ПРАКТИЧЕСКАЯ РАБОТА 1. ЧАСТЬ 2

Практическая работа №1 выполняется индивидуально по методическим указаниям и включает в себя несколько заданий. Практическая работа №1 состоит из 2х частей. В данном документе представлено задание для выполнения второй части практической работы №1.

По практической работе №1 формируется итоговый отчет, содержащий результаты выполнения 2х частей работы. Итоговый отчет должен содержать:

- Титульный лист
- Цель работы.
- Задачи, решаемые при выполнении работы.
- Исходные данные.
- Выполнение работы: Краткое описание процесса выполнения всех задач по шагам (при наличии нескольких шагов) со скриншотами.
- Выводы и анализ результатов работы. Обобщение результатов выполнения всех задач работы: чего должны были достичь, чего фактически достигли и каким образом, с какими трудностями столкнулись, какие проблемы на каких этапах выполнения возникли и как именно были решены.

Задание 1. Создание таблицы в графической среде pgAdmin

1. Раскройте в обозревателе решений базу данных «HR» до узла «Таблицы» и запустите окно создания новой таблицы (рисунок 1.1):

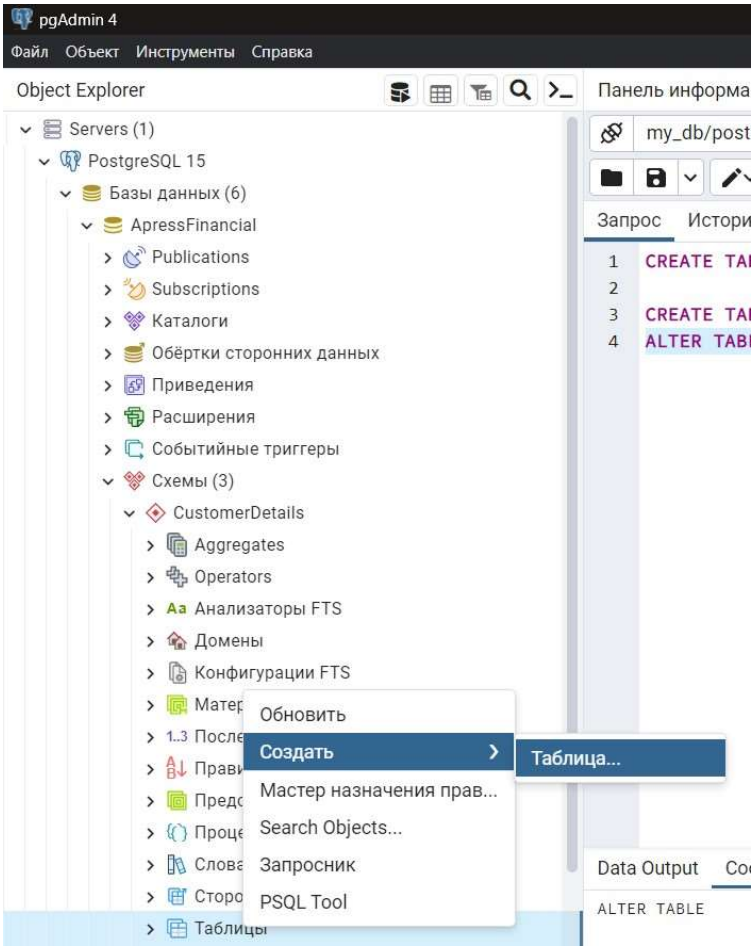

Рисунок 1.1 – Обозреватель объектов. Создание таблицы.

