



**Промышленные СУБД**  
**Лабораторная работа №1**

## Лабораторная работа № 1

### Базы данных. Управление базами данных.

**Цель:** научить использовать SQL-команды для создания баз данных и изменения их структуры, а также выполнять резервное копирование и восстанавливать базы данных из резервных копий. продемонстрировать процесс проектирования базы данных и научить использовать SQL-команды для создания таблиц и управления ограничениями.

**Требования к отчету:** по результатам работы представить отчёт со скриншотами, содержащими SQL-команды и результаты их выполнения для каждой задачи из раздела «Самостоятельная работа». перед выполнением лабораторной работы рекомендуется изучить лекцию №10 «Первые нормальные формы» и лекцию №11 «Четвертая и пятая нормальные формы», в которых рассматривается необходимость и сам процесс нормализации, позволяющий поддерживать целостность базы данных.

### Этапы проектирования базы данных

С точки зрения корпорации Microsoft проектирование базы данных состоит из двух фаз: логического и физического моделирования. Во время логического проектирования разрабатывается модель базы данных, не зависящая от конкретной СУБД. В процессе же физического проектирования создается модель базы данных, оптимизированная для реальной СУБД.

Процесс проектирования базы данных состоит из следующих этапов:

1. Сбор информации.
2. Идентификация объектов и их моделирование.
3. Определение типов информации для каждого объекта.
4. Идентификация отношений.
5. Нормализация.
6. Создание базы данных.

Первые пять этапов относятся к фазе логического моделирования, а шестой этап – к физическому моделированию.

Первые пять этапов относятся к фазе логического моделирования, а шестой этап – к физическому моделированию.

Предлагаемые к реализации базы данных:

### 1. Библиотека

Учет библиографической информации об издании. Учет посетителей библиотеки, отслеживание состояния книг. Наиболее вероятные запросы: поиск книг по параметрам, наличие книг у читателя, статистика популярности издания.

### 2. Бюджет учреждения

План затрат учреждения на очередной финансовый год. Возможность хранить данные за прошлый год. Учет расходов осуществляется по экономическим и функциональным классификаторам, классификатором видов расходов и целевым программам. Первые два классификатора имеют иерархический характер. Наиболее вероятные запросы: свод бюджета (итоговая цифра), сопоставление затрат за разные годы, бюджет в различных разрезах (по экономическим, функциональным статьям, и т.д.).

### 3. Инвентаризация компьютерной техники

Учет компьютеров, периферии, сетевого оборудования в крупных учреждениях. Местоположение, материально ответственное лицо. Ремонт техники. Наиболее вероятные запросы: поиск по типам оборудования, статистика выхода из строя.

### 4. Учет выполнения квалификационных работ

Учет дипломов и курсовых. Авторы, рефераты, классификация по темам (иерархическая). Наиболее вероятные запросы: поиск по ключевым словам, полнотекстовый поиск, статистика по темам.

### 5. Учет кадров

Личная карточка работника, информация об образовании, предыдущем месте работы, и т.д. Наиболее вероятные запросы: поиск человека, статистика по различным параметрам (образование, стаж и т.д.).

### 6. Учет населения

Информация о лице, учет паспортов, регистрация лиц, адресные листки. Наиболее вероятные запросы: поиск по реквизитам лица, поиск документов, поиск по адресам. Статистика по параметрам – возраст, пол и т.д.

### 7. Биллинг

Учет входящего и исходящего интернет-трафика. Учет абонентов. Наиболее вероятные запросы:

- Тестирование
- Спортивная школа.
- Медицинское учреждение.
- Соревновательная деятельность.
- Бухгалтерская книга.
- Учет измерений

- Реестр информационных ресурсов
- Интернет-магазин
- Регистратура медицинского учреждения
- Муниципальный заказ
- Учет льгот
- Интернет-форум
- Учет медицинских услуг
- Учет учащихся
- Интернет-пэйджинг
- Электронный журнал
- Электронная карта
- Адресная система
- Бюро технической инвентаризации
- Электронная биржа

### *Логическая модель*

Разработаем логическую структуру базы данных, в которой хранится информация о преподавателях и читаемых ими дисциплинах, сведения о студентах и посещаемых занятиях.

