|  |  |
| --- | --- |
| **Российский университет транспорта (МИИТ)**  **Институт транспортной техники и систем управления**  **Кафедра «Управление и защита информации»** | |
| **Отчет**  **по курсовому проекту**  **по теме «Разработка БД для отдела кадров»**  **по дисциплине «Основы построения защищенных баз данных»** | |
|  | Выполнил:  Студент группы ТКИ-442  Волочинский И. О.  Проверил:  Доцент кафедры УиЗи, к.т.н.  Васильева М. А. |
| Москва 2024 | |

Оглавление

[ЦЕЛЬ КУРСОВОГО ПРОЕКТА 3](#_Toc167878977)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc167878978)

[2 АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ 4](#_Toc167878979)

[2.1 Функциональные возможности 5](#_Toc167878980)

[2.2 Готовые запросы 5](#_Toc167878981)

[3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БД 5](#_Toc167878982)

[3.1 Преобразование ER-диаграммы в схему базы данных 5](#_Toc167878983)

[3.2 Составление реляционных отношений 7](#_Toc167878984)

[3.3 Определение дополнительных ограничений 8](#_Toc167878985)

[3.4 Нормализация полученных отношений 10](#_Toc167878986)

[4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БД 11](#_Toc167878987)

[4.1 Разработка скриптов для создания базы данных и таблиц 11](#_Toc167878988)

[4.1.1 Ограничения для таблицы «Заказы» 12](#_Toc167878989)

[4.1.2 Ограничения для таблицы «Клиенты» 13](#_Toc167878990)

[4.1.3 Ограничения для таблицы «Транспорт» 14](#_Toc167878991)

[4.2 Диаграмма базы данных 15](#_Toc167878992)

[4.3 Разработка скриптов для добавления данных в таблицы 16](#_Toc167878993)

[4.4 Проверка ограничений 17](#_Toc167878994)

[4.5 Разработка необходимых представлений (view) 20](#_Toc167878995)

[4.6 Разработка необходимых функций и процедур 21](#_Toc167878996)

[4.7 Разработка необходимых триггеров 22](#_Toc167878997)

[ЗАКЛЮЧЕНИЕ 22](#_Toc167878998)

# ЦЕЛЬ КУРСОВОГО ПРОЕКТА

Целью курсового проекта является изучение методов и закрепление знаний в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) PostgreSQL.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данном курсовом проекте для проектирования реляционной базы данных было выбрано издательство.

База данных создаётся для информационного обслуживания сотрудников издательства ВУЗа. Издательство печатает книги, журналы, пособия и т. д. студентов и преподавателей данного ВУЗа. Каждый автор может написать работу в соавторстве.

Данная системы должна выполнять следующие функции:

1. Выдавать список трудов данного автора(учитывать труды, выполненные в соавторстве).
2. Выдавать список трудов по выбранному разделу(книги, журналы, методические указания, пособия и т. д.).
3. Выдавать информацию о данном авторе.
4. Показывать количество напечатанных работ по кафедре, институту или университету в целом за отчётный период(квартал, год).

Выделим базовые сущности предметной области:

1. Сущность «Книги автора» (Общая, связывающая все нижеперчисленные) содержит в себе идентификатор, идентификатор автора, идентификатор публикации, роль автора в работе;
2. Сущность «Роль» содержит в себе идентификатор отдела и название роли;
3. Сущность «Книга» содержит в себе идентификатор книги, название и дату публикации;
4. Сущность «Автор» содержит в себе идентификатор автора, имя, фамилия и краткая биография

ER-диаграмма БД, выполненная в графическом редакторе *DB Designer*, приведена далее (Рисунок 1). Данная БД содержит связи «один-ко-многим» и «многие-к-одному».

|  |
| --- |
|  |
| 1. – уточненная ER-диаграмма |

# 2 АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ

Система создается для информационного обслуживания сотрудников компании.

Далее определяются границы информационной поддержки пользователей. Отдел кадров хранит сведения об отделах, должностях, сотрудниках и их детях, а также занимается управлением персоналом.