2. На вкладке General укажите:
    - имя таблицы: EMPLOYEES;
    - схема – EmployeesDepartments (была создана ранее в лабораторной)
    - поле Комментарий – дайте некое описание таблицы

3. Заполните вкладку «Столбцы» следующим образом (рисунок 1.2):

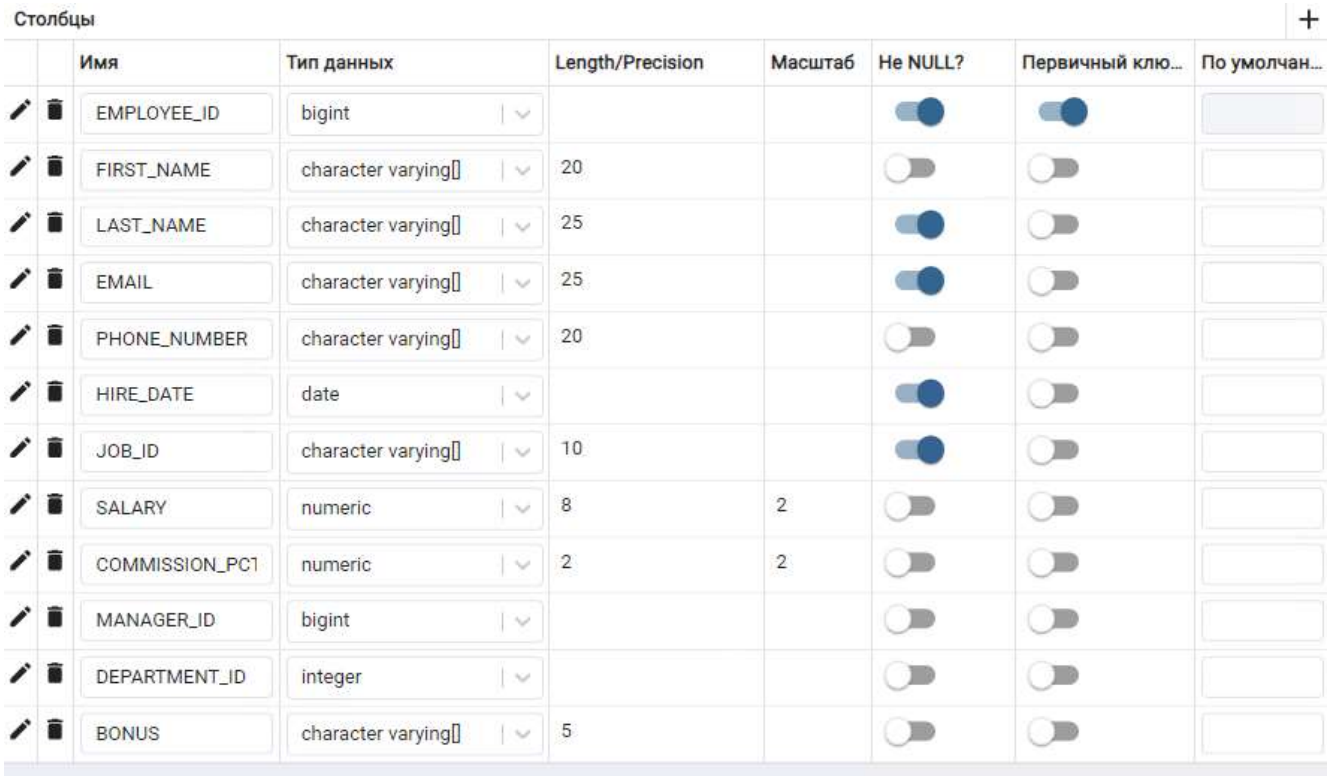

Рисунок 1.2 – Редактирование столбцов таблицы.

NB! Будьте внимательны при указании флагов NOT NULL и Первичного ключа

4. Настройте автоинкремент поля EMPLOYEE_ID (рисунок 1.3), нажав на значок редактирования поля.

