13	Заведение.	00:02:00	1581
14	14	Встать выраженный разуметься.	00:05:00	1589
15	15	Слати госпожа.	00:11:00	304
16	16	Добиться шлем тысяча.	00:11:00	991
17	17	Что монета направо.	00:06:00	1122
18	18	Еврейский некоторый.	00:12:00	749
19	19	Господь потянувшись сходит.	00:24:00	1922
20	20	А возмутиться.	00:26:00	632
21	21	Что монета направо.	00:08:00	1464
22	22	Выкинуть через.	00:22:00	721
23	23	Виднеться ботинок.	00:23:00	905
24	24	Ионч карандаш.	00:07:00	904
25	25	Выражение что.	00:13:00	559
26	26	Советовать.	00:28:00	1093
27	27	Ягода приятель при.	00:19:00	1448
28	28	Неправда.	00:15:00	236
29	29	Через ленинград командающий.	00:12:00	339
30	30	Смеяться.	00:19:00	1414
31	31	Собачиной языком	00:22:00	1774

Рисунок 12 — Результат заполнения таблицы Сцены

1 select * from СценыВРепетициях			
Data Output Сообщения Notifications			
	КодСценыВРепетиции [PK] integer	КодРепетиции integer	КодСцены integer
1		50188	4843
2		32343	22751
3		44181	19817
4		17546	12192
5		57372	15902
6		54247	16415
7		47573	3877
8		46115	34963
9		60061	12624
10		15060	16307
11		58331	14503
12		60967	2153
13		47709	20650
14		29759	3663
15		24713	37829
16		66703	20231
17		45208	17013
18		74518	18353
19		76244	22510
20		22939	6959
21		71885	39514
22		4267	30871
23		5715	18563
24		37461	10280
25		10091	25714
26		65749	14605
27		48986	28459
28		48220	6814
29		39775	26350
30		43691	6408
31		16277	28004
Total rows: 1000 of 80000		Query complete 00:00:00.101	

Рисунок 13 — Результат заполнения таблицы СценыВРепетициях

3 Запросы и триггеры

3.1 Запрос 1

Запрос выбирает актеров, которые заняты более чем в одном спектакле, имеют более трех наград и хотя бы одну премию. Возвращаются их ФИО, список премий, количество спектаклей и звание награды. (Листинг 25, результат на рис. 14)

Листинг 24 — Запрос 1

```
SELECT а.ФИО, премии.Премии, subquery.NumberOfPlays, а.ЗваниеНаграды
  FROM Актеры а
    JOIN (
        SELECT z.КодАктера, COUNT(z.КодСпектакля) AS NumberOfPlays
          FROM ЗанятостьВСпектаклях z
        GROUP BY z.КодАктера
        HAVING COUNT(z.КодСпектакля) > 1
    ) AS subquery ON а.КодАктера = subquery.КодАктера
    JOIN LATERAL (
        SELECT array_agg(award) AS Премии
          FROM unnest(а.ЗваниеНаграды) AS award
        WHERE award LIKE 'Премия%'
    ) AS премии ON true
    WHERE (
        SELECT COUNT(*)
          FROM unnest(а.ЗваниеНаграды) AS awards
    ) > 3
    AND EXISTS (
        SELECT 1
          FROM unnest(а.ЗваниеНаграды) AS award
        WHERE award LIKE 'Премия%'
    )
  ORDER BY subquery.NumberOfPlays DESC;
```

```

1 SELECT a.ФИО, премии.Премии, subquery.NumberOfPlays, a.ЗваниеНаграды
2   FROM Актеры a
3     JOIN (
4       SELECT z.КодАктера, COUNT(z.КодСпектакля) AS NumberOfPlays
5         FROM ЗанятостьВСпектаклях z
6        GROUP BY z.КодАктера
7        HAVING COUNT(z.КодСпектакля) > 1
8     ) AS subquery ON a.КодАктера = subquery.КодАктера
9   JOIN LATERAL (
10     SELECT array_agg(award) AS Премии
11       FROM unnest(a.ЗваниеНаграды) AS award
12      WHERE award LIKE 'Премия%'
13   ) AS премии ON true
14   WHERE (
15     SELECT COUNT(*)
16       FROM unnest(a.ЗваниеНаграды) AS awards
17   ) > 3
18 AND EXISTS (
19   SELECT 1
20     FROM unnest(a.ЗваниеНаграды) AS award
21    WHERE award LIKE 'Премия%'
22 )
23 ORDER BY subquery.NumberOfPlays DESC;

```

Data Output Сообщения Notifications

The screenshot shows a PostgreSQL query results window. The table has four columns: ФИО (Actor Name), Премии (Awards), numberofplays (Count of plays), and ЗваниеНаграды (Award Title). The results list 11 actors, each with their name, a list of awards they have won, the count of plays they have been in, and the specific award titles.

