|  |  |
| --- | --- |
| **Российский университет транспорта (МИИТ)**  **Институт транспортной техники и систем управления**  **Кафедра «Управление и защита информации»** | |
| **Отчет**  **по курсовому проекту**  **по теме «Разработка БД для архива»**  **по дисциплине «Основы построения защищенных баз данных»** | |
|  | Выполнил:  Студент группы ТКИ-441  Мацко А.А.  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 был выбран архив.

БД создается для информационного обслуживания сотрудников архива уголовного розыска. В архиве лежат Дела за разные годы.

Предметная область данной системы включает в себя следующие особенности:

1. Выдавать список Дел по ФИО осужденного;
2. Выдавать список Дел, которые вед данный следователь;
3. Находить Дела по содержанию;
4. Выдавать список Дел по данной статье преступления (по характеру преступления).

Выделим базовые сущности предметной области:

1. Сущность «Дело» содержит в себе идентификатор клиента, также номер дела и содержание;
2. Сущность «Осужденный» содержит в себе идентификатор преступника, имя и фамилию;
3. Сущность «Статья преступления» содержит в себе идентификатор статьи, номер статьи и содержание статьи;
4. Сущность «Следователь» содержит в себе идентификатор следователя, имя и фамилию;

ER-диаграмма БД, выполненная в графическом редакторе drawio, приведена далее (Рисунок 1). Данная БД содержит связи «многие-ко-многим».

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

# 2 АНАЛИЗ ИНФОРМАЦИОННЫХ ЗАДАЧ И КРУГА ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ

Система создается для информационного обслуживания сотрудников архива, где требуется выдавать список Дел по ФИО осужденного, по содержанию, по данной статье преступления (по характеру преступления) и по следователю, который вел данное дело.

## 2.1 Готовые запросы

Наша система будет иметь следующие готовые запросы:

1. Выдавать список Дел по ФИО осужденного;
2. Выдавать список Дел, которые вел данный следователь;
3. Находить Дела по содержанию;
4. Выдавать список Дел по данной статье преступления (по характеру преступления).

# 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) архива 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 "многие ко многим" между таблицами "Case" и "Convict"**: это означает, что одно дело может включать нескольких осужденных и один осужденный может быть связан с несколькими делами. Таблица “ConvictCase” помогает отслеживать, какие осужденные связаны с какими делами.
2. **Отношение "многие ко многим" между таблицами "Case" и "CrimeArticle"**: это означает, что один следователь может вести несколько дел и одно дело может быть связано с несколькими следователями. Таблица “InvestigatorCase” помогает отслеживать, какие следователи ведут какие дела.
3. **Отношение "многие ко многим" между таблицами "Case" и "Investigator"**: это означает, что дно дело может включать несколько статей преступления и одна статья преступления может относиться к нескольким делам. Таблица “CrimeArticleCase” помогает отслеживать, какие статьи преступления относятся к каким делам.

Таким образом, в вашей базе данных реализованы отношения "многие ко многим" между делами и преступниками, делами и криминальными статьями, а также между делами и следователями.

## 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.

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

Таблица 1 –Дело

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Дело  (Case) | Идентификатор дела | CaseID | Целое число | Первичный ключ |
| Номер дела | CaseNumber | Целое число | Обязательное  поле |
| Описание дела | Description | Строка (255) | Обязательное поле |

Таблица 2 –Преступник

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Преступник  (Convict) | Идентификатор преступника | ConvictID | Целое число | Первичный ключ |
| Имя | FirstName | Строка  (255) | Обязательное  поле |
| Фамилия | LastName | Строка  (255) | Обязательное поле |

Таблица 3 –Статья

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Статья  (CrimeArticle) | Идентификатор статьи | ArticleID | Целое число | Первичный ключ |
| Номер статьи | ArticleNumber | Целое число | Обязательное, уникальное поле |
| Описание статьи | Description | Строка  (255) | Обязательное поле |

Таблица 4 –Следователь

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Следователь  (Investigator) | Идентификатор следователя | InvestigatorID | Целое число | Первичный ключ |
| Имя | FirstName | Строка  (255) | Обязательное поле |
| Фамилия | LastName | Строка | Обязательное поле |