Установлено, что:

- студент может посещать любое количество дисциплин;
- преподаватели могут вести несколько дисциплин;
- дисциплина читается в некоторой аудитории;
- в одной аудитории читается только одна дисциплина.

Таким образом, выделяются следующие объекты: *Преподаватели*, *Студенты*, *Дисциплины* и *Аудитории*. Между объектами существуют следующие связи: *Студенты–Дисциплины* (многие-ко-многим, реализуется через дополнительный объект *Регистрация*), *Преподаватели–Дисциплины* (один-ко-многим), *Аудитории–Дисциплины* (один-к-одному).

Каждый объект должен содержать следующую информацию.

#### ***Преподаватели:***

- фамилия, имя, отчество – обязательный атрибут;
- дата рождения – обязательный атрибут, не может быть больше даты сегодняшней даты и меньше 1900 года;
- домашний адрес – обязательный атрибут; если не указан, то по умолчанию устанавливается значение ‘*unknown*’;

- телефон – обязательный атрибут, либо задается в формате 00-00-00, либо при его отсутствии по умолчанию указывается 'no'.
- дата найма – обязательный атрибут, не может быть больше даты сегодняшнего дня и меньше даты рождения, по умолчанию устанавливается сегодняшняя дата;
- стаж – вычисляемое поле, определяется разницей между сегодняшней датой и датой найма.

***Студенты:***

- фамилия, имя, отчество – обязательный атрибут;
- дата рождения – обязательный атрибут, не может быть больше даты сегодняшней даты и меньше 1900 года;

***Дисциплины:***

- наименование дисциплины – обязательный атрибут;
- фамилия преподавателя, читающего дисциплину, список посещающих дисциплину студентов, аудитория – внешний ключ.

***Аудитории:***

- номер – обязательный атрибут;
- тип аудитории – обязательный атрибут.

***Физическая модель***

При переходе от логической модели к физической необходимо обеспечить *целостность данных*, под которой понимается согласованность и корректность информации в базе данных.

Существует 4 группы правил целостности:

- *Целостность области значений (или полей)*: определяет набор допустимых для поля значений, в том числе и допустимость значений NULL.
- *Целостность сущностей*: требует наличия у каждой записи таблицы уникального идентификатора – значения первичного ключа.
- *Ссылочная целостность*: гарантирует поддержание постоянной связи между первичным ключом и внешним ключом, т.е. запрещает удаление записи и изменение значения ее первичного ключа, если на нее ссылается внешний ключ, а также запрещает добавление записи со значением внешнего ключа, не соответствующего ни одному значению первичного ключа.
- *Целостность, определяемая пользователем*, указывает специфические правила пользователя, которые нельзя отнести к какой-либо группе.

Целостность данных может быть обеспечена двумя способами.

1. **Декларативный способ.** Критерии, которым должны удовлетворять данные, задаются при определении объекта и являются частью определения базы данных.

*Преимущества:* контроль целостности выполняется автоматически MS SQL Server, такой способ полностью совместим со стандартом ISO SQL:2008<sup>1</sup>.

2. **Процедурный способ.** Критерии описываются в пакетах операторов, выполнение которых и определяет целостность данных.

*Преимущества:* реализуется как на клиенте, так и на сервере с помощью различных программных средств.

### **Декларативная целостность**

Реализуется при помощи ограничений, которые являются наиболее рекомендуемыми для обеспечения целостности данных. Каждый тип целостности обеспечивается соответствующим ограничением:

- **PRIMARY KEY.** Определение поля или группы полей в качестве первичного ключа позволяет уникально идентифицировать каждую запись таблицы, т.к. в этом случае недопустимы повторяющиеся и неопределенные (NULL) значения.

- **FOREIGN KEY.** Позволяет устанавливать связь между полями, содержащими идентичные данные. Данные в этом поле могут принимать значения, определенные в соответствующем первичном ключе, либо значения NULL.

- **UNIQUE.** Определяет уникальность данных в некотором поле таблицы, причем данное ограничение допускает значения NULL. В одной таблице разрешено применение несколько таких ограничений, поля с таким ограничением могут быть использованы в качестве внешних ключей.