	ФИО character varying (50)	Премии character varying[]	numberofplays bigint	ЗваниеНаграды character varying[] (50)
1	Александров Зосима Харлампьевич	{'Премия критиков','Премия зрительских симпатий'}	1096	{'Премия критиков','Лучший подростковый актер','Лучшее исполнение в романтической роли','Лучше
2	Агафонов Парфен Владиленович	{'Премия зрительских симпатий'}	1089	{'Лучшее исполнение в мюзикле','Лучшее исполнение в спектакле','Лучший международный актер','Л
3	Татьяна Макаровна Васильева	{'Премия критиков'}	1083	{'Лучшее исполнение в комедийной роли','Лучший мужской актер','Премия критиков','Лучший герой',
4	Тихон Харламович Чернов	{'Премия за вклад в искусство'}	1074	{'Лучшая роль в главной роли','Лучшая роль второго плана','Лучшее исполнение в романтической ро
5	Исакова Ия Федоровна	{'Премия критиков'}	1073	{'Актер года','Премия критиков','Лучшее исполнение в документальном фильме','Лучшая женская ак
6	Воронцов Евстафий Зиновьевич	{'Премия за вклад в искусство'}	1065	{'Лучший драматург','Премия за вклад в искусство','Лучший актерский ансамбль','Лучшая комедийная рол
7	Агата Игоревна Белоzerosova	{'Премия за вклад в искусство'}	1059	{'Лучший подростковый актер','Лучшая драматическая роль','Лучшая женская роль','Премия за вкла
8	Борисов Иван Елизарович	{'Премия зрительских симпатий'}	1058	{'Лучшая новая звезда','Лучший герой','Актер года','Премия зрительских симпатий'}
9	Кошелев Корнил Ефимьевич	{'Премия за вклад в искусство'}	1057	{'Лучший трансгендерный актер','Лучшее исполнение в триллере','Актер года','Премия за вклад в ис
10	Прокл Андриевич Николаев	{'Премия критиков'}	1054	{'Премия критиков','Лучшее исполнение в телесериале','Лучшее исполнение в боевой роли','Лучшее
11	Ян Даниилович Лавылов	{'Премия за вклад в искусство'}	1049	{'Лучшее исполнение в научно-фантастической роли','Лучшее исполнение в вестерне','Лучшее исполн

Total rows: 220 of 220 | Query complete 00:00:00.358 | Ln 23, Col 38

Рисунок 14 — Результат выполнения 1 запроса

Используемые функции и концепции:

- **JOIN**: Объединение таблиц для получения данных.
- **GROUP BY** и **HAVING**: Группировка данных и фильтрация групп.
- **ARRAY_AGG** и **UNNEST**: Работа с массивами в PostgreSQL.
- **EXISTS**: Проверка наличия записей.

3.2 Запрос 2

Запрос выводит информацию об актерах, играющих главную роль в спектакле с бюджетом выше среднего и имеющих больше 5 сцен в этом спектакле, включая название спектакля, бюджет, ФИО актера, количество главных ролей и количество сцен в этих ролях. (Листинг 26, результат на рис. 15)

Листинг 25 — Запрос 2

```
SELECT DISTINCT s.Название AS НазваниеСпектакля,
               s.Бюджет,
               a.ФИО,
               COUNT(r.КодРоли) AS КоличествоГлавныхРолей,
               сцены.КоличествоСцен
        FROM Спектакли s
      JOIN ЗанятостьСпектаклях z ON s.КодСпектакля = z.КодСпектакля
      JOIN Актеры a ON z.КодАктера = a.КодАктера
      JOIN Роли r ON a.КодАктера = r.КодАктера
                    AND r.КодПьесы = s.КодПьесы
                    AND r.Главная = TRUE
      JOIN (
            SELECT sr.КодРоли, COUNT(sr.КодСцены) AS КоличествоСцен
                  FROM ЗанятостьРолиВСценах sr
                  GROUP BY sr.КодРоли
                  HAVING COUNT(sr.КодСцены) > 5
          ) AS сцены ON сцены.КодРоли = r.КодРоли
      WHERE s.Бюджет > (
            SELECT AVG(Бюджет)
                  FROM Спектакли
          )
      GROUP BY s.Название, s.Бюджет, a.ФИО, сцены.КоличествоСцен
      ORDER BY s.Бюджет DESC, сцены.КоличествоСцен DESC;
```

```

1 SELECT DISTINCT s.Название AS НазваниеСпектакля, s.Бюджет, a.ФИО, COUNT(r.КодРоли) AS КоличествоГлавныхРолей, сцены.КоличествоСцен
2   FROM Спектакли s
3   JOIN ЗанятостьВСпектаклях z ON s.КодСпектакля = z.КодСпектакля
4   JOIN Актеры a ON z.КодАктера = a.КодАктера
5   JOIN Роли r ON a.КодАктера = r.КодАктера AND r.КодПьесы = s.КодПьесы AND r.Главная = TRUE
6   JOIN (
7     SELECT sr.КодРоли, COUNT(sr.КодСцены) AS КоличествоСцен
8       FROM ЗанятостьРолиВСценах sr
9      GROUP BY sr.КодРоли
10     HAVING COUNT(sr.КодСцены) > 5
11   ) AS сцены ON сцены.КодРоли = r.КодРоли
12 WHERE s.Бюджет > (
13   SELECT AVG(Бюджет)
14     FROM Спектакли
15   )
16 GROUP BY s.Название, s.Бюджет, a.ФИО, сцены.КоличествоСцен
17 ORDER BY s.Бюджет DESC, сцены.КоличествоСцен DESC;

