Министерство науки и образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э.

Баумана

(национальный исследовательский университет)»

(МГТУ им. Н.Э. Баумана)



**МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ ПО**

**КУРСУ «БАЗЫ ДАННЫХ»**

**Лабораторная работа №1**

**«Введение в SQL»**

Авторы:

Скворцова М.А., [magavrilova@bmstu.ru](mailto:magavrilova@bmstu.ru)

Лапшин А.В.

Фомин М.М.

Москва, 2023

**Общие сведения**

# Сокращения

SQL– Structured Query Language («язык структурированных запросов»)

БД – База данных

СУБД – Система управления базами данных

РБД – Реляционная база данных

# Краткая информация о СУБД PostgreSQL

**PostgreSQL** – комплекс программ, относящийся к классу систем управления базами данных. Когда эта система выполняется, мы называем ее сервером PostgreSQL, или экземпляром сервера.

Данные, которыми управляет PostgreSQL, хранятся в базах данных. Один экземпляр PostgreSQL может одновременно управлять несколькими базами, которые вместе называются кластером баз данных.

Чтобы кластер можно было использовать, его необходимо инициализировать (создать). Каталог, в котором размещаются все файлы, относящиеся к кластеру, обычно называют - **PGDATA**, по имени переменной окружения, указывающей на этот каталог.

# Установка программного обеспечения и ссылки на источники

## Установка СУБД PostgreSQL 11.5

Для выполнения заданий практикума предлагается использовать СУБД PostgreSQL 11.5 и новее, дистрибутив которой можно скачать по ссылке: https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

По ссылке выше нужно будет выбрать операционную систему и скачать установщик, который предложит установить нужные компоненты для работы с PostgreSQL (см. рисунок 1). Обязательные компоненты - сам PostgreSQL Server и pgAdmin 4, остальное не существенно.

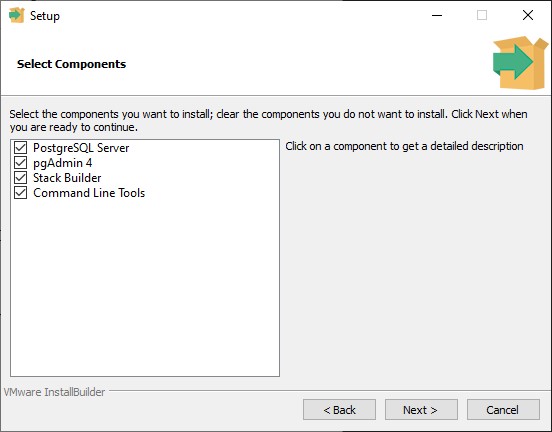


Рис. 1. Выбор компонентов

Следующим выбором будет путь, где планируется хранить ваши базы данных (см. рисунок 2).

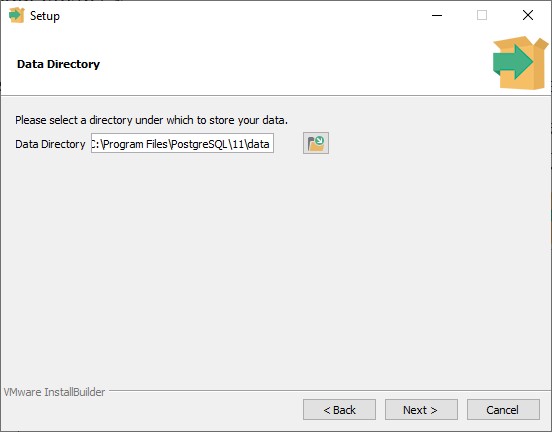


Рис. 2. Путь для хранения данных

Далее нужно будет установить пароль для вашей базы данных (см. рисунок 3), пароль рекомендуется где-нибудь записать.

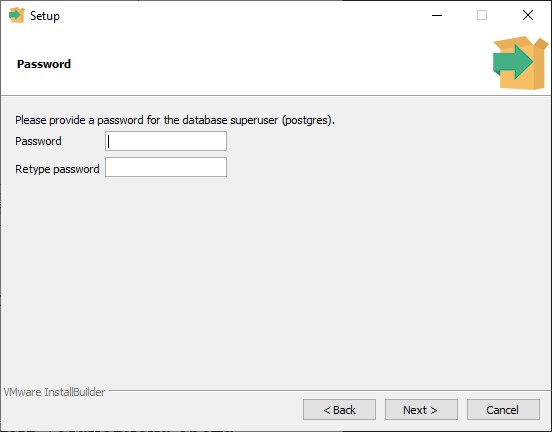


Рис. 3. Установка пароля

Так как СУБД является сервером, вам предложат выбрать порт, по которому будет идти соединение с ним, лучше всего оставить значение порта по умолчанию, если конечно в вашей системе он не занят (см. рисунок 4).

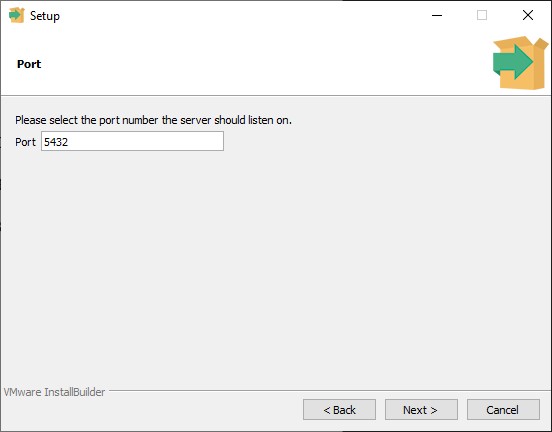


Рис. 4. Установка порта соединения

Последним действием можно выбрать кодировку (язык) данных, его так же можно оставить по умолчанию (Default locale – будет опираться на настройки системы).

После установки предлагается установить дополнительные приложения для СУБД. Этот пункт можно пропустить, а в дальнейшем если понадобится что-то установить просто наберите в строке поиске windows название установщика (Application Stack Builder) и дополните вашу среду.

**Установка и настройка среды pgAdmin 4**

В результате установки СУБД PostgreSQL должна быть установлена среда pgAdmin 4 для подключения к СУБД и выполнения SQL-запросов. Если этого не произошло, то ее необходимо установить отдельно, скачав по ссылке: https://www.pgadmin.org/download/

После установки, программу можно найти с помощью строки поиска (pgAdmin4). С запуском среды она предложит придумать мастер пароль для вашей системы (см. рисунок 5). Устанавливать его необязательно.

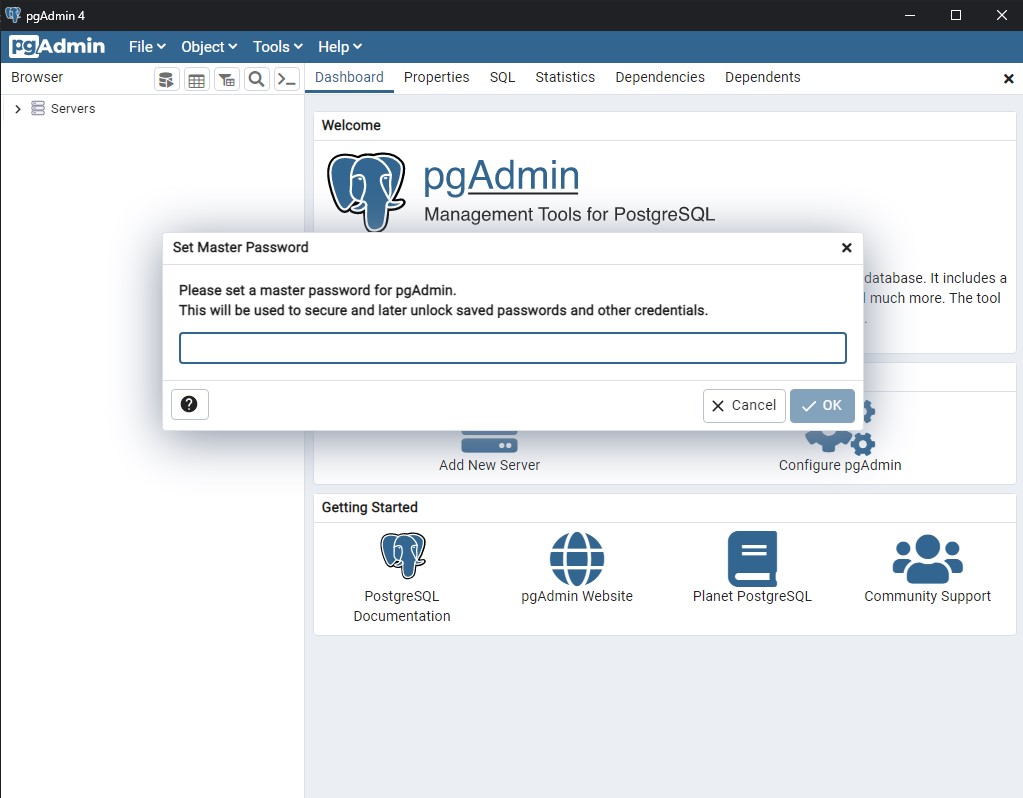


Рис. 5. Мастер пароль

Далее если вам нужно поменять тему или язык (либо что-то ещё) можно зайти в настройки (см. риc. 6).