## 2.1 Функциональные возможности

Система в компании по продаже недвижимости имеет функциональную возможность ведения БД (запись, чтение, модификация и удаление данных), обеспечения логической непротиворечивости БД, а также обеспечения защиты данных от несанкционированного или случайного доступа – это значит, что в базе данных определены права на доступ к информации.

## 2.2 Готовые запросы

Наша система будет иметь следующие готовые запросы:

1. Выдавать список трудов данного автора(учитывать труды, выполненные в соавторстве).
2. Выдавать список трудов по выбранному разделу(книги, журналы, методические указания, пособия и т. д.).
3. Выдавать информацию о данном авторе.
4. Показывать количество напечатанных работ по кафедре, институту или университету в целом за отчётный период(квартал, год).

# 3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БД

## 3.1 Преобразование ER-диаграммы в схему базы данных

База данных создаётся на основании схемы базы данных. Преобразование ER-диаграммы в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения   
(таблицы БД).

Полученная схема реляционной базы данных (далее, РБД) компании по продаже недвижимости приведена ниже (Рисунок 2).

|  |
| --- |
|  |
| 1. – Схема РБД, полученная из ER-диаграммы компании |

В вашем примере создания таблиц базы данных можно определить следующие типы отношений:

1. **Отношение "один ко многим" между таблицами "author\_book" и "authors"**: В таблице "author\_book" есть поле "author\_id", которое является внешним ключом, ссылающимся на таблицу "author". Это означает, что один автор может иметь несколько публикаций, но каждая публикация связана только с одним автором.
2. **Отношение "Многие к одному" между таблицами "Role" и "author\_book"**: В таблице "author\_book" есть поле "role\_id", которое является внешним ключом, ссылающимся на таблицу "role". Это означает, что одна «роль» может быть у нескольких авторов, но каждый автор занимает только одну роль.

## 3.2 Составление реляционных отношений

Одно реляционное отношение соответствует одной сущности (объекту предметной области) и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи. В том случае, если базовое отношение не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несёт смысловой нагрузки и служит только для идентификации записей.

В данной системе сущности «Автор», «Книги» и «Роль» имеют атрибуты «author\_id», «book\_id», «role\_id» соответственно с целью компактного сбора в общей сущности «Книги автора» - то есть можно назвать такие атрибуты суррогатными первичными ключами.

Отношения приведены ниже (см. Таблица 1). Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной.

Таблица 1 – Сущности Базы Данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Книги автора | Идентификатор публикации автора | id | Целое число | Первичный ключ |
| Идентификатор автора | author\_id | Целое число | Обязательное поле,  Внешний ключ к отношению (сущности) Автор |
| Идентификатор книги | book\_id | Целое число | Обязательное поле,  Внешний ключ к отношению(сущности)  Книга |
| Идентификатор роли | role\_id | Целое число | Обязательное поле, Внешний ключ к отношению (сущности) Роль |
| Отдел | dep\_id | Целое число | Обязательное поле, Внешний ключ к отношению (сущности) Отдел |
| Автор | Идентификатор автора | Id | Целое число | Первичный ключ |
| Имя | first\_name | Строка (255) | Обязательное поле |
| Фамилия | second\_name | Стркоа(255) | Обязательное поле |
| Краткая биография | bio | Текст |  |
| Книга | Идентификатор книги | Id | Целое число | Первичный ключ |
| Название | title | Строка (255) | Обязательное, уникальное поле |
| Дата публикации | publising\_date | Дата | Обязательное |
| Роль | Идентификатор роли | id | Целое число | Первичный ключ |
| Название | role\_name | Строка (255) | Обязательное, уникальное поле |

## 3.3 Определение дополнительных ограничений

Необходимо отметить ограничения кроме тех, которые указаны в Таблице 1. Вставка данных в таблицу «Публикации автора» (а именно заполнение атрибутов «role\_id», «author\_id» и «book\_id») возможна только в том случае, если ранее поля role\_id, author\_id и book\_id были заполнены.