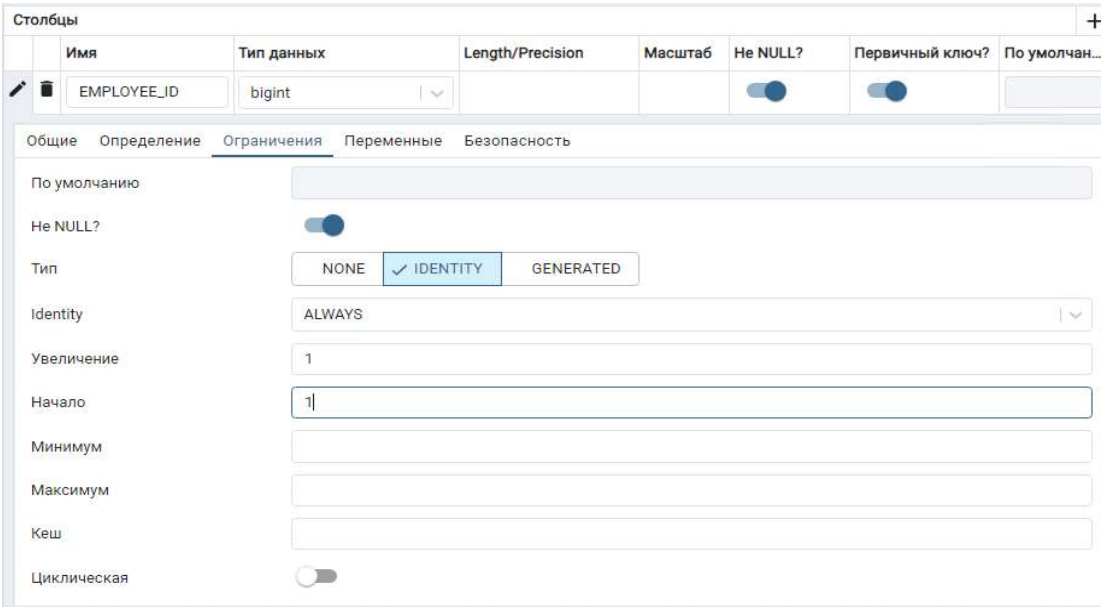

Рисунок 1.3 – Установка автоинкремента для столбца.

5. Для столбца HIRE_DATE поставьте значение по умолчанию current_date (рисунок 1.4).

Имя	Тип данных	Length/Precision	Масштаб	Не NULL?	Первичный ключ?	По умолчанию
PHONE_NUMBER	character varying[]	20				
HIRE_DATE	date					current_date
JOB_ID	character varying[]	10				

Рисунок 1.4 – Установка значения по умолчанию для столбца HIRE_DATE.

6. Посмотрите вкладку SQL.
7. Можно сохранять таблицу.

Задание 2. Создание таблицы в Query Editor

1. Откройте запросник базы HR
2. В окне запроса введите следующий код (рисунок 2.1):

```
CREATE TABLE "EmployeesDepartments"."DEPARTMENTS" (
    DEPARTMENT_ID integer GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1) PRIMARY KEY NOT NULL,
    DEPARTMENT_NAME character varying(30)[] NOT NULL,
    MANAGER_ID bigint NULL,
    LOCATION_ID integer NULL
    );
```

Рисунок 2.1 – DDL оператор создания таблицы CREATE TABLE.

3. Может появиться сообщение об ошибке (рисунок 2.2). Это может произойти по целому ряду причин: начиная с ошибки при печати кода и заканчивая отсутствием разрешения на создание таблиц. Генерируемое сообщение об ошибке, как правило, понятно без пояснения. Например, нижеследующее сообщение информирует, что при наборе сделана ошибка в строке 5:

```
ERROR:  ошибка синтаксиса (примерное положение: "in")
LINE 5:  transactiontype in NOT NULL,
```

Рисунок 2.2 – Сообщение об ошибке.

4. Перейдите в Object Explorer (Обозреватель объектов), обновите узел Tables (Таблицы) и проверьте: должна появиться таблица DEPARTMENTS.

5. Создайте еще одну таблицу (рисунок 2.3):

```
CREATE TABLE  "EmployeesDepartments"."LOCATIONS"
   (LOCATION_ID smallint GENERATED ALWAYS AS IDENTITY NOT NULL,
    STREET_ADDRESS character varying(40),
    POSTAL_CODE character varying(12),
    CITY character varying(30) ,
    COUNTRY_ID CHAR(2)
    );
```

Рисунок 2.3 – DDL оператор создания таблицы CREATE TABLE.

Задание 3. Изменение таблицы

Необходимо добавить столбец. Если таблица была создана с ошибками, то можно таблицу удалить с помощью инструкции DROP TABLE и создать заново. Но если в таблицу уже поместили какие-либо данные, то такое решение не подходит. Есть альтернативный способ: инструкция ALTER TABLE, которая допускает ограниченные(!) преобразования формата и при этом сохраняет ее содержимое.