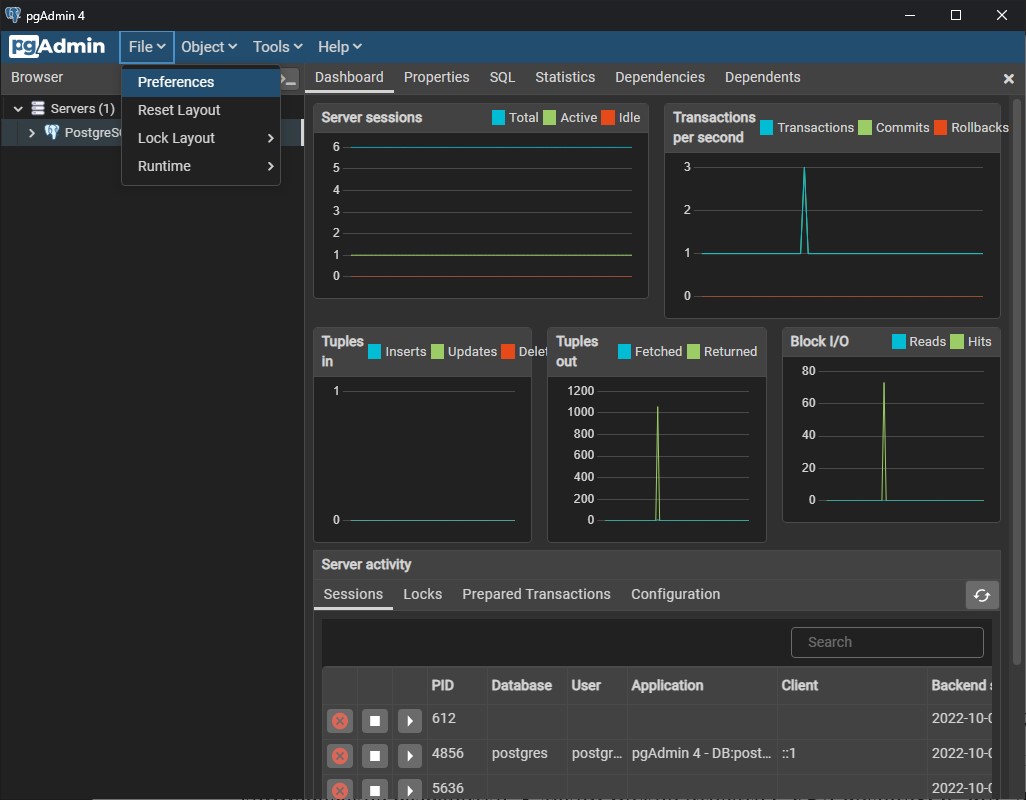


Рис. 6. Настройки

**Коммерческая версия PostgreSQL Pro**

СУБД PostgreSQL является бесплатным и свободно распространяющимся ПО, однако существуют также коммерческие разработки, например, СУБД PostgreSQL Pro. Для нее существует достаточно подробная документация на русском языке, которая также актуальна для СУБД PostgreSQL, так как СУБД PostgreSQL Pro является ее расширением и наследует всю функциональность:

[https://postgrespro.ru/docs/postgrespro/11/index](https://postgrespro.ru/docs/postgrespro/11/index%20)

**Англоязычная документация**

Также можно воспользоваться англоязычной документацией для СУБД

PostgreSQL: [https://www.postgresql.org/docs/11/index.html](https://www.postgresql.org/docs/11/index.html%20)

## Установка СУБД PostgreSQL в Docker

Скачайте «Docker» с официального сайта и установите:

[https://www.docker.com/products/docker-desktop/](https://www.docker.com/products/docker-desktop/%20)

Настоятельно рекомендуется использовать актуальную версию ПО.

Официальные образы PostgreSQL расположены тут:

[https://hub.docker.com/\_/postgres](https://hub.docker.com/_/postgres%20)

На этой же странице описан алгоритм установки образа в Docker.

Зачем добавлять PostgreSQL в контейнер? Главное преимущество контейнеризации — сохранение и быстрое развертывание зависимостей различных программных пакетов необходимых для работы СУБД.

## Альтернатива PgAdmin

Универсальный инструмент для управления СУБД - система DBeaver. Это бесплатный многоплатформенный инструмент для работы разработчиков, администраторов, аналитиков и других специалистов, которым необходимо работать с БД. Поддерживает все популярные базы данных: MySQL, PostgreSQL, SQLite, Oracle, DB2, SQL Server, Sybase, MS Access, и др.

Скачать DBeaver можно на официальном сайте: https://dbeaver.io/

# Порядок сдачи и сроки

Практические задания предполагают предварительную теоретическую и практическую подготовку студента, которую он демонстрирует в процессе сдачи задания. Кроме того, в процессе сдачи задания студент может получить дополнительные вопросы, ответы на которые необходимо дать до окончания защиты лабораторной работы.

Сроки сдачи каждой лабораторной не более 3х недель. После истечения данного срока, за сдачу лабораторной работы будут сниматься баллы. Готовый отчет необходимо загрузить на страницу курса «Базы данных. Лабораторные работы и КР». Защита лабораторных работ будет проходить на неделе, следующей за дедлайном по лабораторной работе №4.

На защиту вы приносите только титульный лист, сам отчет, содержащий в себе результат выполнения всех четырех лабораторных работ в формате ФИО\_Группа\_Вариант\_Названиекурса.pdf загружаете также на соответствующую страницу курса.

Практические задания выполняются по индивидуальным вариантам. Вариант содержит в себе предметную область для лабораторных заданий: <http://sp.cs.msu.ru/prak3/prak3_tasks.pdf>

Узнать актуальный номер своего варианта можно в ЭУ:

<https://eu.bmstu.ru>

# Лабораторная работа №1. Основы языка SQL. SELECT запросы

**Цель:**

Первое практическое задание заключается в знакомстве со средой pgAdmin и написании SQL-запросов с использованием оператора SELECT.

**Задачи:**

* Получить опыт взаимодействия с консолью psql;
* Ознакомится со средой pgAdmin 4;
* Изучение основ языка SQL.

## Теоретическая часть

### Подготовка

Если будете повторять приведённые примеры и выполнять задания в модулях, то стоит посмотреть, раздел «Инструкция по добавлению учебной базы данных» настоящей методички.

После проделанных манипуляций в pgAdmin должна появиться БД с название demo, как на рисунке 7.