```

Data Output Сообщения Notifications

НазваниеСпектакля character varying (50)	Бюджет integer	ФИО character varying (50)	КоличествоГлавныхРолей bigint	КоличествоСцен bigint
Скачай в темноте	9970	Исаева Зоя Альбертовна	1	6
Эго Доминус Тус	9965	Зимин Бронислав Чеславович	1	6
Череп под кожей	9922	Кошелева Виктория Романовна	1	6
Эта отвратительная сила	9885	Овчинников Григорий Харламьевич	2	7
Темная равнина	9816	Лукин Ефимовна Веселова	1	7
Тигр! Тигр!	9778	Егорова Екатерина Ниловна	1	6
Время убивать и творить	9679	Евпраксия Болеславовна Мартынова	2	6
Где ангелы боятся идти	9575	Лазарев Иван Адрианович	2	6
Когда смеется зелёный лес	9567	Лобanova Леонора Станиславовна	3	6
Переход в Индию	9510	Киселева Екатерина Тимуровна	1	6
Какой-то похороненный Цезарь	9370	Дячкова Виктория Богдановна	1	6
Время нашей Тьмы	9286	Евгения Валерьевна Дячкова	1	6
Старикам тут не место	9286	Филатов Прокофий Таймуразович	1	6
Все люди короля	9208	Королов Всемил Жоресович	1	6
В темном стакане	9205	Степанов Нестор Гертрудович	1	6
Золотые блоки Солнца	9161	Мурзяев Андрей Гурьевич	3	6
Ой! Быть в Англии	8990	Савелий Власович Комисаров	2	6
Приключения с...	8990	Елизавета Виталий Борисович	1	7

Total rows: 66 of 66 | Query complete 00:00:01.366 | Ln 17, Col 51

Рисунок 15 — Результат выполнения 2 запроса

Используемые функции и концепции:

- **JOIN**: Объединение таблиц для получения данных.
- **GROUP BY**: Группировка данных.
- **HAVING**: Фильтрация групп.
- **COUNT**: Подсчет записей.
- **AVG**: Вычисление среднего значения.
- **ORDER BY**: Сортировка результатов.

3.3 Запрос 3

Запрос выводит информацию о действующих контрактах актеров, включая ФИО актера, дату подписания текущего контракта и список спектаклей, в которых актеры заняты. (Листинг 27, результат на рис. 16)

Листинг 26 — Запрос 3

```
WITH CurrentContracts AS (
    SELECT k.КодАктера, MAX(k.ДатаПриема) AS ДатаПриема
    FROM Контракт k
    WHERE k.ДатаУвольнения IS NULL OR k.ДатаУвольнения > CURRENT_DATE
    GROUP BY k.КодАктера), ActorPlays AS (
    SELECT z.КодАктера, ARRAY_AGG(s.Название) AS Спектакли
    FROM ЗанятостьВСпектаклях z
    JOIN Спектакли s ON z.КодСпектакля = s.КодСпектакля
    GROUP BY z.КодАктера)
SELECT а.ФИО, cc.ДатаПриема AS ДатаПодписанияДействующегоКонтракта, ap.Спектакли
FROM Актеры а
JOIN CurrentContracts cc ON а.КодАктера = cc.КодАктера
JOIN ActorPlays ap ON а.КодАктера = ap.КодАктера
ORDER BY а.КодАктера;
```

Рисунок 16 — Результат выполнения 3 запроса

Используемые функции и концепции:

- **WITH**: Создание временных таблиц для упрощения запроса.
 - **ARRAY AGG**: Агрегация значений в массив.
 - **JOIN**: Объединение таблиц для получения данных.
 - **ORDER BY**: Сортировка результатов.

3.4 Запрос 4

Запрос выводит информацию о пьесах, включая название пьесы, название спектакля, количество наград и количество репетиций. В выборку входят только пьесы с наградами. (Листинг 28, результат на рис. 17)

Листинг 27 — Запрос 4

```
SELECT p.Название AS НазваниеПьесы,
       s.Название AS НазваниеСпектакля,
       COUNT(DISTINCT nagr.Звание) AS КоличествоНаград,
       COUNT(DISTINCT rep.КодРепетиции) AS КоличествоРепетиций
  FROM Пьесы p
 JOIN Спектакли s ON p.КодПьесы = s.КодПьесы
 JOIN Роли r ON p.КодПьесы = r.КодПьесы
 JOIN Актеры a ON r.КодАктера = a.КодАктера
 LEFT JOIN
      Репетиции rep ON s.КодСпектакля = rep.КодСпектакля,
      UNNEST(a.ЗванияНаграды) AS nagr(Звание)
 GROUP BY p.Название, s.Название
 HAVING COUNT(nagr.Звание) > 0
 ORDER BY КоличествоРепетиций DESC;
```