Таблица 5 –Преступник-Дело

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Преступника‑Дело  (ConvictCase) | Идентификатор таблицы | ConvictCaseID | Целое число | Первичный ключ |
| Идентификатор осужденного | ConvictID | Целое число | Внешний ключ |
| Идентификатор дела | CaseID | Целое число | Внешний ключ |

Таблица 6 –Следователь-Дело

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Следователь‑Дело  (InvestigatorCase) | Идентификатор таблицы | InvestigatortCaseID | Целое число | Первичный ключ |
| Идентификатор следователя | InvestigatorID | Целое число | Внешний ключ |
| Идентификатор дела | CaseID | Целое число | Внешний ключ |

Таблица 7 –Статья-Дело

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование сущности** | **Содержание атрибута** | **Имя атрибута** | **Тип** | **Примечание** |
| Статья‑Дело  (CrimeArticleCase) | Идентификатор таблицы | CrimeArticleCaseID | Целое число | Первичный ключ |
| Идентификатор следователя | CrimeArticleCaseID | Целое число | Внешний ключ |
| Идентификатор дела | CaseID | Целое число | Внешний ключ |

## 3.3 Определение дополнительных ограничений

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.
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.
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 "Reservations" можно установить правило каскадного удаления, чтобы при удалении клиента или вида транспорта все связанные заказы 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.

Для применения нормализации к нашей базе данных, состоящей из таблиц "Case", "Convict" и "CrimeArticle", "Investigator", "ConvictCase", “InvestigatorCase” и “CrimeArticleCase” 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 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 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 «Архив»:

create database "Archive"

Далее, были созданы таблицы «Дело», «Статья» ,«Преступник», «Следователь», «ДелоСтатья», «ДелоПреступник», «ДелоСледователь»:

**create** **table** **"Case"**

(

**"CaseID"** serial **primary** **key**,

**"CaseNumber"** **int** **not** **null**,

**"Description"** **varchar**(255) **not** **null**

)

**create** **table** **"Convict"**

(

**"ConvictID"** serial **primary** **key**,

**"FirstName"** **varchar**(25) **not** **null**,

**"LastName"** **varchar**(25) **not** **null**

)

**create** **table** **"CrimeArticle"**

(

**"ArticleID"** serial **primary** **key**,

**"ArticleNumber"** **int** **not** **null**,

**"Description"** **varchar**(255) **not** **null**

)

**create** **table** **"Investigator"**

(

**"InvestigatorID"** serial **primary** **key**,

**"FirstName"** **varchar**(25) **not** **null**,

**"LastName"** **varchar**(25) **not** **null**

)

**create** **table** **"ConvictCase"**

(

**"ConvictCaseID"** serial **primary** **key**,

**"ConvictID"** serial,

**"CaseID"** serial,

**foreign** **key** (**"ConvictID"**) **references** **"Convict"**(**"ConvictID"**),

**foreign** **key** (**"CaseID"**) **references** **"Case"**(**"CaseID"**)

)

**create** **table** **"InvestigatorCase"**

(

**"InvestigatorCaseID"** serial **primary** **key**,

**"InvestigatorID"** serial,

**"CaseID"** serial,

**foreign** **key** (**"InvestigatorID"**) **references** **"Investigator"**(**"InvestigatorID"**),

**foreign** **key** (**"CaseID"**) **references** **"Case"**(**"CaseID"**)

)

**create** **table** **"CrimeArticleCase"**

(

**"CrimeArticleCase"** serial **primary** **key**,

**"ArticleID"** serial,

**"CaseID"** serial,

**foreign** **key** (**"ArticleID"**) **references** **"CrimeArticle"**(**"ArticleID"**),

**foreign** **key** (**"CaseID"**) **references** **"Case"**(**"CaseID"**)

)

На следующем этапе вносятся ограничения для таблиц.

### 4.1.1 Ограничения для таблицы «Дело»

**alter** **table** **"Case"**

**add** **constraint** UQ\_Case\_Number **unique** (**"CaseNumber"**)

**alter** **table** **"Case"**

**add** **constraint** UQ\_Case\_Description **unique** (**"Description"**)

### 4.1.2 Ограничения для таблицы «Статья»

**alter** **table** **"CrimeArticle"**

**add** **constraint** UQ\_Description **unique** (**"Description"**)

**alter** **table** **"CrimeArticle"**

**add** **constraint** UQ\_Article\_Number **unique** (**"ArticleNumber"**)

## 4.2 Диаграмма базы данных

Результатом разработки скриптов является диаграмма базы данных, представленная ниже (Рисунок 4).

