

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра ИС**

**КУРСОВАЯ РАБОТА**  
**по дисциплине «Управление данными»**  
**Тема: Проектирование базы данных**

Студентка гр. 1373

\_\_\_\_\_

Новикова А.С.

Преподаватель

\_\_\_\_\_

Татарникова Т.М.

Санкт-Петербург

2023

## ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студентка Новикова А.С.

Группа 1373

Тема работы: Проектирование базы данных

Исходные данные:

Спроектировать базу данных, построить программу, обеспечивающую взаимодействия с ней в режиме диалога, для работников технического архива предприятия.

Содержание пояснительной записки:

Требуемые разделы: «Введение», «Анализ предметной области», «Обоснование модели данных», «Обоснование выбора СУБД», «Описание функций групп пользователей», «Описание функций управления данными», «Организация защиты БД», «Заключение», «Руководство пользователя БД», «Листинг программного кода».

Предполагаемый объем пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания: 01.09.2023

Дата сдачи работы: 14.12.2023

Дата защиты работы: 14.12.2023

Студентка

\_\_\_\_\_

Новикова А.С.

Преподаватель

\_\_\_\_\_

Татарникова Т.М.

## **АННОТАЦИЯ**

В данной курсовой работе представлены концепции, используемые при разработке базы данных: анализ области, выбор технологий, создание базы данных. Результатом работы является созданная база данных для взаимодействия с архивом предприятия.

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	5
АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ .....	6
ОБОСНОВАНИЕ МОДЕЛИ ДАННЫХ .....	7
ОБОСНОВАНИЕ ВЫБОРА СУБД .....	7
ОПИСАНИЕ ФУНКЦИЙ ГРУПП ПОЛЬЗОВАТЕЛЕЙ .....	7
ОПИСАНИЕ ФУНКЦИЙ УПРАВЛЕНИЯ ДАННЫМИ .....	8
ОРГАНИЗАЦИЯ ЗАЩИТЫ БАЗЫ ДАННЫХ .....	9
ЗАКЛЮЧЕНИЕ .....	10
ПРИЛОЖЕНИЕ А .....	11
ПРИЛОЖЕНИЕ Б.....	18

## **ВВЕДЕНИЕ**

Для удобства хранения документов на предприятиях есть архивы. Но иногда найти нужный документ бывает довольно трудно, поэтому нужно создать базу данных, с помощью которой можно будет с легкостью получить всю необходимую информацию.

## **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

База данных предназначена для работников технического отдела предприятия. В базе данных должны храниться сведения о документах, находящихся в архиве, а также сведения о работниках предприятия.

Сведения о документах включают в себя: номер документа, название, тема, количество экземпляров и дату поступления в архив. Документ помещается на первое свободное место в архиве. Сведения о положении документа включают в себя номер стеллажа, номер полки и номер ячейки. Шкаф состоит из полок, а полки из ячеек. Для этой задачи были взяты стеллажи с двумя полками и полки с двумя ячейками. В одной ячейке может храниться либо один документ, либо ничего.

Задачи, которые может решать БД:

1. Определить название наиболее часто требуемого документа;
2. Определить общее количество документов на заданную тему;
3. Определить тему по названию документа;
4. Определить название документа, который имеется в наибольшем количестве экземпляров;
5. Определить отдел работника, который чаще всех обращается к архиву;
6. Определить ФИО сотрудника, обратившимся последним к указанному документу;
7. Добавить документ;
8. Удалить экземпляр документа;
9. Изменить номер телефона отдела;

Также БД может выдать справки об абонентах отдела, пользующихся архивом, и отчет о работе архива (число единиц хранения, названия документов, поступивших в архив за последний месяц, количество экземпляров каждого документа, место его хранения).

## ОБОСНОВАНИЕ МОДЕЛИ ДАННЫХ

Для выполнения этой задачи была выбрана реляционная модель данных, так как в ней все данные структурированы. Также для создания функций нужно строить итоговые запросы и производить поиск по нескольким характеристикам одновременно.