НазваниеПьесы	НазваниеСпектакля	КоличествоНаград	КоличествоРепетиций
Чудовищный полк женщин	Античное сено	50	98
Плащ обезьяны	Возвращение к жизни	50	98
Давайте теперь похвалим известных людей	Путь ясной плоти	50	94
Прощай оружие	Благословение народа	50	90
Неизвестному Богу	Безголовый в Газе	50	89
Мир, плоть и дьявол	Мистер Стендфаст	50	89
Не погибнет	Нет шансов	50	89
Путь всякой плоти	Один в широком, широком море	50	89
Желтые луга Афродиды	В Земле Смерти	50	88
Что стало с Уорином	Область свидетелей	50	88
Бесконечная ночь	Лилии полевые	50	87
Человек внутри	Русалки поют	50	86
Танец Танец Танец	Парламент человечества	50	85
Приручение морского конька	Какой-то похороненный Цезарь	50	83
Блаженный дух	Золотая Чаща	50	83
За Мексиканским заливом	Великое дело времени	50	82
Ярмарка Цццзялэя	Нет шансов	50	81
К востоку от рая	Дни образцов	50	79
За Мексиканским заливом	Это Доминик Тус	50	79
Сердце лукаво более всего	К своим разбросанным телам, идите	50	79

Рисунок 17 — Результат выполнения 4 запроса

Используемые функции и концепции:

- **JOIN**: Объединение таблиц для получения данных.
- **LEFT JOIN**: Левое объединение для включения всех записей из левой таблицы.
- **UNNEST**: Работа с массивами в PostgreSQL.
- **GROUP BY**: Группировка данных.
- **HAVING**: Фильтрация групп.

- **COUNT**: Подсчет записей.
- **ORDER BY**: Сортировка результатов.

3.5 Запрос 5

Запрос выводит информацию о спектаклях, включая название спектакля, бюджет, количество актеров и количество сцен. Сортируется по количеству актеров. (Листинг 29, результат на рис. 18)

Так же добавим график зависимости количества сцен в спектакле и количества актеров от бюджета.(рис. 19)

Листинг 28 — Запрос 5

```
SELECT s.Название AS НазваниеСпектакля, s.Бюджет, COUNT(DISTINCT
z.КодАктера) AS КоличествоАктеров,
       (SELECT COUNT(DISTINCT сц.КодСцены)
        FROM Сцены сц
        JOIN СценыВРепетициях ср ON сц.КодСцены = ср.КодСцены
        JOIN Репетиции р ON ср.КодРепетиции = р.КодРепетиции
        WHERE р.КодСпектакля = s.КодСпектакля) AS КоличествоСцен
       FROM Спектакли s
       JOIN ЗанятостьВСпектаклях z ON s.КодСпектакля = z.КодСпектакля
       GROUP BY s.КодСпектакля, s.Бюджет, s.Название
       ORDER BY КоличествоАктеров DESC;
```

	НазваниеСпектакля	Бюджет	КоличествоАктеров	КоличествоСцен
1	После многих лет умирает лебедь	1403	443	49
2	Горсть пыли	2017	440	36
3	Плыть за закатом	2617	438	35
4	За заливом Мексики	6031	438	31
5	Рассмотрим Флебасу	8794	437	44
6	Оружие и человек	5700	434	34
7	Тень в ночи	6483	434	47
8	Умирание света	9909	433	41
9	Теннисные мячи звезд	7805	433	47
10	Под кровотечением	9213	433	44
11	Дульсе и декорум Эст	5231	433	41
12	Последний враг	2940	433	50
13	Страх и трепет	1232	433	48
14	Вот такие были радости	6460	432	28
15	Золотые яблочки Солнца	3449	432	38
16	Кровь - это вездеход	4220	431	47
17	Время нашей тьмы	5443	430	40
18	Парламент человечества	2039	430	56
19	Шутка Пилата	4622	430	42
20	Эго Доминус Тус	5195	429	35
21	В сухой сезон	8117	429	40
22	Куда ангелы боятся ступить	7086	429	47
23	Вниз к бессолнечному морю	5530	429	52

Рисунок 18 — Результат выполнения 5 запроса



Рисунок 19 — Зависимость количества сцен в спектакле и количества актеров от бюджета

Используемые функции и концепции:

- **JOIN**: Объединение таблиц для получения данных.
- **GROUP BY**: Группировка данных.
- **COUNT**: Подсчет записей.
- **ORDER BY**: Сортировка результатов.