- **DEFAULT.** Указывает значение поля по умолчанию, если значение явно не указано при вставке данных. Для поля можно применить только одно такое ограничение.

- **CHECK.** Накладывает ограничение на значение поля в виде логического выражения, что позволяет определить диапазон допустимых значений поля. Причем данное ограничение позволяет ссылаться на значения других полей.

---

<sup>1</sup> Стандарт ISO SQL:2008 является шестой версией языка запросов SQL, которая включает в себя расширения предыдущего стандарта SQL:2003.

**Лабораторная работа выполняется студентом самостоятельно, по результатам выполненных действий формируется отчёт.**

### Задание 0.

Подключитесь к серверу WS0481\SQLEXPRESS с помощью утилиты Management Studio.

Указания к выполнению:

1. Запустите PgAdmin4 через меню Пуск – PgAdmin4
2. Создайте новый сервер: нажмите кнопку «Add New Server» укажите Имя для сервера – WS0481. Во вкладке «Connection» укажите адрес в поле «Host name/address» – localhost, установите пароль и нажмите кнопку «Save».

### Задание 1.

1. Создайте новую базу данных.
2. Спроектируйте базу данных.
3. Создайте схему базы данных.
4. Убедитесь, что база данных создана должным образом.

Для создания базы данных в среде PgAdmin4 необходимо во вкладке обозревателя объектов ПКМ выбрать «Databasees», после чего откроется соответствующее контекстное меню, в котором необходимо выбрать «Create – Database...» (рис.1).

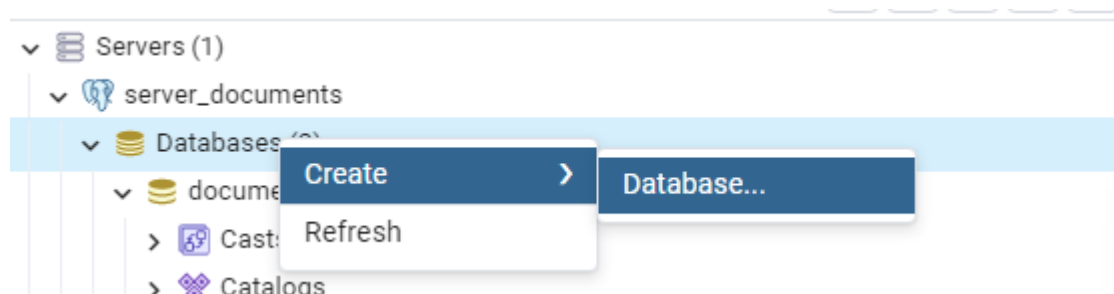


Рис. 1. Создание базы данных в PgAdmin4

Откроется новое окно, в котором необходимо настроить вашу базу данных. После настройки и создания, она отобразится в обозревателе.

После создания базы данных приступите к **созданию таблиц** (5-7 ед.). Каждую таблицу наполните данными.

Работу можно продолжить с помощью графического интерфейса либо с помощью пользовательских запросов.

### С помощью пользовательских запросов:

Чтобы открыть окно для ввода пользовательских запросов, необходимо выбрать ранее созданную базу данных и в панели инструментов выбрать «PSQL Tool» (рис.2).

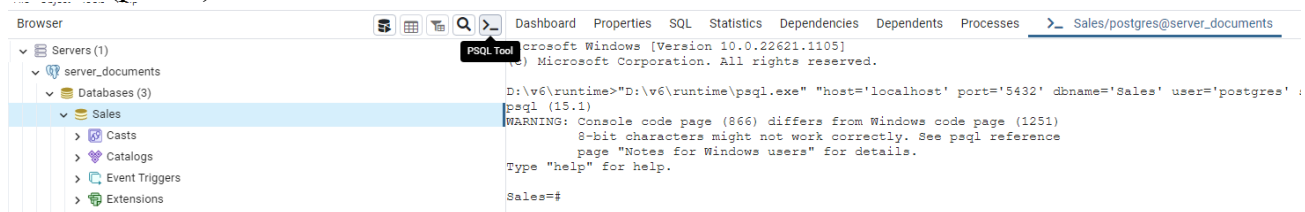


Рис. 2. Открытие окна для ввода пользовательских запросов

Для примера, в созданной базе данных Sales и создадим таблицу Clients(обратите внимания, в открывшемся окне ввода пользовательских запросов, все запросы будут работать только с выбранной базой данных):

```
CREATE TABLE clients( /*Создаем таблицу с названием Clients */
Clients integer generated always as identity(increment by 1), /*Создаем столбцы
присваивая им типы данных */
LastName varchar(20),
FirstName varchar(20),
BirthDate date
);
```

Созданная таблица так же должна появиться в соответствующем разделе обозревателя решений.