Так же, для обеспечения целостности данных и правильности отношений в базе данных, можно добавить следующие ограничения:

1. **Первичные ключи**: Каждая таблица должна иметь первичный ключ, который гарантирует уникальность записей и не допускает NULL значений.
2. **Ограничение на тип данных**: Поля, которые хранят числовые данные, должны иметь соответствующие ограничения на тип данных.
3. **Ограничения ссылочной целостности**: эти ограничения гарантируют, что значения внешних ключей всегда ссылается на существующие значения в связанных таблицах. Н
4. **Ограничение на уникальность**: ограничения, гарантируют, что каждая запись в таблице имеет уникальное значение для определенного поля или комбинации полей.
5. **Ограничения целостности проверки**: они позволяют определить допустимые диапазоны значений для определенных полей. Например, для таблицы "Position" можно добавить ограничение, что зарплата не может быть отрицательной.
6. **Ограничения на обновление и удаление**: можно установить правила, которые определяют, что происходит при обновлении или удалении записи, связанной с другими записями. Например, для таблицы "Employee" можно установить правило каскадного удаления, чтобы при удалении сотрудника все связанные с ним дети также удалялись.
7. **Ограничения на нулевые значения**: гарантируют, что определенные поля не могут содержать нулевые значения.

## 3.4 Нормализация полученных отношений

Нормализация отношений в БД является одним из ключевых аспектов проектирования и играет важную роль в обеспечении целостности данных, эффективности и удобства их использования.

Из представленных атрибутов (Таблица 1) единственное поле, которое можно было бы разбить на более простые составляющие – это адрес. В реальной задаче данные выносятся в отдельные отношения и понимается необходимость в этом, но в данной БД момент нормализации отношений пропускается с целью облегчения проектирования базы данных.

Для применения нормализации к нашей базе данных, состоящей из таблиц "Author\_book", "Author", "Book" и "Role", мы проверим каждую таблицу на соответствие требованиям первой (1NF), второй (2NF) и третьей (3NF) нормальных форм.

**1. Первая нормальная форма (1NF)**

Таблицы, представленные выше, уже находятся в первой нормальной форме, так как:

* Все атрибуты атомарны.
* Каждая строка уникальна.
* Таблицы не содержат повторяющихся групп.

**2. Вторая нормальная форма (2NF)**

Все три таблицы также находятся во второй нормальной форме, так как:

* Находятся в первой нормальной форме.
* Все неключевые атрибуты зависят от всего первичного ключа.

**3. Третья нормальная форма (3NF)**

Все три таблицы находятся в третьей нормальной форме, так как:

* Находятся во второй нормальной форме.
* Все неключевые атрибуты зависят только от первичного ключа и не имеют транзитивных зависимостей.

Таким образом, все таблицы в вашей базе данных соответствуют требованиям третьей нормальной формы. Это означает, что данные в базе данных нормализованы, что уменьшает избыточность и помогает избежать аномалий при обновлении, вставке или удалении данных.

# 4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БД

## 4.1 Разработка скриптов для создания базы данных и таблиц

Первоначально создаётся база данных «Издательство»:

CREATE DATABASE "Publisher"

Далее, были созданы таблицы «Книги», «Автор», «Роль» и «Публикации автора»:

CREATE TABLE "books" (

id INTEGER,

title VARCHAR(255),

publishing\_date DATE,

author VARCHAR(255)

);

CREATE TABLE "authors" (

id integer,

first\_name varchar(255),

second\_name varchar(255),

bio text

);

CREATE TABLE "role" (

id integer,

role\_name varchar(255)

)

CREATE TABLE "author\_books" (

id integer,

author\_id integer,

book\_id integer,

role\_id integer

)

Добавление в созданные таблицы первичных и внешних ключей. Первичные ключи:

alter table "books" add constraint "PK\_books" primary key ("id");

alter table "authors" add constraint "PK\_authors" primary key ("id");

alter table "author\_books" add constraint "PK\_ab" primary key ("id");

alter table "role" add constraint "PK\_role" primary key ("id");

Внешние ключи:

На следующем этапе вносятся ограничения для таблиц.