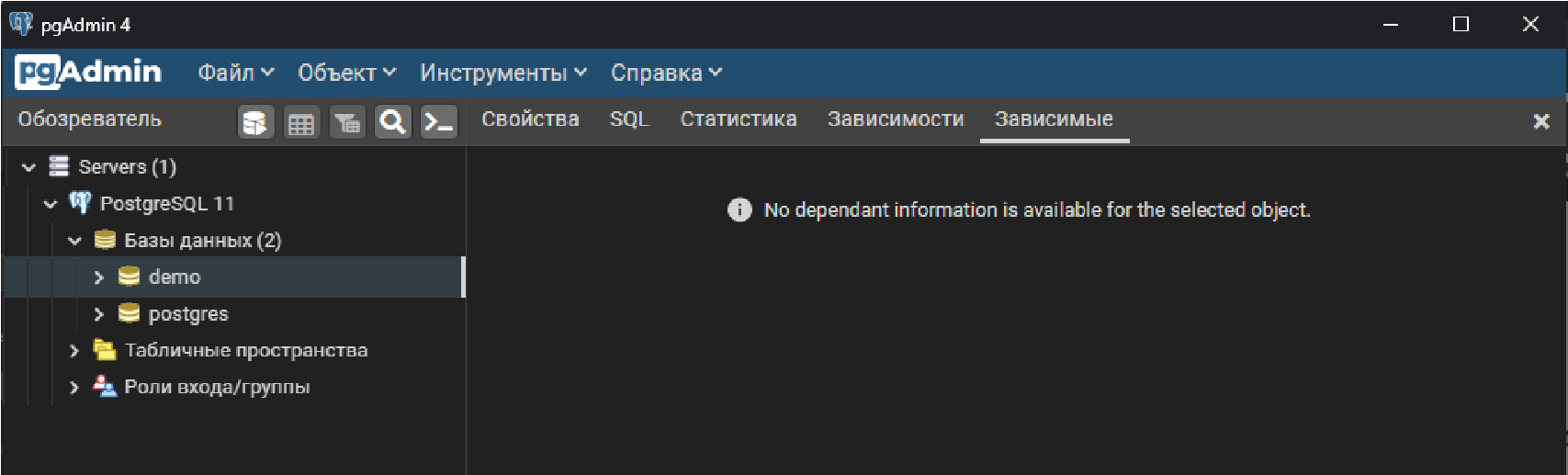


Рис. 7. Главное меню

**Что такое SQL?**

***SQL*** (*structured query language* - «язык структурированных запросов») – формальный, непроцедурный, декларативный язык программирования, применяемый для создания, модификации и управления данными в произвольной реляционной базе данных, управляемой соответствующей системой управления базами данных (СУБД).

На рисунке 8 показана схема работы пользователя с СУБД. Согласно этой схеме, в вычислительной системе имеется база данных, в которой хранится информация о предметной области. Если вычислительная система относится к сфере бизнеса, то в базе данных могут содержаться сведения о материальных ценностях, выпускаемой продукции, объемах продаж и зарплате. Комплекс программ, которая управляет базой данных, называется системой управления базой данных, или же СУБД.

На самом деле СУБД - это совокупность программных, лингвистических и методических средств общего или специального назначения, обеспечивающих управление созданием и использованием БД.

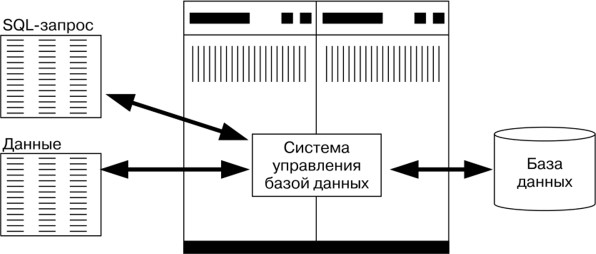


Рис. 8. Применение SQL для обращения к базе данных

SQL позволяет пользователю выполнять такие действия как:

* описывать данные (их структуру);
* определять данные в БД и управлять ими;
* читать, создавать и удалять данные, а также другие объекты БД, такие как представления или хранимые процедуры;
* разграничивать доступ пользователей к объектам БД.

### Все операторы языка SQL

Одним из основных требований к системам управления базами данных является наличие высокоуровневых средств выполнения запросов. В системах, реализующих реляционную модель данных, в качестве такого средства используется язык SQL. Фактически этот язык содержит полный набор операций, необходимых для выполнения любых действий с базой данных. Стандарты SQL предусматривают подразделение средств языка на несколько категорий. Далее представлены все операторы языка SQL, разбитые на группы в соответствии с современным стандартом:

* Операторы определения данных (*Data Definition Language,* ***DDL***) обеспечивают создание, модификацию и удаление элементов описания структуры базы данных, как логической, так и структуры хранения :
  + CREATE создает объект БД,
  + ALTER изменяет объект,
  + DROP удаляет объект;
* Операторы манипуляции данными (*Data Manipulation Language,* ***DML****)* обеспечивают выполнение поиска, извлечения, добавления, изменения и удаления данных, определенных в описании логической структуры базы данных, но не позволяют изменять эту структуру:
  + SELECT выбирает данные, удовлетворяющие заданным условиям,
  + INSERT добавляет новые данные,
  + UPDATE изменяет существующие данные,
  + DELETE удаляет данные;
* Операторы определения доступа к данным (*Data Control Language,* ***DCL***) обеспечивают разграничение доступа пользователей к данным:
  + GRANT предоставляет пользователю разрешения на определенные операции с объектом БД,
  + REVOKE отзывает ранее выданные разрешения,
* Операторы управления транзакциями (*Transaction Control Language,* ***TCL****)* обеспечивают сохранение БД в согласованном состоянии:
  + START TRANSACTION отмечает начало транзакции,
  + COMMIT сохраняет все изменения текущей транзакции,
  + ROLLBACK откатывает все изменения текущей транзакции,
  + SAVEPOINT делит транзакцию на более мелкие участки.

В расширениях и диалектах языка SQL присутствуют дополнительные средства, обеспечивающие описание особенностей, специфических для конкретной СУБД. Чаще всего это определения конкретных структур хранений и их параметров.

В данной лабораторной рассмотрим только один оператор из группы DML - это SELECT.

Оператор SELECT, синтаксис которого приведен ниже, извлекает данные упомянутые в предложении SELECT из таблиц, указанных в предложении FROM в соответствии с правилами, устанавливаемыми в остальных предложениях оператора SELECT.

**SELECT** [{ALL | DISTINCT}] *извлекаемый\_ элемент*

[AS *псевдоним*][, ...]

**FROM** {*таблица* [[AS] *псевдоним*] | *представление*

[[AS] *псевдоним*]}[, ...]

[[*тип\_ соединения*] JOIN *условие\_ соединения*]

[**WHERE** *условие\_ отбора*] [{AND | OR | NOT} *условие\_ отбора* [...]]

[**GROUP BY** *выражение\_ группировки*]

[**HAVING** *условие\_ отбора*]]

[**ORDER BY** {*выражение\_сортировки* [ASC | DESC]}[, ...]]

[**LIMIT** *количество\_строк*]

Таблица 1. Предложения оператора SELECT

|  |  |
| --- | --- |
| **Предложение** | **Описание** |
| FROM | Перечисляет используемые в запросе таблицы из базы данных. Все перечисленные таблицы объединяются при помощи «декартова произведения» |
| WHERE | Определяет условие, которое используется для выборки строк. Условие вычисляется для каждой строки таблицы, полученной в предложении FROM, и в итоговую таблицу попадают только строки для которых это условие - TRUE |
| GROUP BY | Используется для группировки строк |
| HAVING | Определяет условие, которое используется для выборки групп. Работает аналогично предложению WHERE, но не для строк, а для групп. |
| ORDER BY | Определяет выражение по которому производится сортировка итоговой таблицы |
| LIMIT | Команда LIMIT задает ограничение на количество записей итоговой таблицы |

**Типы данных в языке SQL**

Изначально применение СУБД ограничивалось преимущественно решением финансово-экономических задач. Независимо от модели данных, базы данных обрабатывали такие основные типы данных:

* числовые – наиболее используемыми являются целочисленные, вещественные и денежные (финансовые);
* символьные, например, значения данных «четверг», «столбец», «менеджер»;
* логические, которые принимают значения «истина» или «ложь»;
* даты, которые задаются с помощью типа «Дата» или в виде обычных символьных данных.