## ОБОСНОВАНИЕ ВЫБОРА СУБД

В качестве СУБД была выбрана PostgreSQL, так как она работает с реляционными моделями данных. PostgreSQL также содержит все необходимые функциональные возможности стандартной модели SQL. И, что немаловажно, она имеет хорошие характеристики по производительности и совместимости с различными платформами, у нее открытый исходный код и свободное распространение.

## ОПИСАНИЕ ФУНКЦИЙ ГРУПП ПОЛЬЗОВАТЕЛЕЙ

Группы пользователей: сотрудник предприятия и администратор архива. Сотрудники могут смотреть информацию по данным архива, а администратор может добавлять документы, удалять их экземпляры и изменять номер телефона отдела.

Назначения прав доступа:

Функция	Сотрудник	Администратор
Получить информацию по архиву	S	SUID
Прямое взаимодействие с архивом		SUID

- S — чтение данных;
- I — добавление данных;
- U — модификация данных;
- D — удаление данных.

## ОПИСАНИЕ ФУНКЦИЙ УПРАВЛЕНИЯ ДАННЫМИ

В БД есть 3 функции для управления данными:

### 1. Добавить документ

Функция принимает на вход название документа, название темы и количество экземпляров. В результате выполнения добавляется запись в таблицу «Документы» об указанном документе. Если документов на эту тему не было, то о ней также добавляется запись в соответствующую таблицу. Если все существующие в БД ячейки заняты, то создаётся новая ячейка, в которую помещается документ.

Пример использования:

```
select add_document('График отпусков', 'Отпуск', 5);
```

### 2. Удалить экземпляры документа

Функция принимает на вход название документа и количество экземпляров, которые нужно удалить. Если после удаления оставшееся число экземпляров меньше или равно 0, то запись о документе удаляется, а ячейка становится свободной.

Пример использования:

```
select delete_document_copy('График отпусков', 2);
```

### 3. Изменить номер телефона отдела

Функция принимает на вход название отдела и актуальный номер телефона, а затем изменяет номер телефона соответствующего отдела.

Пример использования:

```
select change_phone_number('Бухгалтерия', '+7907777777');|
```



## ОРГАНИЗАЦИЯ ЗАЩИТЫ БАЗЫ ДАННЫХ

Для каждого информационного объекта выбраны следующие ограничения целостности:

- Идентификаторы стеллажа, полки, ячейки, темы, документа, отдела, работника и запроса — первичные ключи.
- Название темы, документа, отдела, номер телефона отдела и имя сотрудника не могут быть NULL.
- Количество экземпляров документа должно быть положительным.
- Комбинация стеллажа, полки и ячейки должны быть уникальны для каждого отдельного документа.

Кроме этого, рекомендуется делать на другие носители резервные копирования: еженедельное полное копирование и ежедневное инкрементное копирование.

Для администратора архива также предусмотрена парольная идентификация. Для сотрудников она не предусмотрена.

## **ЗАКЛЮЧЕНИЕ**

В ходе работы были изучены принципы и технологии разработки баз данных. На основе полученных знаний была создана собственная база данных. Для этого была использована система управления БД PostgreSQL.

Созданная база данных может быть использована для удобного хранения документов и получения информации о них, а также прямой работы с ними.

## ПРИЛОЖЕНИЕ А

### РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ БАЗЫ ДАННЫХ

Для получения доступа к данным необходимо войти в базу в качестве сотрудника предприятия или администратора. Администратору необходимо применить пароль 'Jsp4j4JcKQXl' для успешного получения доступа.

#### add\_document

Чтобы добавить документ, необходимо воспользоваться функцией add\_document. Необходимо указать название документа, название темы документа и количество экземпляров документа в качестве входных аргументов функции. Документ будет помещен на первую свободную ячейку.