### 4.1.1 Ограничения для таблиц

В таблицах добавлено ограничение на обязательное заполнение следующих полей:

alter table "books" alter column "title" set not null;

alter table "books" alter column "author" set not null;

alter table "books" alter column "publishing\_date" set not null;

alter table "authors" alter column "first\_name" set not null;

alter table "authors" alter column "second\_name" set not null;

alter table "role" alter column "role\_name" set not null;

Ограничение на ввод уникальных значений:

alter table "Child"

add constraint "U\_Child\_id"

unique ("child\_id");

alter table "Child"

add constraint "U\_Child\_name"

unique ("name");

alter table "Department"

add constraint "U\_Department\_id"

unique ("dep\_id");

alter table "Department"

add constraint "U\_Department\_name"

unique ("name");

alter table "Employee"

add constraint "U\_Employee\_id"

unique ("emp\_id");

alter table "Position"

add constraint "U\_Position\_id"

unique ("pos\_id");

alter table "Position"

add constraint "U\_Position\_name"

unique ("name");

Ограничение, которое устанавливает значения по умолчанию:

alter table "books" alter column "id" set default 0;

alter table "authors" alter column "id" set default 0;

alter table "author\_books" alter column "id" set default 0;

## alter table "role" alter column "id" set default 0;

## 4.2 Диаграмма базы данных

Результатом разработки скриптов является диаграмма базы данных, представленная ниже (Рисунок 3).

|  |
| --- |
|  |
| 1. – Диаграмма базы данных |

## 4.3 Разработка скриптов для добавления данных в таблицы

Заполнение таблицы «Роли»:

insert into "role" values (1, 'Автор'), (2, 'Соавтор');

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Роли» |

Заполнение таблицы «Авторы»:

insert into "authors" values (1, 'Джордж', 'Оруэлл', ''), (2, 'Евгений', 'Замятин', '');

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Авторы» |

Заполнение таблицы «Книги»:

insert into "books" values (1, 'Мы', '1923-01-12'), (2, '1984', '1984-08-19');

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Orders» |

Заполнение таблицы «Публикации автора»:

insert into "author\_books" values (1, 1, 2, 1), (2, 2, 1, 1);

|  |
| --- |
|  |
| 1. – Внесение данных в таблицу «Публикации автора» |

## 4.5 Разработка необходимых представлений (view)

К необходимому представлению, относится вывод списка заказов с информацией о клиенте и транспорте – рисунок 9, которое реализуется следующим образом:

CREATE VIEW author\_books\_expanded AS

SELECT

ab.id,

ab.author\_id,

a.first\_name AS author\_first\_name,

a.second\_name AS author\_second\_name,

ab.book\_id,

b.title AS book\_title,

ab.role\_id,

r.role\_name AS role\_name

FROM

author\_books ab

JOIN authors a ON ab.author\_id = a.id

JOIN books b ON ab.book\_id = b.id

JOIN role r ON ab.role\_id = r.id;

|  |
| --- |
|  |
| 1. – Представление и результат |

## 4.7 Разработка необходимых триггеров

В данной работе был создан единственный необходимый триггер - триггер на запрет удаления таблицы из базы данных. Его создание представлено далее::

# CREATE OR REPLACE FUNCTION prevent\_table\_deletion()

# RETURNS EVENT\_TRIGGER AS $$

# BEGIN

# RAISE EXCEPTION 'Удаление таблицы запрещено!';

# END;

# $$ LANGUAGE plpgsql;

# CREATE EVENT TRIGGER prevent\_table\_deletion\_trigger

# ON sql\_drop

# EXECUTE FUNCTION prevent\_table\_deletion();

# ЗАКЛЮЧЕНИЕ

Проект позволил закрепить знания по проектированию реляционных баз данных в системе управления базами данных PostgreSQL. База данных для компании по продаже недвижимости позволит эффективно управлять информацией об объектах недвижимости, риелторах и клиентах, что способствует повышению эффективности работы компании и улучшению обслуживания клиентов.