В разных СУБД эти типы несущественно отличались названиями, диапазоном значений и видом представления. С развитием информационных технологий разрабатывались специализированные системы обработки данных (системы обработки видеоизображений, геоинформационные системы и т.д.), что привело к введению в СУБД их разработчиками поддержки новых типов данных:

* время и дата/время, которые были предназначены для хранения информации о времени и/или дате (диапазоне дат);
* символьные переменной длины, которые хранили текстовую информацию большой длины (например, документ);
* двоичные для хранения графических, аудио- и видео- объектов, хронологической, пространственной и другой специальной информации. Такие данные часто называются мультимедиа-данными. Реляционные системы могут только хранить мультимедиа-данные, а создавать и редактировать такие данные способны только внешние специализированные программы;
* данные в формате XML.

Далее в таблице 2. приведены все типы данных определенные в стандарте SQL2003.

Таблица 2. Типы данных SQL2003

|  |  |  |
| --- | --- | --- |
| **Категория** | **Примеры типов данных (и сокращенные названия)** | **Описание** |
| BINARY | BINARY LARGE OBJECT (BLOB) | Этот тип хранит массив двоичных данных. Значение хранится без указания кодовой страницы и ограничения на длину. |
| BOOLEAN | BOOLEAN | Этот тип данных используется для хранения булевых значений (TRUE или FALSE) |
| CHARACTER | CHAR  CHARACTER VARYING (VARCHAR, VARCHAR2) | боры символов конкретной кодовой страницы. Тип VARCHAR допускает хранение значений переменной длины, а тип CHAR – только фиксированной. Также в значениях типа VARCHAR автоматически удаляются пробелы в конце строки, а в CHAR, напротив, все оставшееся место заполняется пробелами. |
| NATIONAL CHARACTER (NCHAR),  NATIONAL VARYING CHARACTER (NCHAR VARYING) | Национальные символьные типы данных используются для поддержки специальной кодовой страницы, определяемой реализацией. |
| CHARACTER LARGE OBJECT (CLOB) | Типы данных CHARACTER LARGE OBJECT (CLOB) и BINARY LARGE OBJECT (BLOB)  относятся к типу LARGE OBJECT |
| NATIONAL CHARACTER LARGE OBJECT (NCLOB) | То же, что и CHARACTER LARGE OBJECT, но с поддержкой кодовой страницы, определяемой реализацией. |
| DATALINK | DATALINK | Описывает указатель на файл или другой внешний источник данных, не являющийся частью базы данных. |
| NTERVAL | NTERVAL | Описывает набор значений времени или временной интервал. |
| COLLECTION | ARRAY  MULTISET | ARRAY был введен в SQL99, а MULTISET был добавлен в SQL2003. ARRAY – это упорядоченная коллекция элементов, имеющая ограниченную длину. MULTISET – неупорядоченная коллекция, не имеющая ограничений на размер. Элементы MULTISET и ARRAY должны быть стандартного типа. |
| NUMERIC | INTEGER(INT)  SMALLINT  BIGINT  NUMERIC(p,s)  DEC[IMAL](p,s)  FLOAT(p,s)  REAL  DOUBLE PRECISION | Эти типы данных хранят точные (целые и десятичные) или приблизительные (с плавающей точкой) числовые значения. INT, BIGINT и SMALLINT хранят точные числовые значения с фиксированным масштабом и точностью. NUMERIC и DEC хранят точные числовые значения с настраиваемыми масштабом и точностью. FLOAT хранит приблизительное числовое значение с настраиваемой точностью, а точность REAL и DOUBLE PRECISION фиксирована. Вы можете указать точность (precision) и масштаб (scale) типов DECIMAL, FLOAT и NUMERIC для указания общего числа хранимых цифр и десятичных знаков. INT, SMALLINT, DEC иногда называют точными типами данных, а FLOAT, REAL и DOUBLE PRECISION – приблизительными. |
| TEMPORAL | DATE  TIME  TIME WITH TIMEZONE  TIMESTAMP  TIMESTAMP WITH TIMEZONE | Эти типы данных используются для хранения значений времени. TIME и DATE говорят сами за себя. Типы данных с суффиксом WITH TIMEZONE содержат указание на часовой пояс. Тип TIMESTAMP используется для хранения более точного момента времени. Эти типы данных часто называют временными. |
| XML | XML | Хранит XML данные и может использоваться везде, где может использоваться любой другой стандартный тип данных (например, для столбца таблицы, поля в записи и т. д.). Операции со значениями типа XML предполагают наличие древовидной внутренней структуры данных. Внутренняя структура основана на XML Information Set Recommendation (Infoset), и использует новый внутренний информационный элемент с названием корневой информационный элемент XML |

**Операции в языке SQL**

Таблица 3. Арифметические операции (здесь a=10 и b=20)

|  |  |  |
| --- | --- | --- |
| Операция | Описание | Пример |
| + (сложение) | Сложение значений | a + b = 30 |
| — (вычитание) | Вычитание правого операнда из левого | b - a = 10 |
| \* (умножение) | Умножение значений | a \* b = 200 |
| / (деление) | Деление левого операнда на правый | b / a = 2 |
| % (деление по модулю) | Деление левого операнда на правый с остатком (возвращается остаток) | b % a = 0 |

Таблица 4. Операции сравнения (здесь a=10 и b=20)

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | Описание | | Пример |
| = | Определяет равенство значений | | a = b -> false |
| != | Определяет НЕравенство значений | | a != b -> true |
| <> | Определяет НЕравенство значений | | a <> b -> true |
| > | Значение левого операнда больше значения правого операнда? | | a > b -> false |
| < | Значение левого операнда меньше значения правого операнда? | | a < b -> true |
| >= | Значение левого операнда больше или равно значению правого операнда? | | a >= b -> false |
| <= | Значение левого операнда меньше или равно значению правого операнда? | | a <= b -> true |
| !< | Значение левого операнда НЕ меньше значения правого операнда? | | a !< b -> false |
| !> | Значение левого операнда НЕ больше значения правого операнда? | | a !> b -> true |
| Таблица 5. Логические операции (проводятся над булевыми выражениями) | | | |
| Операция | | Описание | |
| ALL | | TRUE, если булево выражение истинно для всех строк, возвращаемых подзапросом | |
| AND | | TRUE, если оба булевых выражения истинны | |
| ANY | | ANY и его синоним SOME возвращает TRUE если булево выражение истинно хотя бы для одной строки подзапроса (множества). | |
| BETWEEN | | TRUE, если операнд попадает в заданный интервал | |
| EXISTS | | TRUE, если подзапрос возвращает хотя бы одну строку | |
| IN | | TRUE, если операнд равен хотя бы одному выражению из списка или строке из результата подзапроса | |
| LIKE | | TRUE, если операнд удовлетворяет шаблону | |
| NOT | | Заменяет булево значение на противоположное | |
| OR | | TRUE, если хотя бы одно из двух булевых выражений истинно | |
| IS NULL | | TRUE, если поле не заполнено (содержит NULL) | |

**Термины**

***База данных*** (БД) ‑ организованная в соответствии с определёнными правилами и поддерживаемая в памяти компьютера совокупность данных, характеризующая актуальное состояние некоторой предметной области и используемая для удовлетворения информационных потребностей пользователей.

***Реляционная база данных*** представляет собой набор взаимосвязанных двухмерных таблиц. Эта модель предложена сотрудником фирмы “IBM” Эдгаром Коддом в 1970 году.

Таблица реляционной БД соответствует одной сущности предметной области и состоит из фиксированного числа полей, собранных в записи, каждая запись соответствует экземпляру сущности.

Большинство современных СУБД поддерживают реляционные БД и, следовательно, являются реляционными.

Все реляционные СУБД поддерживают язык SQL.

***Сущность (****объект****)*** - предмет, человек или событие, о котором собирается и хранится информация в БД.

***Таблица*** - совокупность записей с фиксированным числом полей. Таблица содержит данные о всех экземплярах сущности.

***Запись*** – данные об одном конкретном экземпляре сущности - предмете, человеке или событии. Запись состоит из полей.

Пример. Сущность “СОТРУДНИК”, экземпляр сущности - данные об Иванове И. составляют одну запись таблицы «Сотрудники»

***Поле записи (****атрибут, столбец таблицы****)*** - отдельная характеристика (свойство) сущности.

***Домен*** - область значений одного или нескольких атрибутов.

Далее приведено соответствие терминов в различных областях знаний.

|  |  |  |
| --- | --- | --- |
| База данных | Реляционная алгебра | ГОСТ (EXCEL) |
| Таблица | Отношение | Таблица |
| Запись | Кортеж | Строка |
| Поле | Атрибут | Столбец |

**Как писать запросы в среде pgAdmin4**

Переходя к практике, начнём с самого простого, а именно получение данных с таблицы без каких-либо условий с помощью операторов SELECT и FROM.