1. В окне запроса введите следующий код (рисунок 3.1):

```
ALTER TABLE "EmployeesDepartments"."LOCATIONS"
ADD STATE_PROVINCE character varying(25) NULL;
```

Рисунок 3.1 – DDL оператор изменения таблицы ALTER TABLE.

Примечание. Так как мы не хотим, чтобы существующие данные принимали значения по умолчанию, то при добавлении столбца следует допустить значения NULL. Ваша таблица пока не содержит никаких данных. Но если бы данные были, то после добавления столбца можно было бы внести в них изменения, а затем отменить разрешение на значение NULL.

2. Изменим столбец CITY, он обязательно должен содержать значения. Для этого введите и выполните следующий код (рисунок 3.2):

```
ALTER TABLE "EmployeesDepartments"."LOCATIONS"
ALTER COLUMN CITY SET NOT NULL;
```

Рисунок 3.2 – DDL оператор изменения столбца таблицы
ALTER TABLE ->ALTER COLUMN.

3. Добавим первичный ключ (рисунок 3.3):

```
ALTER TABLE "EmployeesDepartments"."LOCATIONS"
ADD CONSTRAINT  "PK_LokationId" PRIMARY KEY (LOCATION_ID);
```

Рисунок 3.3 – DDL оператор изменения таблицы ALTER TABLE, добавление ограничения целостности ADD CONSTRAINT

4. Теперь таблица LOCATIONS стала правильной.

5. Необходимо к трем созданным таблицам добавить еще восемь. Для этого в окне запроса введите и выполните следующий код из файла script1_create.sql, который находится в разделе «Литература и ПО». Рекомендуется весь код выполнять последовательно, чтобы избежать ошибки.

Задание 4. Создание отношения графическим интерфейсом pgAdmin

1. Нажмите ПКМ на таблицу EmployeesDepartments.LOCATIONS и выберите меню создания внешнего ключа (рисунок 4.1):

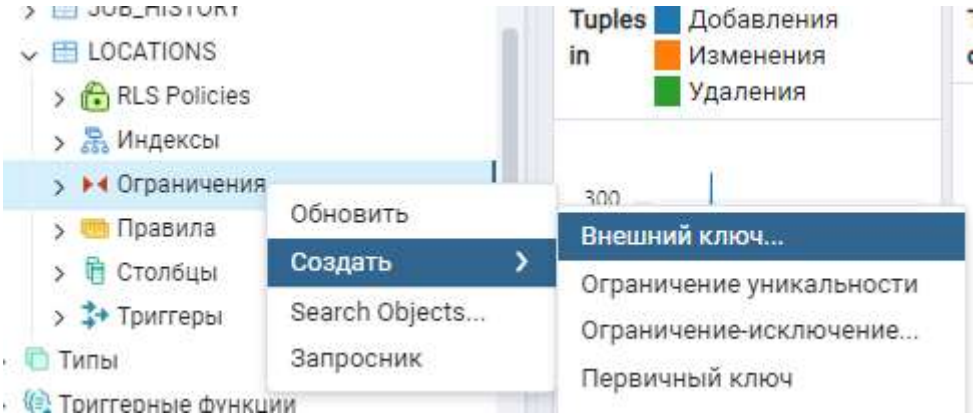

Рисунок 4.1 – Обозреватель объектов. Создание внешнего ключа.

2. Введите название отношения – "LOC_C_ID_FK" (ограничение внешнего ключа, рисунок 4.2).