3.6 Запрос 6

Запрос выводит информацию о занятости актеров в спектаклях и репетициях, включая ФИО актера, название спектакля, роль, роль, дата и время репетиции, сцена и продолжительность сцены. (Листинг 30, результат на рис. 20)

Листинг 29 — Запрос 6

```
SELECT а.ФИО AS Актер,
       s.Название AS Спектакль,
       г.НазваниеРоли AS Роль,
       г.Главная AS ГлавнаяРоль,
       реп.ДатаВремя AS ДатаВремяРепетиции,
       сц.Название AS Сцена,
       сц.Продолжительность AS ПродолжительностьСцены
  FROM Актеры а
 JOIN ЗанятостьВСпектаклях zvs ON а.КодАктера = zvs.КодАктера
 JOIN Спектакли s ON zvs.КодСпектакля = s.КодСпектакля
 LEFT JOIN Роли г ON а.КодАктера = г.КодАктера AND г.КодПьесы = s.КодПьесы
 LEFT JOIN Репетиции реп ON s.КодСпектакля = реп.КодСпектакля
 LEFT JOIN СценыВРепетициях свр ON реп.КодРепетиции = свр.КодРепетиции
 LEFT JOIN Сцены сц ON свр.КодСцены = сц.КодСцены
 WHERE г.КодРоли IS NOT NULL AND сц.КодСцены IS NOT NULL
 ORDER BY а.ФИО, s.Название, реп.ДатаВремя;
```

The screenshot shows the execution of Listing 29 in a database environment. The SQL code is displayed at the top, followed by the resulting table below.

Актер	Спектакль	Роль	ГлавнаяРоль	ДатаВремяРепетиции	Сцена	ПродолжительностьСцены
character varying (50)	character varying (50)	character varying (50)	boolean	timestamp without time zone	character varying (50)	interval
59159 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Ратибор Меркушева	false	2014-11-21 14:29:34	Сходить задорье.	00:05:00
59160 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Ратибор Меркушева	false	2014-11-21 14:29:34	Ночь нажать пропасть.	00:30:00
59161 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Агафон Маслова	true	2015-01-08 09:05:03	Бетонный мусор.	00:17:00
59162 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Бронислав Стрелков	true	2015-01-08 09:05:03	Бетонный мусор.	00:17:00
59163 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Яков Сергеев	false	2015-02-19 22:56:29	Расстегнуть писни светило.	00:11:00
59164 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Яков Сергеев	false	2015-02-19 22:56:29	Собеседник грудь скользить.	00:07:00
59165 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Андрей Дроздова	false	2015-05-16 02:50:47	зато прежний.	00:17:00
59166 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Андрей Дроздова	false	2015-05-16 02:50:47	зато прежний.	00:17:00
59167 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Мариян Субботина	true	2015-05-16 02:50:47	зато прежний.	00:17:00
59168 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Мариян Субботина	true	2015-05-16 02:50:47	зато прежний.	00:17:00
59169 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Бронислав Стрелков	true	2015-07-09 17:24:46	Зеленый.	00:21:00
59170 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Агафон Маслова	true	2015-07-09 17:24:46	Зеленый.	00:21:00
59171 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Ратибор Меркушева	false	2015-09-18 07:09:51	Житель направо.	00:09:00
59172 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Ратибор Меркушева	false	2015-09-18 07:09:51	Тюрама.	00:29:00
59173 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Алексей Карпова	true	2015-09-18 07:09:51	Житель направо.	00:09:00
59174 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Алексей Карпова	true	2015-09-18 07:09:51	Тюрама.	00:29:00
59175 Абрамова Лидия Эдуардовна	Ярмарка выдержала вете..	Бронислав Стрелков	true	2015-09-30 06:00:32	Полевой при.	00:08:00

Рисунок 20 — Результат выполнения 6 запроса

Используемые функции и концепции:

- **JOIN и LEFT JOIN**: Объединение таблиц для получения данных.
- **ORDER BY**: Сортировка результатов.

3.7 Триггер

Триггер обновляет стаж работы актера при вставке, обновлении или удалении записей в таблице «Контракт».(Листинг 31)

Листинг 30 — Триггер

```
CREATE OR REPLACE FUNCTION обновить_стаж_работы() RETURNS TRIGGER AS
$$
DECLARE
    начальная_дата DATE;
    конечная_дата DATE;
    стаж INTERVAL := INTERVAL '0 days';
BEGIN
    SELECT MIN(ДатаПриема) INTO начальная_дата
    FROM Контракт
    WHERE КодАктера = NEW.КодАктера;
    SELECT MAX(COALESCE(ДатаУвольнения, CURRENT_DATE))
    INTO конечная_дата
    FROM Контракт
    WHERE КодАктера = NEW.КодАктера;
    IF начальная_дата IS NOT NULL AND
        конечная_дата IS NOT NULL THEN
        стаж := age(конечная_дата, начальная_дата);
    END IF;
    UPDATE Актеры
    SET СтажРаботы = стаж
    WHERE КодАктера = NEW.КодАктера;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE TRIGGER обновить_стаж_работы_триггер
AFTER INSERT OR UPDATE OR DELETE ON Контракт
FOR EACH ROW
EXECUTE FUNCTION обновить_стаж_работы();
```

Используемые функции и концепции:

- **TRIGGER**: Автоматическое выполнение функций при определенных действиях.
- **PL/pgSQL**: Процедурный язык PostgreSQL для написания триггерных функций.
 - **AGE**: Вычисление разницы между датами.
 - **COALESCE**: Возвращение первого ненулевого значения.
 - **MIN и MAX**: Нахождение минимального и максимального значений.