В качестве дополнительного примера рассмотрим создание таблицы, хранящей сведения о студентах:

```
CREATE TABLE students (
StudentId integer generated always as identity(increment by 1),
prkStudent integer Primary key,
LastName varchar(20),
FirstName varchar(20),
BirthDate date CONSTRAINT bdCheck CHECK (BirthDate < current_date)
);
```

Для получения информации об используемых ограничениях указанной таблицы предназначена специальный запрос:



```
SELECT constraint_name FROM information_schema.table_constraints  
WHERE table_name = 'Имя таблицы';
```

Для *создания первичного ключа* в уже существующей таблице используется функция:

```
ALTER TABLE Clients  
ADD PRIMARY KEY (Clients);
```

После выполнения в таблице Clients столбец Clients будет объявлен первичным ключом.

Связи между ранее созданными таблицами создаются с помощью *внешнего ключа*:

```
ALTER TABLE имя_таблицы  
ADD CONSTRAINT название_внешнего_ключа FOREIGN KEY имя_столбца  
REFERENCES имя_таблицы_на_которую_ссылается_ключ (имя_столбца);
```

**С помощью графического интерфейса:**

Для создания таблицы с помощью графического интерфейса в обозревателе необходимо выбрать вашу базу данных и раскрыть её. Далее раскрыть «Schemas» и выбрать ПКМ «Tables», в открывшемся контекстном меню выбрать «Create – Table...» (рис.3).