Пример использования:

```
select add_document('График отпусков', 'Отпуск', 5);
```

Перед использованием функции БД была пустая. В таблицах появились следующие записи:

- documents:

	<small>123</small> id	<small>ABC</small> name	<small>123</small> cell_id	<small>123</small> theme_id	<small>123</small> amount	<small>🕒</small> receipt_date
1	1	График отпусков	1 <small>🔗</small>	1 <small>🔗</small>	5	2023-12-13

- theme

	<small>123</small> id	<small>ABC</small> name
1	1	Отпуск

- cell

	<small>123</small> id	<small>123</small> shelf_id	<small>☑</small> is_empty
1	1	1 <small>🔗</small>	[ ]

- shelf

	<small>123</small> id	<small>123</small> rack_id
1	1	1 <small>🔗</small>

- rack

	<small>123</small> id
1	1

## delete\_document\_copy

Для удаления экземпляров документа нужно воспользоваться функцией `delete_document_copy`. В качестве аргументов на вход подаются название документа и количество экземпляров, которые нужно удалить.

Для демонстрации примеров возьмём уже заполненную БД. Так выглядит таблица `documents`:

	id	name	cell_id	theme_id	amount	receipt_date
1	1	График отпусков	1	1	5	2023-12-13
2	2	Штатное расписание	2	2	10	2023-12-05
3	3	Положение об оплате труда	3	3	1	2023-01-01
4	4	Выплаты за ноябрь	4	3	4	2023-12-13

Пример использования:

```
select delete_document_copy('График отпусков', 2);
```

После применения запись о документе «График отпусков» изменилась и стала выглядеть так:

	id	name	cell_id	theme_id	amount	receipt_date
	1	График отпусков	1	1	3	2023-12-13

Если введенное количество экземпляров будет равно количеству экземпляров в базе данных или будет превышать его, то запись о документе пропадёт, а ячейка, в которой хранился документ, станет пустой.

Пример использования:

```
select delete_document_copy('График отпусков', 5);
```

После применения данные в БД изменились и стали выглядеть так:

- `documents`

	id	name	cell_id	theme_id	amount	receipt_date
	2	Штатное расписание	2	2	10	2023-12-05
	3	Положение об оплате труда	3	3	1	2023-01-01
	4	Выплаты за ноябрь	4	3	4	2023-12-13

- `cell`

	id	shelf_id	is_empty
	1	1	[v]
	2	1	[ ]
	3	2	[ ]
	4	2	[ ]

## get\_document\_name\_with\_max\_requests

Чтобы определить название наиболее часто требуемого документа, нужно воспользоваться функцией `get_document_name_with_max_requests`.

Так сейчас выглядят заполненные таблицы:

- department

123 id	ABC name	ABC phone
1	Бухгалтерия	+79088888888
2	Отдел кадров	+79500000000

- employee

123 id	ABC name	123 department_id
1	Иванов И.И.	1 ↗
2	Петров П.П.	2 ↗
3	Андреев А.А.	1 ↗
4	Романов Р.Р.	2 ↗

- requests

123 id	request_date	123 document_id	123 employee_id
6	2023-12-06	2 ↗	1 ↗
7	2023-12-07	2 ↗	2 ↗
8	2023-12-08	2 ↗	3 ↗
9	2023-12-07	3 ↗	1 ↗
10	2023-12-09	3 ↗	4 ↗

Пример использования:

```
select * from get_document_name_with_max_requests();
```

Получаем следующий результат:

ABC documents
Штатное расписание

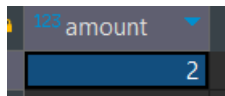
### **get\_document\_amount\_by\_theme**

Чтобы определить количество документов на определённую тему, необходимо воспользоваться функцией `get_document_amount_by_theme`. В качестве аргументов на вход подаётся название темы.

Пример использования:

```
select * from get_document_amount_by_theme('Оплата');
```

Получаем следующий результат:



amount
2

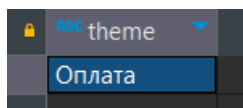
### **get\_theme\_by\_document\_name**

Чтобы определить тему документа по его названию, нужно воспользоваться функцией `get_theme_by_document_name`. В качестве аргументов на вход подаётся название документа.