Для того чтобы открыть среду написания запросов нужно нажать на базы данных demo и сверху загорится нужная кнопка «Запросник», как на рисунке 9.

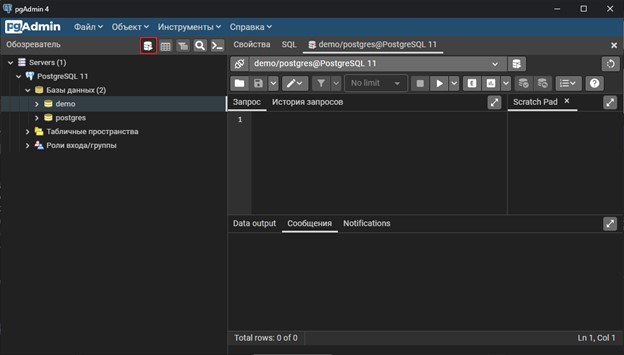


Рис. 9. Запросник

Проанализируем структуру БД demo, и начнем с главного в этой лабораторной работе, а именно со схем. В PostgreSQL схема - это пространство имен, которое содержит именованные объекты базы данных, такие как таблицы, представления, индексы, типы данных, функции и т.д. Схема принадлежит конкретному пользователю (хозяину схемы) и доступна только ему. Для доступа другим пользователям хозяину схемы должен дать соответствующие разрешения. По умолчанию при создании базы данных создается схема «public» доступная всем, а каждая создаваемая таблица, у которой не указана привязка к какой-либо схеме, будет автоматически принадлежать к «public». В нашей же базе есть отдельная схема «booking» - с ней, а точнее с её таблицами, и будут связанны большинство запросов (см. рисунок 10).

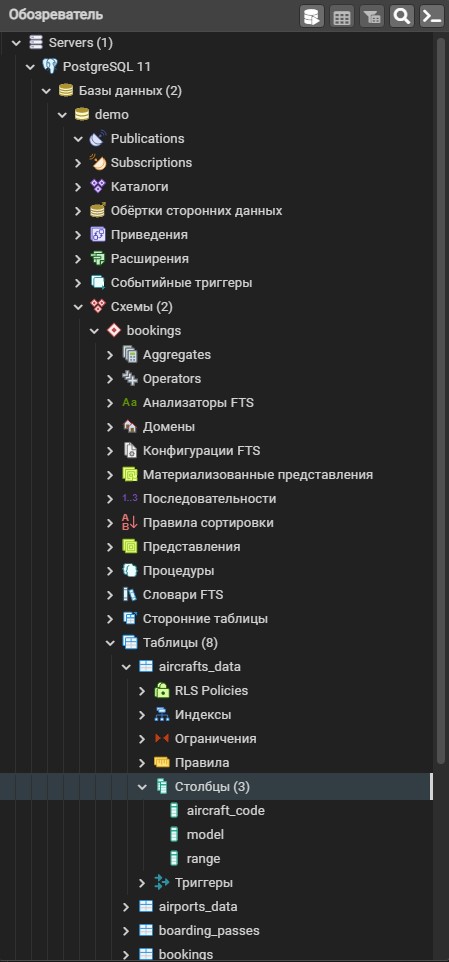


Рис. 10. Структура БД

### Оператор SELECT

Оператор SELECT, как и говорилось выше предназначен для извлечения записей из таблиц, которые определяются предложением FROM. Если входных таблиц нет, можно записать оператор «SELECT (10-4)/3» или же просто «SELECT 2», мы получим результат в виде цифры 2. Однако, чаще всего оператор извлекает данные из таблиц и после соответствующих преобразований мы получаем итоговую таблицу. К примеру, нужно посчитать скидку 30% на билет.

**Запрос 1:**

SELECT amount, ROUND ((amount / 100), 2) \* 30 as sale

FROM ticket\_flights

LIMIT 100;

**Последовательность выполнения запроса такова:**

* С помощью предложения SELECT определяются какие данные попадут в итоговую таблицу, в этом запросе выдаются данные о полной стоимости билета, (amount) и подсчёт скидки (функция ROUND используется для округления результата – в данном случае до двух цифр после запятой)
* С помощью ключевого слова «as» назначается имя будущему столбцу в таблице результатов запроса, в данном случае -«sale»
* С помощью предложения FROM определяем таблицу откуда будем брать данные, в нашем случае - «ticket\_flights», из этой таблицы мы и берем значения поля «amount»
* Предложение «LIMIT 100» ограничивает количество записей в итоговой таблице до 100.

Как это выглядит в среде «pgAdmin4» можно посмотреть рисунке 11.

**Для запуска кода нужно нажать кнопку «Play», на рисунке выделено синей рамкой!**

**А для сохранения запроса нужно нажать кнопку с «дискетой», на рисунке выделено красной рамкой!**

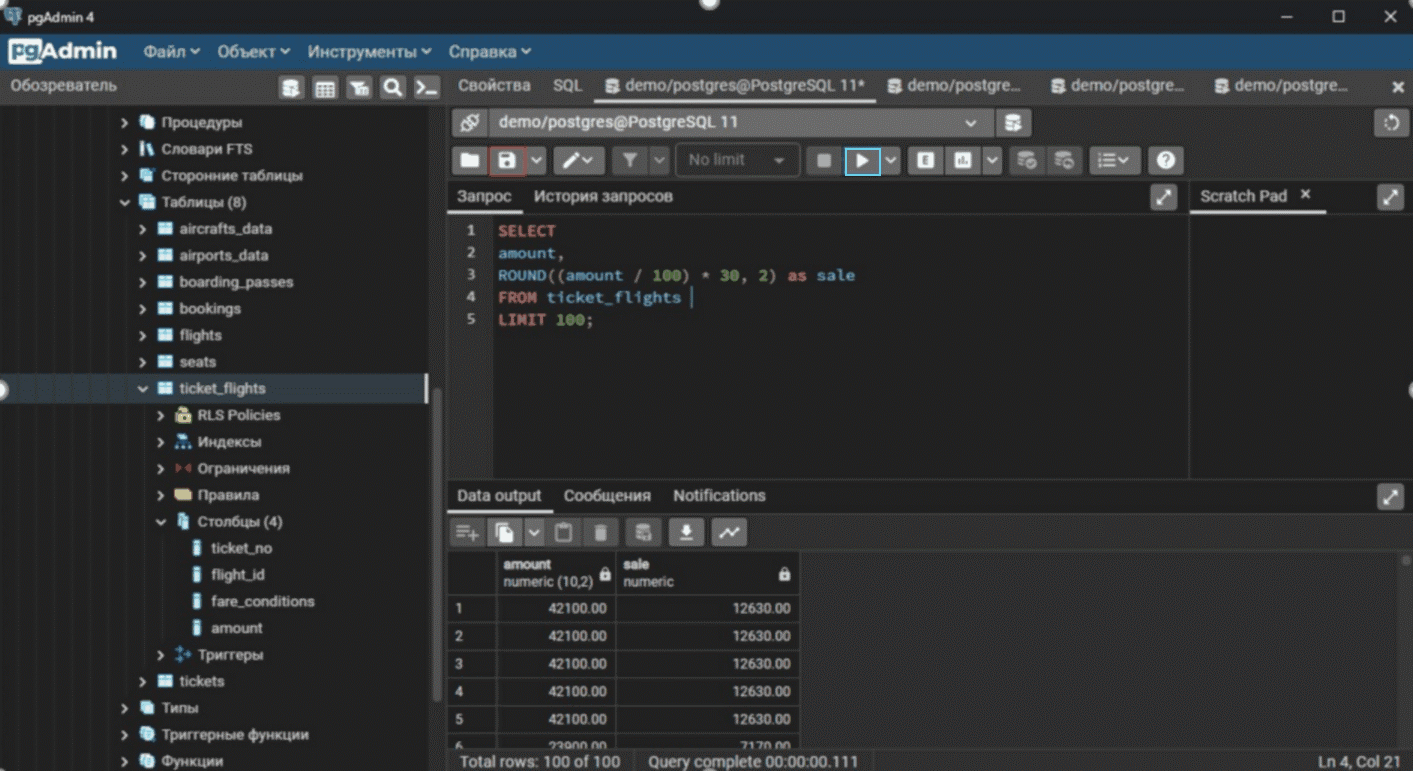
****

Рис. 11. Запрос №1

**Задание для практики**

Добавить ещё один столбец с итоговой ценой билета со скидкой 30%.