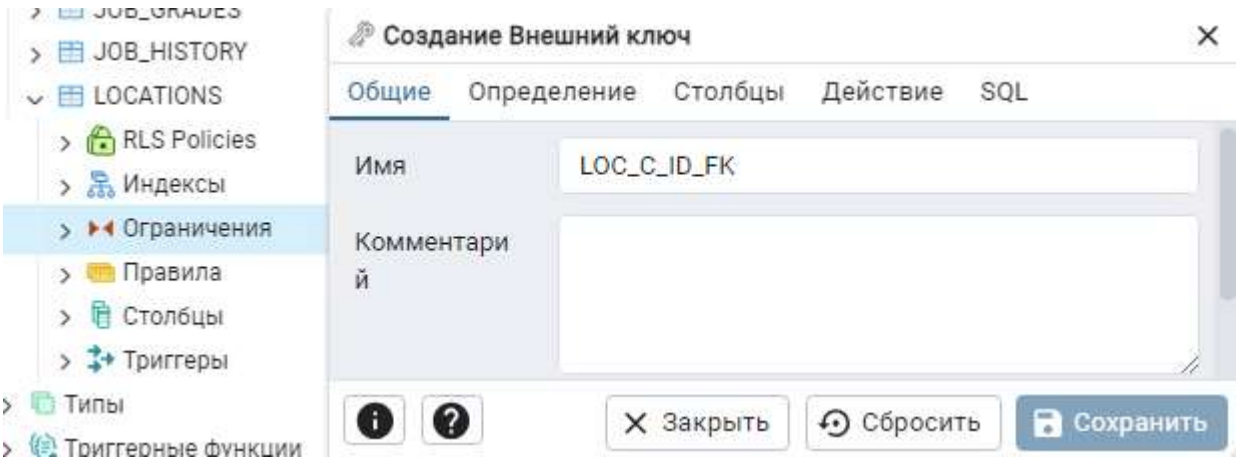

Рисунок 4.2 – Создание внешнего ключа LOC_C_ID_FK.

3. В разделе «Столбцы» сделайте связь от локального столбца "COUNTRY_ID" текущей таблицы до столбца "COUNTRY_ID" таблицы "COUNTRIES" в схеме "Countries" (рисунок 4.3):

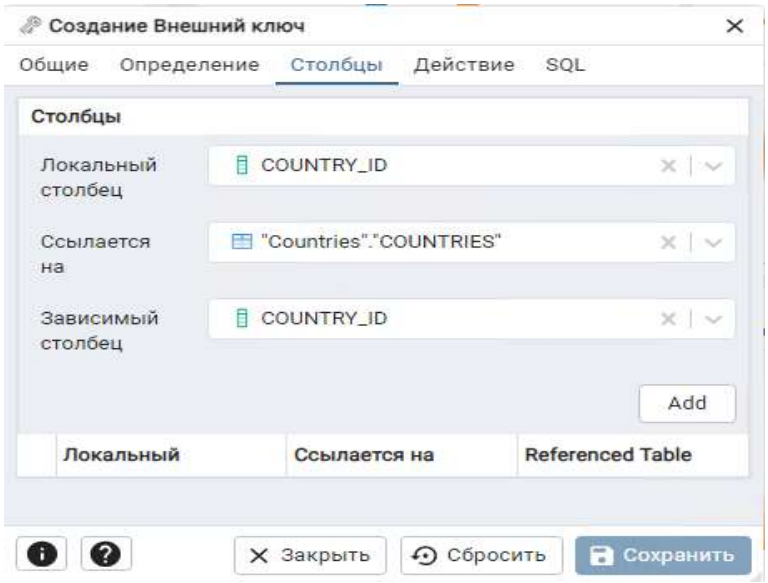

Рисунок 4.3 –Установка связи между таблицами.

4. Нажмите кнопку «Add».
5. Посмотрите вкладку SQL и нажмите «Сохранить».
6. Посмотрите ограничения таблицы LOCATIONS в обозревателе объектов.
7. Создайте отношение с помощью DDL оператора ALTER TABLE, для этого выполните следующий код (рисунок 4.4):

```
ALTER TABLE  "Countries"."COUNTRIES"
ADD CONSTRAINT "COUNTR_REG_ID_FK1" FOREIGN KEY ("REGION_ID")
REFERENCES  "Countries"."REGIONS" ("REGION_ID");
```

Рисунок 4.4 – DDL оператор изменения таблицы ALTER TABLE, добавление ограничения целостности ADD CONSTRAINT