ЗАКЛЮЧЕНИЕ

В процессе выполнения данной работы были проведены анализы и проектирование базы данных «Занятость актеров театра».

- Был проведен детальный анализ предметной области, а ходе которого были выявлены следующие сущности: Контракт, Актеры, Занятость в спектаклях, Роли, Спектакли, Пьесы, Репетиции, Занятость роли в сценах, Сцены, Сцены в репетициях.
- Определены основные связи между сущностями, построена инфологическая и датологическая модель базы данных, разработаны бизнес-процессы.
- Был написан код для создания базы данных и таблиц.
- Была разработана модель вставки псеводорандомных, осмысленных данных и написан код для их генерации.
- Было написано 6 сложных запросов и триггер для базы данных.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Моргунов, Е. П. PostgreSQL. Основы языка SQL: учеб. пособие / Е. П. Моргунов; под ред. Е. В. Рогова, П. В. Лузанова. — СПб.: БХВ-Петербург, 2018. — 336 с.: ил.
2. Новиков Б. А. Основы технологий баз данных: учебное пособие / Б. А. Новиков, Е. А. Горшкова, Н. Г. Графеева; под ред. Е. В. Рогова. — 2-е изд. — М.: ДМК Пресс, 2020. — 582 с.

ПРИЛОЖЕНИЯ

Приложение А Генерация фотографий

Листинг 31 — Генерируем 1000 фото людей

```
# -*- coding: utf-8 -*-
import requests
from time import time, sleep

def parse(i):
    """Function for parsing images from thispersondoesnotexist.com"""
    URL = "https://thispersondoesnotexist.com/"

    headers = {
        "Host": "thispersondoesnotexist.com",
        "User-Agent": r"Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:65.0) Gecko/20100101 Firefox/65.0",
        "Accept": r"text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
        "Accept-Language": r"en,en-US;q=0.5",
        "Accept-Encoding": "gzip, deflate, br",
        "DNT": "1",
        "Connection": "keep-alive",
        "Upgrade-Insecure-Requests": "1",
        "Cache-Control": r"max-age=0",
        "TE": "Trailers",
    }
    r = requests.get(URL, headers=headers, stream=True)

    jpg_file = "{}.jpeg".format(str(i))
    with open(jpg_file, "wb") as f:
        f.write(r.content)
        f.close()

if __name__ == "__main__":
    for i in range(1000):
        parse(i+1)
        sleep(0.05)
        print(i)
```

Приложение Б Генерация основного объема данных

Листинг 32 — Генерируем актеров

```
import psycopg2
from faker import Faker
import random

fake = Faker('ru_RU')

awards_list = [
    'Лучшая мужская роль', 'Лучшая женская роль', 'Лучшая роль второго плана',
    'Выдающееся исполнение', 'Премия за вклад в искусство', 'Лучшая новая звезда',
    'Премия критиков', 'Премия зрительских симпатий', 'Лучшая комедийная роль',
    'Лучшая драматическая роль', 'Лучший актерский ансамбль', 'Лучший актер озвучивания',
    'Лучший международный актер', 'Лучшая роль в главной роли', 'Лучшая роль второго плана',
    'Лучшее исполнение в короткометражке', 'Лучшее исполнение в полнометражном фильме',
    'Лучшее исполнение в спектакле', 'Лучшее исполнение в мюзикле',
    'Лучшее исполнение в телесериале', 'Лучшее исполнение в веб- сериале',
    'Лучшее исполнение в рекламе', 'Лучшее исполнение в музыкальном видео',
    'Лучшее исполнение в анимационном фильме', 'Лучшее исполнение в видеоигре',
    'Лучший детский актер', 'Лучший подростковый актер', 'Лучший актер старшего возраста',
    'Лучший мужской актер', 'Лучшая женская актерская роль', 'Лучший трансгендерный актер',
    'Актер года', 'Лучший прорыв', 'Лучшее гостевое исполнение', 'Лучшее камео',
    'Лучший злодей', 'Лучший герой', 'Лучший дуэт', 'Лучший актерский дебют',
    'Лучшее исполнение в исторической роли', 'Лучшее исполнение в научно-фантастической роли',
    'Лучшее исполнение в фэнтезийной роли', 'Лучшее исполнение в роли ужасов',
    'Лучшее исполнение в боевой роли', 'Лучшее исполнение в комедийной роли',
    'Лучшее исполнение в драматической роли', 'Лучшее исполнение в романтической роли',
    'Лучшее исполнение в триллере', 'Лучшее исполнение в военной роли',
    'Лучшее исполнение в вестерне', 'Лучшее исполнение в документальном фильме'
]

def insert_actors(cursor):
    for i in range(1000):
        name = fake.name()
        birth_date = fake.date_of_birth(minimum_age=18, maximum_age=65)
        f = open(f'/Volumes/pr/BDkurs/imgs/{i+1}.jpeg', 'rb')
        photo = f.read()
        f.close()
        experience = f'{random.randint(1, 50)} years'
        awards = random.sample(awards_list, random.randint(0, 10))
        cursor.execute(
            'INSERT INTO \\"Актеры\\" (\\"ФИО\\", \\"ДатаРождения\\", \\"Фото\\", \\"СтажРаботы\\", \\"ЗваниеНаграды\\")\n'
            'VALUES (%s, %s, %s, %s, %s)', (name, birth_date, photo, experience, awards)
        )
```