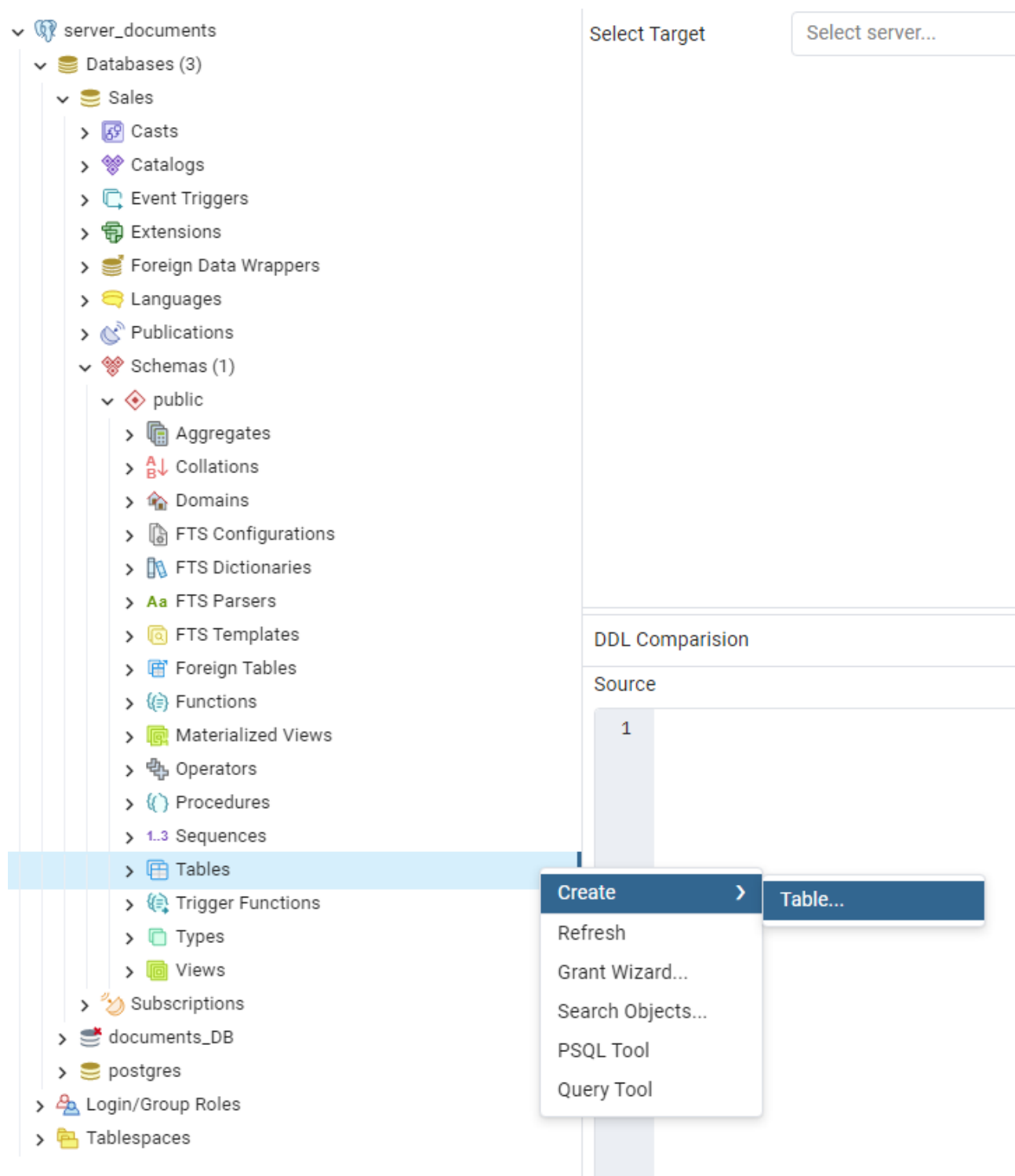
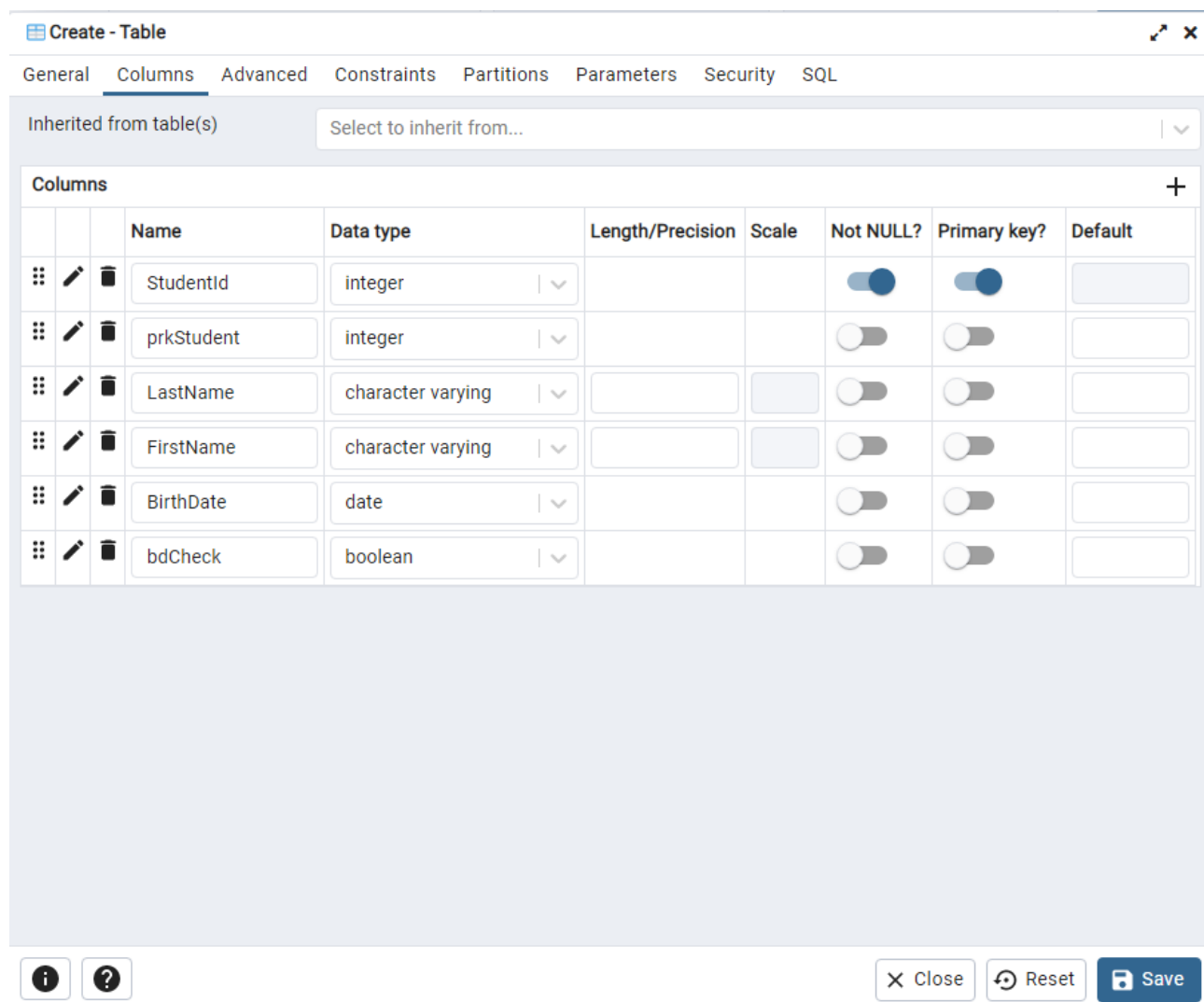