**Предложение WHERE**

При выборке записей из таблицы практически всегда требуется задавать определённые условия, которые определяют, какие записи нам нужны, а какие нет. И именно эти условия можно задавать с помощью предложения WHERE. Для примера рассмотрим туже таблицу с билетами и напишем запрос с выборкой билетов по цене в диапазоне от 3000 до 9000 и классу билета - Economy.

**Запрос 2:**

SELECT \*

FROM ticket\_flights

WHERE BETWEEN ‘3000’ AND ‘9000’ AND fare\_conditions = ‘Economy’;

-- Важно! Текстовые строки в одинарных кавычках

**Последовательность выполнения запроса такова:**

* С помощью предложения SELECT отбирается информация, которая будет отображена в итоговой таблице. В этом запросе результат - все поля исходной таблицы (для этого после SELECT нужно поставить символ «звездочка» - «\*»).
* В предложении WHERE определяется условие, которое вычисляется для каждой строки и если его значение TRUE то запись выбирается. В данном случае это билеты, стоимость которых в диапазоне от 3000 до 9000 в эконом классе. С помощью логического оператора AND указывается какие ещё параметры должны быть учтены при выборке.

Как это выглядит в самой среде можно посмотреть рисунке 12.

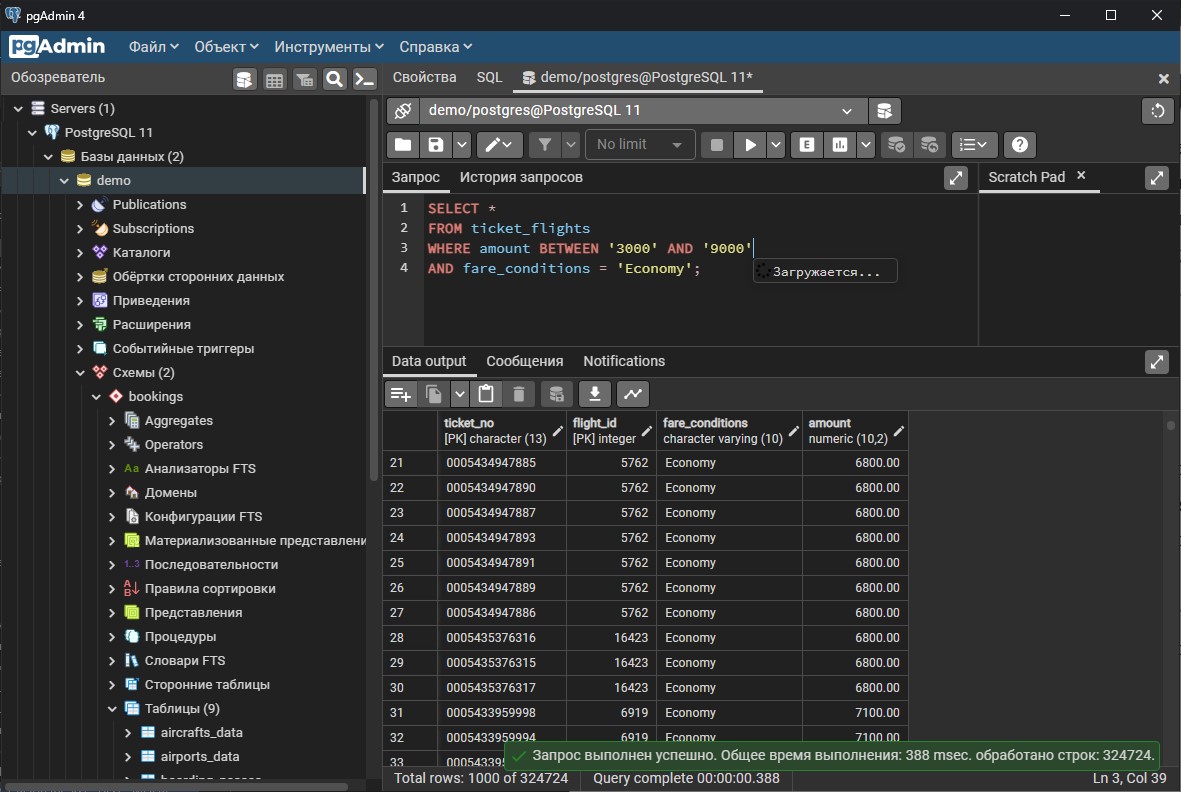


Рис. 12. Запрос №2

**Предложение ORDER BY**

При выполнении SELECT запроса, строки возвращаются в произвольном порядке. Для упорядочивания записей используется предложение ORDER BY. Для примера возьмем предыдущий запрос и модифицируем его, чтобы упорядочить записи по убыванию цены билета.

**Запрос 3:**

SELECT \*

FROM ticket\_flights

WHERE amount < 12000 AND fare\_conditions = ‘Economy’

ORDER BY amount DESC; -- «DESC» указывает на сортировку по убыванию

**Последовательность выполнения запроса такова:**

* В этом запросе выбираются все поля таблицы ticket\_flights (для этого после ключевого слова SELECT нужно поставить символ «звездочка» - «\*»).
* В предложении WHERE определяется условие, которое вычисляется для каждой строки и если его значение TRUE то запись выбирается. В данном случае это билеты со стоимостью не выше 12 000, эконом класса.
* Далее в предложении ORDER BY выбирается по какому полю будет идти сортировка: в нашем случае указываем стоимость, и ставим ключевое слово DESC, означающее сортировку по убыванию стоимости билета.

Как выглядит этот оператор в среде pgAdmin можно посмотреть рисунке 13.

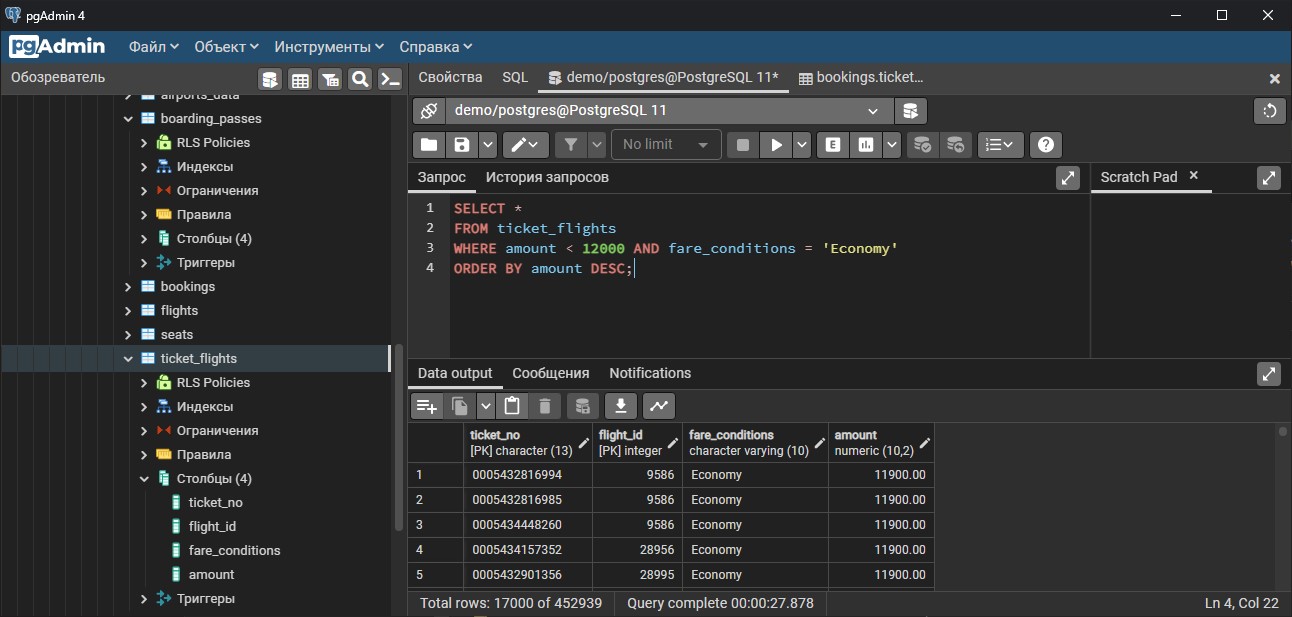


Рис. 13. Запрос №3

**Предложение GROUP BY**

В результате выполнения предложения GROUP BY формируется сгруппированная таблица, в которой записи таблицы, расставлены в минимальное число групп, таких, что во всех записях одной группы значения полей, указанных в списке имен полей предложения GROUP BY (полей группировки), одинаковы. Группировку чаще всего используют вместе с агрегатными функциями, такими как:

* COUNT(\*) - подсчёт кол-ва строк,
* SUM(поле) - сумма значений поля,
* MAX(поле) - максимальное значение поля,
* MIN(поле) - минимальное значение поля,
* AVG(поле) - среднее значение поля.

Пример такого запроса может быть таким: «нужно подсчитать количество мест по классу обслуживания (Economy, Comfort, Business)».

Функция COUNT принимает один из нескольких параметров:

* «\*» – означает то, что функция COUNT возвращает количество всех записей в таблице;
* «column\_name» – функция COUNT возвращает количество записей с непустыми значениями конкретного поля (только NOT NULL);
* «DISTINCT column\_name» – функция COUNT возвращает количество уникальных записей конкретного поля (только NOT NULL).

**Запрос 4:**

SELECT fare\_conditions, COUNT(\*) as seats

FROM seats

GROUP BY fare\_conditions;

**Последовательность выполнения запроса такова:**

* Предложение SELECT определяет информацию, которая будет содержаться в итоговой таблице. В этом запросе «fare\_conditions» - поле группировки и следом идет количество мест, подсчитанное с помощью функции COUNT (второму столбцу присваивается название «seats» - не путайте с названием таблицы)
* Далее в предложении GROUP BY пишем название поля, по которому будет проведена группировка

Как выглядит в среде pgAdmin можно посмотреть рисунке 14.

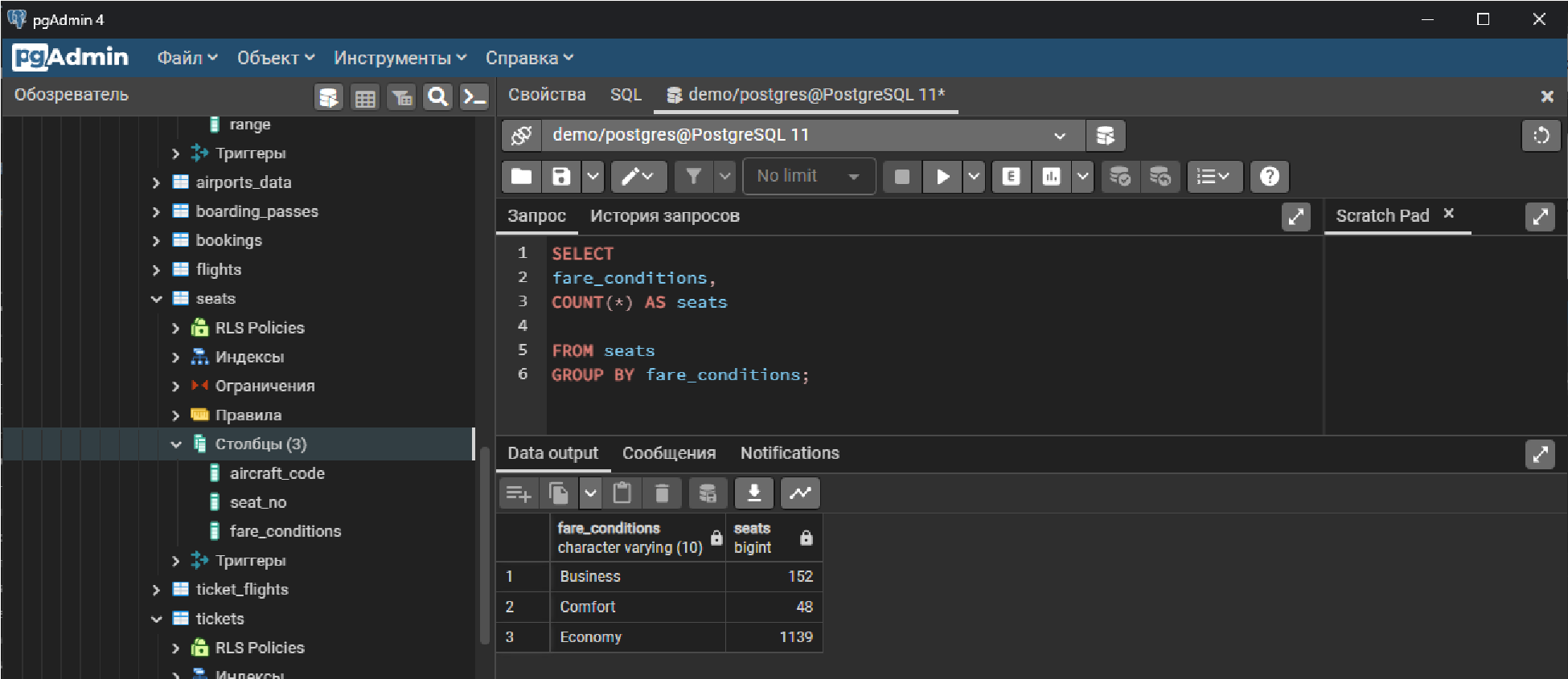


Рис. 14. Запрос №4

**Задание для практики**

Придумать свой запрос с предложением GROUP BY по имеющимся в базе таблицам.

## Практическая часть

### Задание для выполнения лабораторной работы

Первое практическое задание заключается в знакомстве со средой pgAdmin и написании SQL-запросов с использованием оператора SELECT.

Для учебной базы данных должны быть составлены 4 произвольных SELECT-запроса, демонстрирующие полученные знания. Запросы должны охватывать проработанные темы. После составления запросов следует убедиться в правильности их работы при помощи более простых запросов.

Дополнительные вопросы могут заключаться в построении более сложных запросов или объяснении работы подготовленных заданий.

### Инструкция по добавлению учебной базы данных

Скрипт для создания и заполнения учебной базы данных и её описание: [https://postgrespro.ru/education/demodb](https://postgrespro.ru/education/demodb%20) .

Для данной задачи нам понадобиться psql, чтобы выполнить .sql файл. Программа psql – это терминальный клиент для работы с PostgreSQL. Она позволяет интерактивно вводить запросы, передавать их в PostgreSQL и видеть результаты. Также запросы могут быть получены из файла или из аргументов командной строки. Кроме того, psql предоставляет пользователю различные возможности, подобные тем, что имеются у командных оболочек, для облегчения написания скриптов и автоматизации широкого спектра задач.

Как только вы откроете консоль вам сразу же предложат ввести ip сервера, названия базы данных, порт, имя пользователя. В процессе установки PostgreSQL был создан сервер и пустая БД со значениями по умолчанию, они и показаны в квадратных скобках, поэтому можно просто нажать несколько раз на Enter до момента запроса пароля (как на рисунке 15) и ввести тот пароль, который вы вводили при установке СУБД. (символы при вводе пароля могут не отображаться).