Пример использования:

```
select * from get_theme_by_document_name('Выплаты за ноябрь');
```

Получаем следующий результат:



theme
Оплата

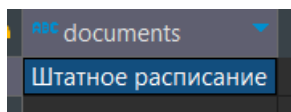
### **get\_document\_name\_with\_max\_amount**

Чтобы определить название документа с наибольшим количеством экземпляров, есть функция `get_document_name_with_max_amount`.

Пример использования:

```
select * from get_document_name_with_max_amount();
```

Получаем следующий результат:



documents
Штатное расписание

### **get\_department\_of\_employee\_with\_max\_requests**

Чтобы определить отдел работника, который чаще всего обращается к архиву, есть функция `get_department_of_employee_with_max_requests`.

Пример использования:

```
select * from get_department_of_employee_with_max_requests();
```

Получаем следующий результат:

department
Бухгалтерия

### **get\_last\_employee**

Чтобы определить сотрудника, который последний обращался к определенному документу, необходимо воспользоваться функцией `get_last_employee`. На вход в качестве аргумента подаётся название документа.

Пример использования:

```
select * from get_last_employee('Штатное расписание');
```

Получаем следующий результат:

employee	requests
Андреев А.А.	2023-12-08

### **change\_phone\_number**

Чтобы поменять номер телефона отдела, нужно воспользоваться функцией `change_phone_number`. На вход в качестве аргументов подаются название отдела и номер телефона.

Пример использования:

```
select change_phone_number('Бухгалтерия', '+7907777777');
```

Запись в таблице «Department» изменилась и стала выглядеть так:

id	name	phone
1	Бухгалтерия	+7907777777

## report\_employee\_by\_department

Чтобы получить отчёт об использовании архива сотрудниками определённого отдела, есть функция `report_employee_by_department`. На вход в качестве аргумента подаётся название отдела.

Пример использования:

```
select * from report_employee_by_department('Бухгалтерия');
```

Получаем следующий результат:

employee	document	request
Иванов И.И.	Штатное расписание	2023-12-06
Андреев А.А.	Штатное расписание	2023-12-08
Иванов И.И.	Положение об оплате труда	2023-12-07

## report\_archive\_summary

Чтобы получить отчёт об архиве, нужно воспользоваться функцией `report_archive_summary`.

Пример использования:

```
select report_archive_summary();
```

Получаем следующий результат:

```
Document Count: 3
Copy Count: 15
Documents Added For The Last Month:
- Штатное расписание
- Выплаты за ноябрь
All Documents Information:
- Положение об оплате труда
--- Amount: 1
--- Cell: 3
--- Shelf: 2
--- Rack: 1
- Штатное расписание
--- Amount: 10
--- Cell: 2
--- Shelf: 1
--- Rack: 1
- Выплаты за ноябрь
--- Amount: 4
--- Cell: 4
--- Shelf: 2
--- Rack: 1
```



Возможные проблемы: БД не предусматривает удаление и добавление тем, отделов и прочего. Для этого можно воспользоваться командами `insert` и `delete`. Подробную инструкцию по этим командам можно найти в документации PostgreSQL.

## ПРИЛОЖЕНИЕ Б

### ЛИСТИНГ ПРОГРАММНОГО КОДА

Создание таблиц:

```
• create table rack (  
•     id int primary key  
• );  
•  
• create table shelf (  
•     id int primary key,  
•     rack_id int references rack on delete restrict  
• );  
•  
• create table cell (  
•     id int primary key,  
•     shelf_id int references shelf on delete restrict,  
•     is_empty bool default true not null  
• );  
•  
• create table theme (  
•     id serial primary key,  
•     name varchar(96) not null  
• );  
•  
• create table documents (  
•     id serial primary key,  
•     name varchar(96) not null,  
•     cell_id int references cell on delete restrict,  
•     theme_id int references theme on delete restrict,  
•     amount int check (amount > 0),  
•     receipt_date date  
• );  
•  
• create table department (  
•     id serial primary key,  
•     name varchar(96) not null,  
•     phone varchar(12) not null  
• );  
•  
• create table employee (  
•     id serial primary key,  
•     name varchar (96) not null,  
•     department_id int references department on delete restrict  
• );  
•  
• create table requests (  
•     id serial primary key,  
•     request_date date,  
•     document_id int references documents on delete restrict,  
•     employee_id int references employee on delete restrict  
• );
```