Рис. 3. Создания таблицы с помощью графического интерфейса PgAdmin4.

В качестве примера создадим таблицу Students.

В открывшемся окне выберите имя для таблицы «Students» и создайте столбцы во вкладке «Columns» (рис. 4).



Create - Table

General Columns Advanced Constraints Partitions Parameters Security SQL

Inherited from table(s) Select to inherit from...

Columns

		Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
⋮	✎	StudentId	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
⋮	✎	prkStudent	integer			<input type="checkbox"/>	<input type="checkbox"/>	
⋮	✎	LastName	character varying			<input type="checkbox"/>	<input type="checkbox"/>	
⋮	✎	FirstName	character varying			<input type="checkbox"/>	<input type="checkbox"/>	
⋮	✎	BirthDate	date			<input type="checkbox"/>	<input type="checkbox"/>	
⋮	✎	bdCheck	boolean			<input type="checkbox"/>	<input type="checkbox"/>	

Close Reset Save

Рис. 4. Создания таблицы Clients в PgAdmin4.

Для настройки identity для поля «StudentId» нужно выбрать нужный столбец, слева от имени нажать на «edit row», перейти во вкладку «Constraints» и выбрать «identity» (рис. 5).

The screenshot shows the 'Create - Table' dialog box with the 'Columns' tab selected. The 'StudentId' column is configured with the following settings:

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
StudentId	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Below the table, the 'Constraints' sub-tab is active for the 'StudentId' column. The settings are as follows:

- Default: (empty field)
- Not NULL?: ☒
- Type: ☒ NONE ☒ IDENTITY ☐ GENERATED
- Identity: ALWAYS
- Increment: 1
- Start: (empty field)
- Minimum: (empty field)
- Maximum: (empty field)
- Cache: (empty field)
- Cycled: ☐

At the bottom of the dialog, there are buttons for 'Close', 'Reset', and 'Save'.

Рис. 5. Настройка identity для поля «StudentId».

Для добавления ограничений нужно перейти во вкладку «Constraints» и настроить их по аналогии с identity.

Добавление Primary Key для поля «StudentId» (рис. 6).

The screenshot shows a 'Create - Table' dialog box with the 'Constraints' tab selected. Under the 'Primary Key' sub-tab, a table lists the columns of the table being created. The column 'StudentId' is highlighted, indicating it is the primary key.

Name	Columns
StudentId	StudentId

At the bottom of the dialog, there are buttons for 'Close', 'Reset', and 'Save'.

Рис. 6. Добавления Primary Key для поля «Student»  
Добавление ЧЕК для поля «BirthDate» (рис. 7).

The screenshot shows the 'Create - Table' dialog box with the 'Constraints' tab selected. Under the 'Check' sub-tab, a new constraint named 'bdCheck' is being defined with the expression '"BirthDate" < current\_date'. The 'Definition' sub-tab is also visible, showing the same expression. The 'No inherit?' toggle is off, and the 'Don't validate?' toggle is on. At the bottom, there are buttons for 'Close', 'Reset', and 'Save'.