Задание 5. Создание индекса с помощью графического интерфейса pgAdmin

1. Нажмите ПКМ на таблицу EmployeesDepartments.LOCATIONS, выберите ПКП раздел меню для создания индекса (рисунок 5.1):

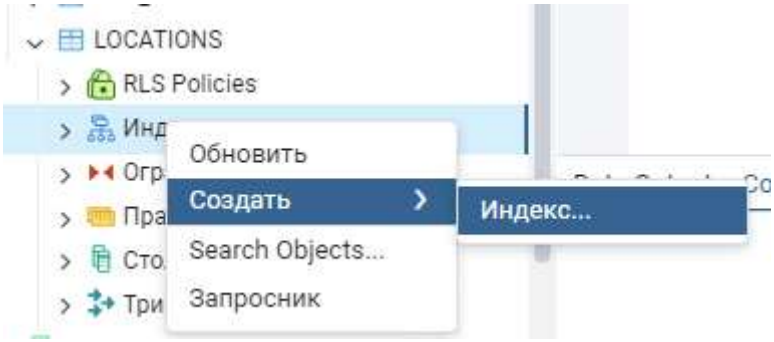

Рисунок 5.1 – Обозреватель объектов. Создание индекса.

2. На вкладке General введите название индекса - LOC_CITY_IX (рисунок 5.2)

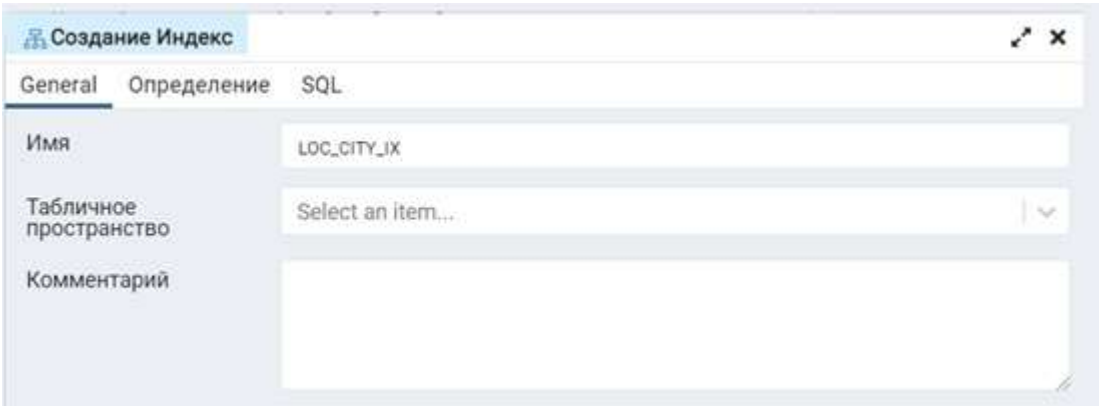

Рисунок 5.2 –Создание индекса LOC_CITY_IX.

Такое имя, которое содержит постфикс (IX), имя таблицы (LOCATIONS) и имя столбца (CITY), даже не нуждается в дополнительном описании, которое можно было бы поместить в поле Комментарий.

3. В разделе Определение оставьте тип индекса btree (рисунок 5.3):

Рисунок 5.3 –Установка свойств индекса LOC_CITY_IX.

А в разделе столбцы укажите построение по полю CITY по возрастанию (ASC) (рисунок 5.4):