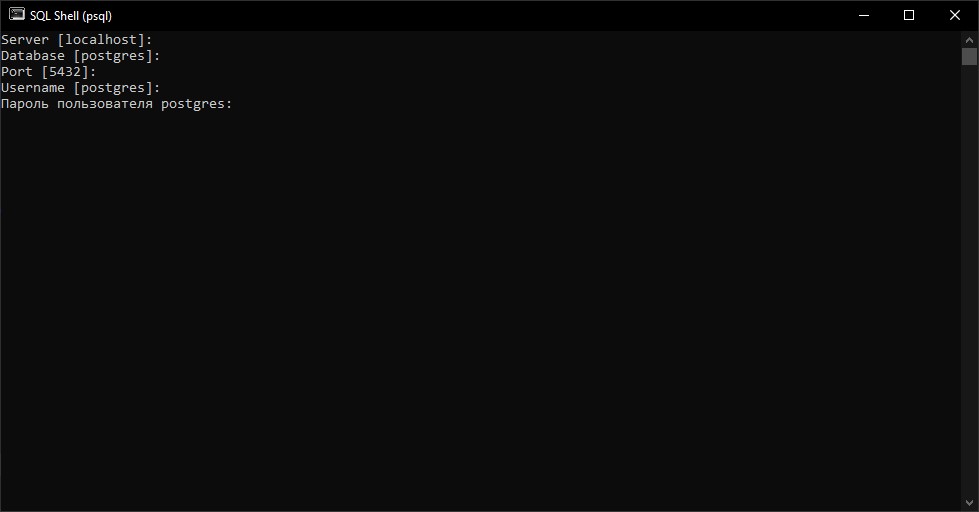


Рис. 15. Ввод данных

Если у вас наблюдаются ошибки с кодировкой, то это можно исправить с помощью команды: «chcp 1251» (см. рисунок 16).

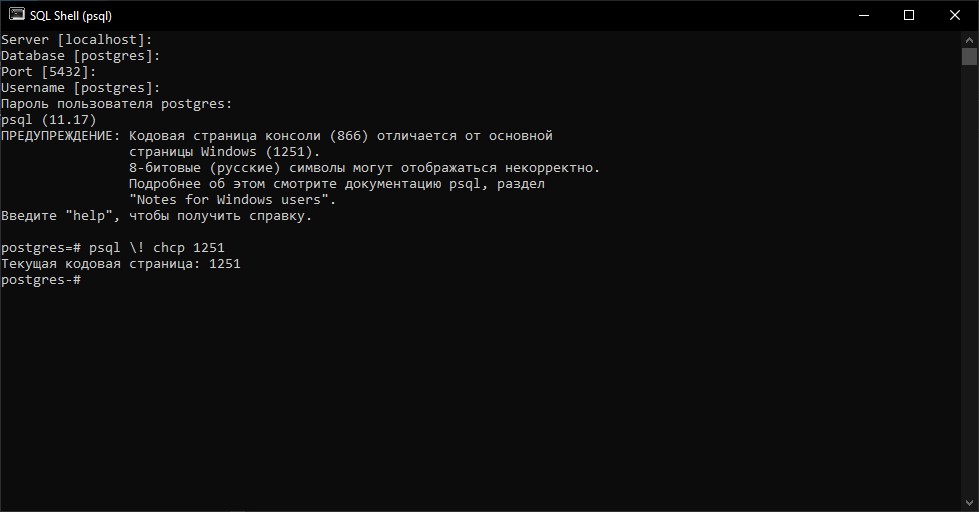


Рис. 16. Изменение кодировки

Далее приступим к добавлению БД с помощью команды \i (include)

\i ‘C:\\DataLab\\demo-small.sql’

Здесь нужно указать команду и путь к выполняемому файлу в одинарных кавычках как на рисунке 17. Как только вы нажмете клавишу «Enter» начнется выполнение файла, которое займет некоторое время и, после завершения работы, учебная БД будет добавлена на ваш сервер.

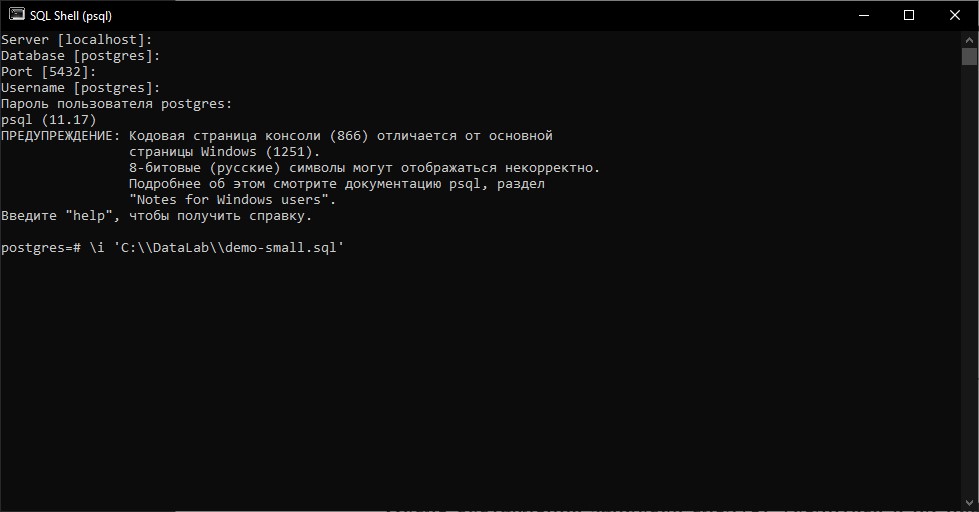


Рис. 17. Выполнение sql-файла

## Вопросы для самостоятельной проработки

1. Объяснить, как работают написанные запросы.
2. Рассказать про операцию соединения (JOIN) и различные её разновидности.
3. Рассказать про агрегатные функции, предложения GROUP BY и HAVING.
4. Как выбрать только уникальные значения какого-либо поля?
5. Как осуществить сортировку по возрастанию/убыванию по значению какого-либо поля?
6. Как агрегатные функции ведут себя по отношению к неопределённым значениям?
7. Чем отличаются UNION и UNION ALL?
8. Чем отличаются COUNT(\*) и COUNT(field)?
9. Как подсчитать количество уникальных значений поля?
10. Как можно осуществить проверку на неопределенное значение?
11. Рассказать про предикат LIKE.
12. Как можно выбрать только определенное количество строк?
13. Исправить неверно работающий запрос (запросы).
14. Упростить один или несколько запросов.
15. Округлить результирующее значение до 3 знаков после точки.
16. Округлить вещественное число до целого.
17. Переписать запрос, не используя функцию MAX (MIN).
18. Изменить формат вывода данных (например, формат даты и времени).
19. Написать или модифицировать запрос по сформулированному заданию.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Фомин М.М. Реляционные базы данных. Учебное пособие для бакалавров <https://e-learning.bmstu.ru/iu6/mod/resource/view.php?id=6634>
2. Карпова И.П. Базы данных. Учебное пособие. – Московский государственный институт электроники и математики (Технический университет): учебное пособие / И.П. Карпова; – М., 2009. – 140-141 с, 102 c.
3. PostgreSQL. Основы языка SQL: учеб. пособие / Е. П. Моргунов; под ред. Е. В. Рогова, П. В. Лузанова. - СПб.: БХВ-Петербург, 2018. - 65-68 c, 68-72 с.
4. Стоунз Р., Мэтью Н. PostgreSQL. Основы. – Пер. с англ. – СПб: Символ\_Плюс, 2002. – 640 с., ил. ISBN 5\_93286\_043\_X
5. Болье, А. Изучаем SQL. Генерация, выборка и обработка данных, 3-е изд./ пер. с англ. И.В. Красикова. — Киев.: “Диалектика”,2021. — 402 с.: ил. ISBN 978-1-492-05761-1
6. Синтаксис SQL: <https://postgrespro.ru/docs/postgrespro/11/sql-syntax>
7. SELECT-запросы: <https://postgrespro.ru/docs/postgrespro/11/queries>
8. Функции и операторы: https://postgrespro.ru/docs/postgrespro/11/functions
9. Полное описание синтаксиса встретившихся команд: [https://postgrespro.ru/docs/postgrespro/11/sql-commands](https://postgrespro.ru/docs/postgrespro/11/sql-commands%20)
10. Оператор With и рекурсивные запросы: <https://postgrespro.ru/docs/postgrespro/11/queries-with>
11. Работа в среде pgAdmin: https://metanit.com/sql/postgresql/1.1.php