Рис. 7. Добавления Check для поля «BirthDate»

Чтобы изменить таблицу или добавить какие-то ограничения, нужно выбрать нужную таблицу ПКМ и выбрать поле «Properties...».

***Откройте схему базы данных, убедитесь, что база данных создана должным образом.***

## Задание 2.

1. Просмотрите список параметров базы данных, которые могут быть установлены.
2. Просмотрите список установленных параметров созданной Вами базы данных.
3. Определите использование базы данных только владельцем и в режиме поддержки одного пользователя.
4. Убедитесь в изменении параметров базы данных.

Изучите свойства созданной вами БД, внимательно ознакомьтесь с параметрами. Для ознакомления со списком параметров базы данных необходимо в обозревателе объектов подвести мышь к рассматриваемой БД, нажать ПКМ, выбрать пункт «Properties...».

### Задание 3.

1. Переименуйте созданную Вами базу данных.

Переименовать базу данных можно одним из возможных способов, с помощью графического интерфейса в обозревателе объектов или с помощью выполнения соответствующего кода:

```
ALTER DATABASE «Имя_базы_данных»  
RENAME TO «Новое_имя_базы_данных»;
```

### Задание 4.

1. Для своей базы данных установите возможность автоматического сжатия данных.

Установить возможность автоматического сжатия данных можно изменив соответствующий пункт в меню «Properties...» для нужной таблицы (рис. 8).

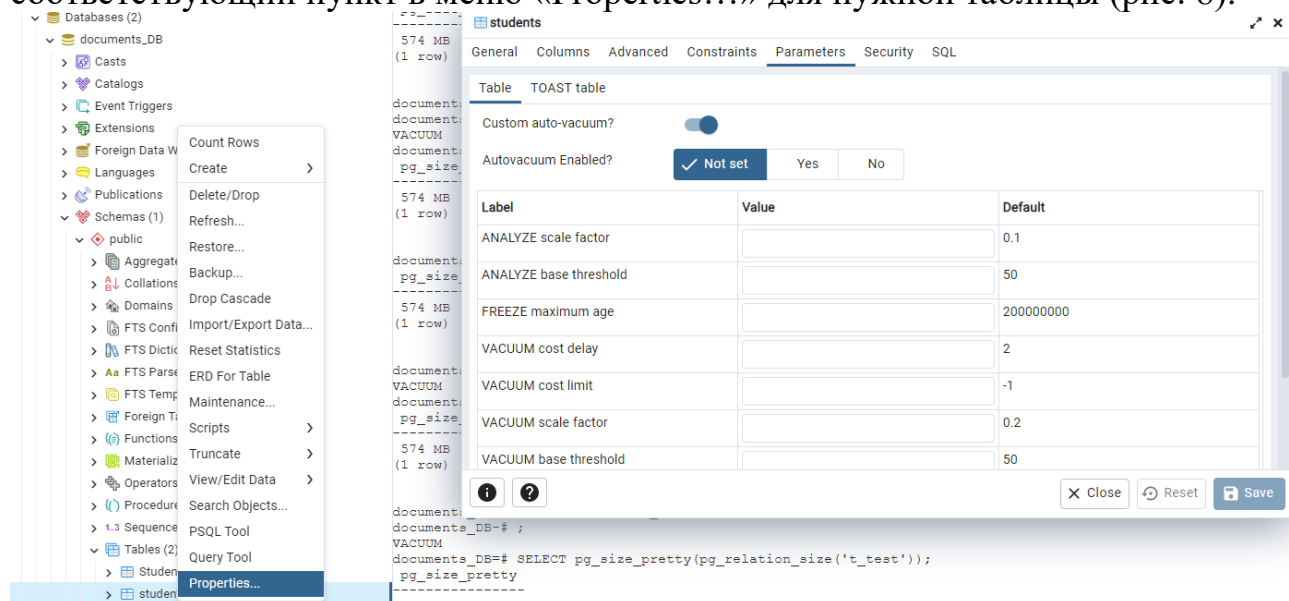


Рис. 8. Включение авто-сжатия для определенной таблицы.