Листинг 33 — Генерируем пьесы, контракты, занятость, репетиции и сцены

```
def insert_plays(cursor):
    for i in range(2000):
        title = fake.sentence(nb_words=random.randint(2,3), variable_nb_words=True)
        budget = random.randint(1000, 10000)
        year = random.randint(1900, 2023)
        age_limit = f'{random.randint(0, 18)}'
        cursor.execute(
            'INSERT INTO \'Спектакли\' (\\"Название\\", \\"Бюджет\\", \\"ГодПостановки\\",
            \\"ОграничениеПоВозрасту\\", \\"КодПьесы\\") '
            'VALUES (%s, %s, %s, %s, %s)',
            (title, budget, year, age_limit, i + 1)
        )

def insert_contracts(cursor):
    for i in range(1000):
        start_date = fake.date_between(start_date='-20y', end_date='today')
        end_date = fake.date_between(start_date='today', end_date='+7y')
        salary = random.randint(30000, 100000)
        cursor.execute(
            'INSERT INTO \'Контракт\' (\\"ДатаПриема\\", \\"ДатаУвольнения\\", \\"Ставка\\",
            \\"КодАктера\\") '
            'VALUES (%s, %s, %s, %s)',
            (start_date, end_date, salary, i + 1)
        )

def insert_employment(cursor):
    for _ in range(1000000):
        actor_id = random.randint(1, 1000)
        play_id = random.randint(1, 2000)
        cursor.execute(
            'INSERT INTO \'ЗанятостьВСпектаклях\' (\\"КодСпектакля\\", \\"КодАктера\\") '
            'VALUES (%s, %s)',
            (play_id, actor_id)
        )

def insert_rehearsals(cursor):
    for _ in range(80000):
        date_time = fake.date_time_between(start_date='-20y', end_date='now')
        duration = f'{random.randint(1,4)}:{random.randint(0, 59)}:00'
        read_through = fake.boolean()
        play_id = random.randint(1, 2000)
        cursor.execute(
            'INSERT INTO \'Репетиции\' (\\"ДатаВремя\\", \\"Продолжительность\\", \\"Читка\\",
            \\"КодСпектакля\\") '
            'VALUES (%s, %s, %s, %s)',
            (date_time, duration, read_through, play_id)
        )

def insert_scenes(cursor):
    for i in range(40000):
        title = f'{fake.sentence(nb_words=random.randint(2,3), variable_nb_words=True)}'
        duration = f'00:{random.randint(1, 30)}:00'
        play_id = random.randint(1, 2000)
        cursor.execute(
            'INSERT INTO \'Сцены\' (\\"Название\\", \\"Продолжительность\\", \\"КодПьесы\\") '
            'VALUES (%s, %s, %s)',
            (title, duration, play_id)
        )
```

Листинг 34 — Генерируем роли, роли в сценах, сцены в репетициях, пьесы

```
def insert_roles(cursor):
    for i in range(1000000):
        title = fake.first_name() + ' ' + fake.last_name()
        main_role = fake.boolean()
        play_id = random.randint(1, 2000)
        actor_id = random.randint(1, 1000)
        cursor.execute(
            'INSERT INTO \'Роли\' (\\"НазваниеРоли\\", \\"Главная\\", \\"КодПьесы\\",
            \"КодАктера\") '
            'VALUES (%s, %s, %s, %s)',
            (title, main_role, play_id, actor_id)
        )

def insert_role_engagement(cursor):
    for _ in range(1000000):
        role_id = random.randint(1, 1000000)
        scene_id = random.randint(1, 40000)
        cursor.execute(
            'INSERT INTO \'ЗанятостьРолиВСценах\' (\\"КодРоли\\", \\"КодСцены\\") '
            'VALUES (%s, %s)',
            (role_id, scene_id)
        )

def insert_scenes_in_rehearsals(cursor):
    for _ in range(80000):
        rehearsal_id = random.randint(1, 80000)
        scene_id = random.randint(1, 40000)
        cursor.execute(
            'INSERT INTO \'СценыВРепетициях\' (\\"КодРепетиции\\", \\"КодСцены\\") '
            'VALUES (%s, %s)',
            (rehearsal_id, scene_id)
        )

def insert_plays_details(cursor):
    for i in range(2000):
        title = f'{fake.sentence(nb_words=random.randint(2,3),
variable_nb_words=True)}'
        year = random.randint(1900, 2023)
        author = fake.name()
        cursor.execute(
            'INSERT INTO \'Пьесы\' (\\"Название\\", \\"ГодВыпуска\\", \\"Автор\\") '
            'VALUES (%s, %s, %s)',
            (title, year, author)
        )
```