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

## 4.3 Разработка скриптов для добавления данных в таблицы

Заполнение таблицы «Следователь»:

**insert** **into** "Investigator" ("FirstName", "LastName") **values** (**'Виктор'**, **'Любимов'**)

**insert** **into** **"Investigator"** (**"FirstName"**, **"LastName"**) **values** (**'Евгений'**, **'Скрябин'**)

**insert** **into** **"Investigator"** (**"FirstName"**, **"LastName"**) **values** (**'Александер'**, **'Седых'**)

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

Заполнение таблицы «Case»:

**insert** **into** "Case" ("CaseNumber", "Description") **values** (1, **'Убийство и хищение'**)

**insert** **into** **"Case"** (**"CaseNumber"**, **"Description"**) **values** (45, **'Разбой'**)

**insert** **into** **"Case"** (**"CaseNumber"**, **"Description"**) **values** (14, **'Нанесение особотяжких'**)

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

Заполнение таблицы «Преступник»:

**insert** **into** "Convict" ("FirstName", "LastName") **values** (**'Иван'**, **'Иванов'**)

**insert** **into** **"Convict"** (**"FirstName"**, **"LastName"**) **values** (**'Олег'**, **'Рылов'**)

**insert** **into** **"Convict"** (**"FirstName"**, **"LastName"**) **values** (**'Павел'**, **'Селезнев'**)

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

Заполнение таблицы «Статья»:

**insert** **into** "CrimeArticle" ("ArticleNumber", "Description") **values** (288, **'Хранение наркотиков'**)

**insert** **into** **"CrimeArticle"** (**"ArticleNumber"**, **"Description"**) **values** (67, **'Убийство'**)

**insert** **into** **"CrimeArticle"** (**"ArticleNumber"**, **"Description"**) **values** (88, **'Воровство'**)

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

1. – Внесение данных в таблицу «Статья»

Заполнение таблицы «ПреступникДело»:

**insert** **into** "ConvictCase" ("ConvictID", "CaseID") **values** (2, 2)

**insert** **into** **"ConvictCase"** (**"ConvictID"**, **"CaseID"**) **values** (1, 1)

**insert** **into** **"ConvictCase"** (**"ConvictID"**, **"CaseID"**) **values** (3, 3)

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

1. – Внесение данных в таблицу «ПреступникДело»

Заполнение таблицы «СледовательДело»:

**insert** **into** "InvestigatorCase" ("InvestigatorID", "CaseID") **values** (1, 1)

**insert** **into** **"InvestigatorCase"** (**"InvestigatorID"**, **"CaseID"**) **values** (2, 2)

**insert** **into** **"InvestigatorCase"** (**"InvestigatorID"**, **"CaseID"**) **values** (3, 3)

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

Заполнение таблицы «СтатьяДело»:

**insert** **into** "CrimeArticleCase" ("ArticleID", "CaseID") **values** (1, 1)

**insert** **into** **"CrimeArticleCase"** (**"ArticleID"**, **"CaseID"**) **values** (2, 2)

**insert** **into** **"CrimeArticleCase"** (**"ArticleID"**, **"CaseID"**) **values** (3, 3)

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

## 4.5 Разработка необходимых представлений (view)

**create** **or** **replace** **view** ConvictDetailsView **as**

**select**

conv.**"FirstName"**,

conv.**"LastName"**,

ca.**"ArticleNumber"**,

ca.**"Description"** **as** **"ArticleDescription"**,

c.**"CaseNumber"**,

c.**"Description"** **as** **"CaseDescription"**

**from**

**"Convict"** conv

**join**

**"ConvictCase"** cc **on** conv.**"ConvictID"** = cc.**"ConvictID"**

**join**

**"Case"** c **on** cc.**"CaseID"** = c.**"CaseID"**

**join**

**"CrimeArticleCase"** cca **on** c.**"CaseID"** = cca.**"CaseID"**

**join**

**"CrimeArticle"** ca **on** cca.**"ArticleID"** = ca.**"ArticleID"**;

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

## 4.6 Разработка необходимых функций

**Функция вывода по Статье**:

**create** **or** **replace** **function** GetCasesByCrimeArticle(articleNumber **int**)

**returns** **table**(CaseID **int**, CaseNumber **int**, Description **varchar**) **as** **$$**

**begin**

**return** query

**select** c.**"CaseID"**, c.**"CaseNumber"**, c.**"Description"**

**from** **"CrimeArticle"** ca

**join** **"CrimeArticleCase"** cca **on** ca.**"ArticleID"** = cca.**"ArticleID"**

**join** **"Case"** c **on** cca.**"CaseID"** = c.**"CaseID"**

**where** ca.**"ArticleNumber"** = articleNumber;

**end**;

**$$** **language** plpgsql;

|  |
| --- |
|  |
| 1. – Выполнение функции вывода по Статье |

**Функция вывода по ФИО преступника**:

**create** **or** **replace** **function** GetCasesByConvictName(convictFirstName **varchar**, convictLastName **varchar**)

**returns** **table**(**"CaseID"** **int**, **"CaseNumber"** **int**, **"Description"** **varchar**) **as** **$$**

**begin**

**return** query

**select** c.**"CaseID"**, c.**"CaseNumber"**, c.**"Description"**

**from** **"Convict"** conv

**join** **"ConvictCase"** cc **on** conv.**"ConvictID"** = cc.**"ConvictID"**

**join** **"Case"** c **on** cc.**"CaseID"** = c.**"CaseID"**

**where** conv.**"FirstName"** = convictFirstName **and** conv.**"LastName"** = convictLastName;

**end**;

**$$** **language** plpgsql;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

1. – Выполнение функции для вывода по ФИО преступника

**Функция вывод по следователю**:

**create** **or** **replace** **function** GetCasesByInvestigatorName(investigatorFirstName **varchar**, investigatorLastName **varchar**)

**returns** **table**(**"CaseID"** **int**, **"CaseNumber"** **int**, **"Description"** **varchar**) **as** **$$**

**begin**

**return** query

**select** c.**"CaseID"**, c.**"CaseNumber"**, c.**"Description"**

**from** **"Investigator"** inv

**join** **"InvestigatorCase"** ic **on** inv.**"InvestigatorID"** = ic.**"InvestigatorID"**

**join** **"Case"** c **on** ic.**"CaseID"** = c.**"CaseID"**

**where** inv.**"FirstName"** = investigatorFirstName **and** inv.**"LastName"** = investigatorLastName;

**end**;

**$$** **language** plpgsql;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

1. – Выполнение функции вывод по следователю

**Функция вывод по Содержанию:**

**CREATE** **OR** **REPLACE** **FUNCTION** get\_cases\_by\_content(**content** **TEXT**)

**RETURNS** **TABLE**(**"CaseID"** **INT**, **"CaseNumber"** **INT**, **"Description"** **VARCHAR**)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT**

c.**"CaseID"**, c.**"CaseNumber"**, c.**"Description"**

**FROM**

**"Case"** c

**WHERE**

c.**"Description"** **ILIKE** **'%'** || **content** || **'%'**;

**END**;

**$$**;

|  |
| --- |
|  |
| 1. – Выполнение функции вывода по содержанию |

## 4.7 Разработка необходимых триггеров

**create** **or** **replace** **function** get\_all\_investigators()

**returns** **table**(**"InvestigatorID"** **int**, **"FirstName"** **varchar**, **"LastName"** **varchar**) **as** **$$**

**begin**

**return** query

**select** **"Investigator"**.**"InvestigatorID"**, **"Investigator"**.**"FirstName"**, **"Investigator"**.**"LastName"**

**from** **"Investigator"**;

**end**;

**$$** **language** plpgsql;

**create** **trigger** convict\_insertion\_trigger

**after** **insert** **on** **"Convict"**

**for** **each** **row**

**execute** **function** notify\_convict\_insertion();

# ЗАКЛЮЧЕНИЕ

Проект позволил закрепить знания по проектированию реляционных баз данных в системе управления базами данных PostgreSQL.