## Создание функций манипулирования данными:

```
• create function add_document(  
•     document_name_ varchar(96),  
•     theme_name_   varchar(96),  
•     amount_       int  
• ) returns void language plpgsql as $$  
•     declare  
•         theme_id_ int;  
•         cell_id_  int;  
•         shelf_id_ int;  
•         rack_id_  int;  
•  
•     begin  
•         select id from theme where name = theme_name_ into theme_id_  
•         if theme_id_ is null then  
•             insert into theme (name) values (theme_name_) returning  
id into theme_id_  
•         end if;  
•  
•         select id from cell where is_empty = true limit 1 into  
cell_id_  
•         if cell_id_ is null then  
•             select max(id)+1 from cell into cell_id_  
•             select id from shelf where id = (cell_id_+1)/2 into  
shelf_id_  
•             if shelf_id_ is null then  
•                 select max(id)+1 from shelf into shelf_id_  
•                 select id from rack where id = (shelf_id_+1)/2  
into rack_id_  
•                 if rack_id_ is null then  
•                     select max(id)+1 from rack into rack_id_  
•                     if rack_id_ is null then  
•                         set rack_id_ = 1;  
•                         set shelf_id_ = 1;  
•                         set cell_id_ = 1;  
•                     end if;  
•                     insert into rack values (rack_id_);  
•                 end if;  
•                 insert into shelf values (shelf_id_, rack_id_);  
•             end if;  
•             insert into cell values (cell_id_, shelf_id_, false);  
•         end if;  
•  
•         update cell  
•         set is_empty = false  
•         where id = cell_id_  
•  
•         insert into documents (name, cell_id, theme_id, amount,  
receipt_date)  
•         values (document_name_, cell_id_, theme_id_, amount_,  
current_date);  
•     end  
•  
•     $$;  
•  
•     create function delete_document_copy(  
•         document_name_ varchar(96),
```

```

•         copy_amount_ int
•     ) returns void language plpgsql as $$
•         declare
•             amount_ int;
•             id_ int;
•
•         begin
•             select amount-copy_amount_ from documents where name =
document_name_ into amount_;
•             if amount_ > 0 then
•                 update documents
•                 set amount = amount_
•                 where name = document_name_;
•             end if;
•
•             if amount_ <= 0 then
•                 select cell_id from documents where name =
document_name_ into id_;
•                 delete from documents where name = document_name_;
•                 update cell
•                 set is_empty = true
•                 where id = id_;
•             end if;
•         end
•     $$;
•
• create function change_phone_number(
•     department_ varchar(96), phone_ varchar(15)
• ) returns void language plpgsql as $$
•     begin
•         update department
•         set phone = phone_
•         where name = department_;
•     end
•     $$;

```

Создание функций поисковых запросов:

```

• create function get_theme_by_document_name(
•     documents_name_ varchar(96)
• ) returns table(theme varchar) language plpgsql as $$
•     begin
•         return query
•             select theme.name from documents
•             join theme on theme.id = documents.theme_id
•             where documents.name = documents_name_;
•     end
•     $$;
•
• create function get_document_name_with_max_amount()
•     returns table(documents varchar) language plpgsql as $$
•     begin
•         return query
•             select documents.name from documents