Листинг 35 — Вызываем все функции и записываем в базу

```
def main():
    try:
        conn = psycopg2.connect(
            dbname='Занятость актеров театра',
            user='pluttan',
            password=input('Password: '),
            host='localhost',
            port='5432'
        )
        print('Connected to the database')
        cursor = conn.cursor()
        print('generation actors')
        insert_actors(cursor)
        conn.commit()
        print('generation plays')
        insert_plays(cursor)
        conn.commit()
        print('generation contracts')
        insert_contracts(cursor)
        conn.commit()
        print('generation employment')
        insert_employment(cursor)
        conn.commit()
        print('generation rehearsals')
        insert_rehearsals(cursor)
        conn.commit()
        print('generation scenes')
        insert_scenes(cursor)
        conn.commit()
        print('generation roles')
        insert_roles(cursor)
        conn.commit()
        print('generation role engagement')
        insert_role_engagement(cursor)
        conn.commit()
        print('generation scenes in rehearsals')
        insert_scenes_in_rehearsals(cursor)
        conn.commit()
        print('generation plays details')
        insert_plays_details(cursor)
        conn.commit()
        print('Data inserted successfully')
    except Exception as e:
        print(f'Error: {e}')
    finally:
        if conn:
            conn.close()

if __name__ == '__main__':
    main()
```

Приложение В Генерация названий пьес и спектаклей

Листинг 36 — Перегенерируем названия пьес и спектаклей, заменяя их на осмысленные

```
require 'active_record'
require 'faker'
require 'google_translate_scraper'

# Настройка соединения с базой данных
ActiveRecord::Base.establish_connection(
  adapter: 'postgresql',
  host: 'localhost',
  username: 'pluttan',
  password: 'Pluttan2004',
  database: 'Занятость актеров театра'
)

# Определение моделей
class Spectacle < ActiveRecord::Base
  self.table_name = 'Спектакли'
end

class Play < ActiveRecord::Base
  self.table_name = 'Пьесы'
end

# Функция перевода
def translate_text(text, target_lang = 'ru')
  t = GoogleTranslateScraper.translate(
    :source_language => 'en',
    :target_language => 'ru',
    :search_text => text)
  puts t.translations.first
  t.translations.first
end

# Обновление названий спектаклей
updated_spectacles = 0
Spectacle.find_each do |spectacle|
  new_title = translate_text(Faker::Book.title)
  spectacle.update(Название: new_title)
  updated_spectacles += 1
end

puts "Названия спектаклей обновлены: #{updated_spectacles} записей."

# Обновление названий пьес
updated_plays = 0
Play.find_each do |play|
  new_title = translate_text(Faker::Book.title)
  play.update(Название: new_title)
  updated_plays += 1
end

puts "Названия пьес обновлены: #{updated_plays} записей."
puts "Обновление завершено!"
```

Приложение Г Перевод названий пьес и спектаклей

Листинг 37 — Переводим все названия на русский язык

```
import psycopg2
from deep_translator import GoogleTranslator
from faker import Faker
from random import randint

def translate_and_update_table(db_config,
                               table_name, column_name, id_column,
                               record_id, src_lang='en', dest_lang='ru'):
    cursor.execute(f"SELECT {column_name} FROM {table_name} WHERE {id_column} = %s",
    (record_id,))
    result = cursor.fetchone()
    if result:
        a = Faker('ru_RU')
        original_text = result[0]
        translator = GoogleTranslator(source=src_lang, target=dest_lang)
        translated_text = translator.translate(original_text)
        cursor.execute(
            f"UPDATE {table_name} SET {column_name} = %s WHERE {id_column} = %s",
            (translated_text, record_id))
        print("Перевод выполнен и значение обновлено в таблице.", id_column, record_id)
    else:
        print("Запись не найдена.")

db_config = {
    'dbname': 'Занятость актеров театра',
    'user': 'pluttan',
    'password': 'Pluttan',
    'host': 'localhost',
    'port': '5432'
}
conn = psycopg2.connect(**db_config)
cursor = conn.cursor()
for i in range(1, 2001):
    translate_and_update_table(db_config,
                               'Пьесы', 'Название', 'КодПьесы',
                               i, src_lang='en', dest_lang='ru')
    translate_and_update_table(db_config,
                               'Спектакли', 'Название', 'КодСпектакля',
                               i, src_lang='en', dest_lang='ru')
    if i%1000 == 0: conn.commit()
cursor.close()
conn.close()
```