### Задание 5.

1. Определите сведения о дисковом пространстве, занимаемом базой данных.
2. Произведите сжатие базы данных так, чтобы она содержала только 25% пространства, доступного ей на текущий момент.
3. Докажите правильность выполненного действия.

Для выполнения этого задания необходимо открыть окно показанное на рис.8 и в поле «VACUUM scale factor» указать 0.75(75%). Чтобы выполнить сжатие данных, необходимо выполнить следующую команду:

VACUUM full «Имя\_таблицы»;

<https://postgrespro.ru/docs/postgrespro/10/sql-vacuum>

### Задание 6.

1. Создайте резервную копию базы данных, задав физическое имя устройства резервного копирования.
2. Выполните резервное копирование журнала транзакций базы данных.

Данное задание можно выполнить одним из возможных способов, с помощью графического интерфейса в обозревателе объектов, воспользовавшись контекстным меню по ПКМ (рис. 8).

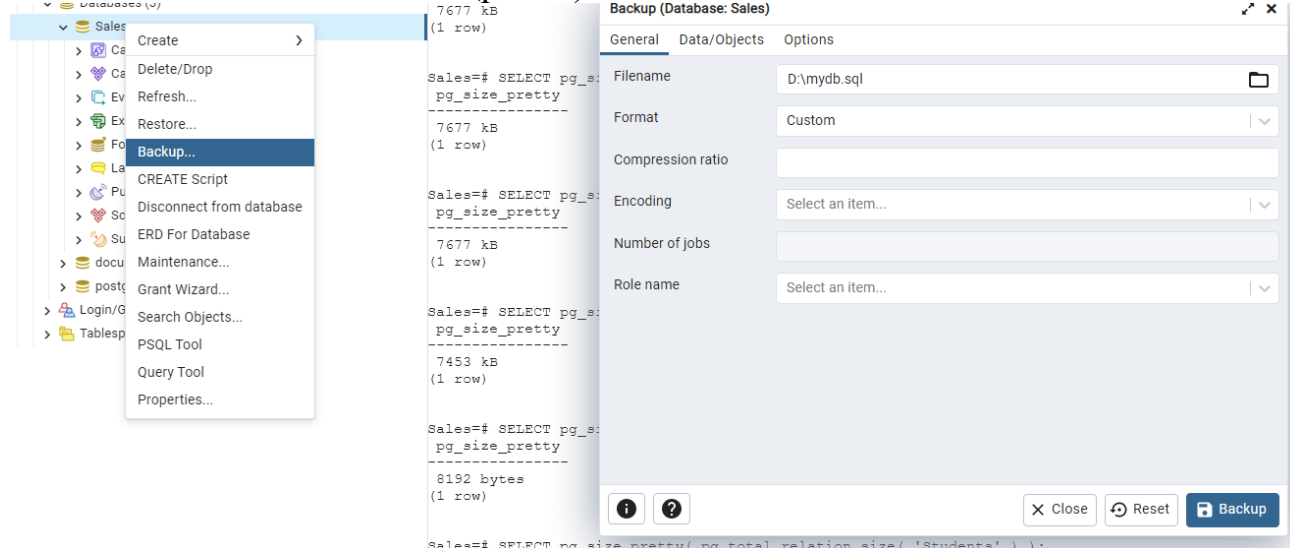


Рис. 8. Резервное копирования базы данных.

### Задание 7.

1. Удалите базу данных.
2. Восстановите удаленную базу с устройства с указанным Вами физическим именем.
3. Докажите правильность выполненного действия.

Для восстановления базы данных нужно нажать правой кнопкой мыши по нужной базе данных в обозревателе объектов и выбрать функцию «Restore...», после чего выбрать заранее созданный файл с резервной копией.

### Задание 8.



1. Открепите базу данных от сервера.
2. Прикрепите базу данных к серверу.
3. Докажите правильность выполненного действия.

В области «Обозреватель объектов» в ветке «Databases» выбираем БД, которую нужно отключить и через контекстное меню «Disconnected from database\server» отсоединяем выбранную базу данных. Для присоединения нужно выбрать «Databases» и в контекстном меню нажать кнопку «Connect Database».