```

```

•         where amount = (select max(amount) from documents);
•     end
•
•     $$;
•
•     create function get_document_name_with_max_requests()
•         returns table(documents varchar) language plpgsql as $$
•     begin
•         return query
•             select documents.name
•             from documents inner join requests
•             on documents.id = requests.document_id
•             group by documents.id
•             having count(document_id) = (select max(Cyết) from (
•                 select count(document_id) as Cyết from requests
•                 group by document_id));
•     end
•
•     $$;
•
•     create function get_document_amount_by_theme(
•         theme_name_ varchar(96)
•     ) returns table(amount bigint) language plpgsql as $$
•     begin
•         return query
•             select count(documents.amount)
•             from documents inner join theme
•             on documents.theme_id = theme.id
•             where theme.name = theme_name_
•             group by documents.theme_id;
•     end
•
•     $$;
•
•     create function get_last_employee(
•         document_name_ varchar(96)
•     ) returns table(employee varchar(96), requests date) language plpgsql as $$
•     begin
•         return query
•             select employee.name, request_date from requests
•             inner join employee
•             on employee_id = employee.id
•             inner join documents
•             on documents.id = document_id
•             where documents.name = document_name_
•             and request_date = (select max(request_date) from
requests
•
•                 inner join documents
•                 on documents.id = document_id
•                 where documents.name =
document_name_);
•     end
•
•     $$;
•
•     create function get_department_of_employee_with_max_requests()
•         returns table(department varchar(96)) language plpgsql as $$
•     begin
•         return query

```

```
•      select department.name  
•      from department  
•      where department.id = (select  
employee.department_id  
•  
•          from employee inner join requests  
•          on employee.id = employee_id  
•          group by employee.id  
•          having count(employee_id) =  
(select max(Cvër) from  
•          (select count(employee_id)  
as Cvër from requests  
•          group by employee_id)))  
•      end  
•  $$;
```

## Создание функций отчётов:

```

•      create function report_employee_by_department(
•          department_ varchar(96)
•      )
•      returns table (employee varchar(96), document varchar(96), request date)
language plpgsql as $$
•      begin
•          return query
•              select employee.name, documents.name, request_date
•              from requests
•              inner join employee
•              on employee_id = employee.id
•              inner join documents
•              on document_id = documents.id
•              inner join department
•              on department_id = department.id
•              where department.name = department_;
•      end
•      $$;
•
•      create function report_archive_summary()
•      returns void language plpgsql as $$
•          declare
•              i record;
•          begin
•              raise notice 'Document Count: %', (select count(*) from
documents);
•              raise notice ' ';
•              raise notice 'Copy Count: %', (select sum(amount) from
documents);
•              raise notice ' ';
•              raise notice 'Documents Added For The Last Month: ';
•              for i in select name from documents
•              where receipt_date >= current_date - 31 and receipt_date <=
current_date loop
•                  raise notice '- %', i.name;
•              end loop;
•              raise notice ' ';
•              raise notice 'All Documents Information: ';

```

```

•         for i in select name, amount, cell_id, shelf_id, rack_id from
documents
•
•         inner join cell
•         on cell_id = cell.id
•         inner join shelf
•         on shelf_id = shelf.id loop
•             raise notice '- %', i.name;
•             raise notice '--- Amount: %', i.amount;
•             raise notice '--- Cell: %', i.cell_id;
•             raise notice '--- Shelf: %', i.shelf_id;
•             raise notice '--- Rack: %', i.rack_id;
•         end loop;
•
•     end
•
• $$;

```

### Создание и настройка пользователей:

```

•     create user archive_admin superuser password 'Jsp4j4JcKQXl';
•     create user employee;
•     grant execute on function get_theme_by_document_name to employee;
•     grant execute on function get_document_name_with_max_amount to employee;
•     grant execute on function get_document_name_with_max_requests to employee;
•     grant execute on function get_document_amount_by_theme to employee;
•     grant execute on function get_last_employee to employee;
•     grant execute on function get_department_of_employee_with_max_requests to
employee;
•     grant execute on function report_employee_by_department to employee;
•     grant execute on function report_archive_summary to employee;

```