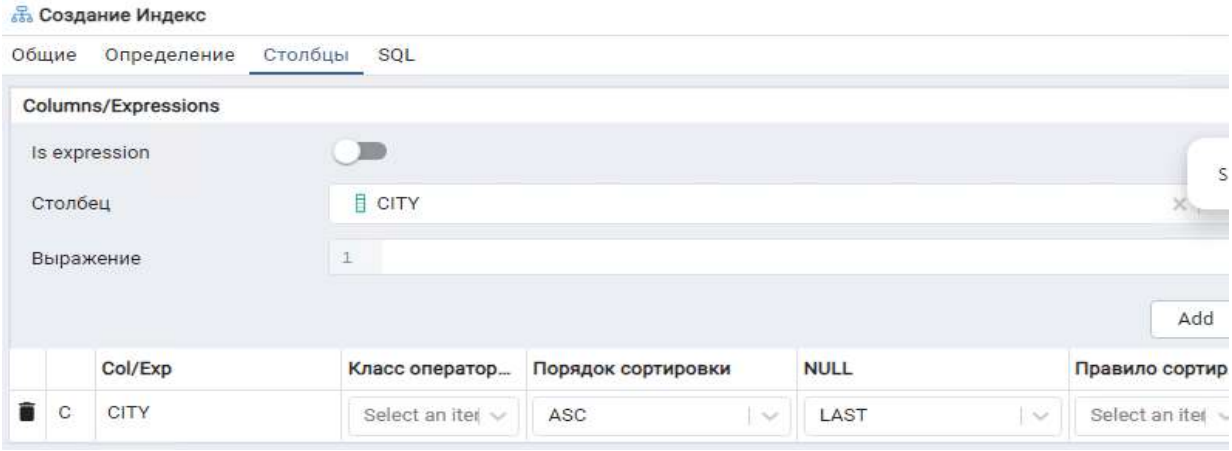

Рисунок 5.4 –Установка свойств индекса LOC_CITY_IX.

Порядок сортировки – по возрастанию (ASC) – важное свойство для индекса столбцов, которые будут указываться в предложении ORDER BY запроса с различными порядками сортировки. Если порядок сортировки в индексе совпадает с порядком сортировки в ORDER BY запроса, то производительность выполнения запроса повышается.

4. Значения параметров об уникальности (позволяет проверять уникальность значений столбца перед обновлением или вставкой.), кластеризации и параллельном создании оставьте без изменения в отключенном положении.
5. Посмотрите на вкладку SQL, сохраните индекс.

NB! Для получения дополнительной информации об индексе выделите его, ПКМ по индексу и выберите пункт Properties (Свойства). Откроется Index Properties (Свойства индекса), которое представляет не только графическую версию индекса, но и список многих его параметров. Подробнее о параметрах индекса можно почитать в справке https://postgrespro.ru/docs/postgrespro/13/sql-createindex

Задание 6. Построение диаграмм базы данных

Диаграмма БД позволяет немедленно представить общую картину БД. Диаграммы – это идеальный метод документирования БД! Диаграммы могут быть распечатаны для обсуждений, отчетов, анализа, дискуссий по поводу дальнейшей разработки и.т.п. Диаграмма БД по умолчанию должна включать в себя все таблицы и все отношения, хранящиеся в этой БД.

1. Для создания диаграммы нажмите ПКМ на узел базы данных и выберите параметр «ERD For Database» (рисунок 6.1)

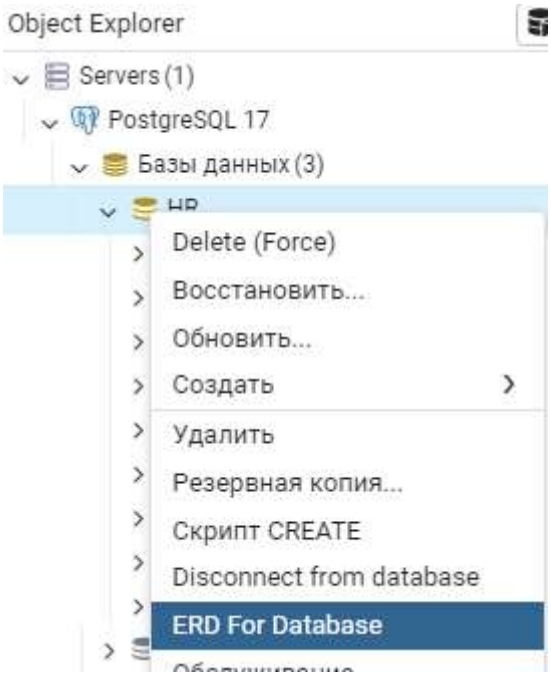

Рисунок 6.1 – Обозреватель объектов. Построение диаграммы «сущность-связь»

2. Можно перемещать элементы и отношения курсором, чтобы сделать диаграмму более читаемой. Скорректируйте вид своей диаграммы на следующий (рисунок 6.2):

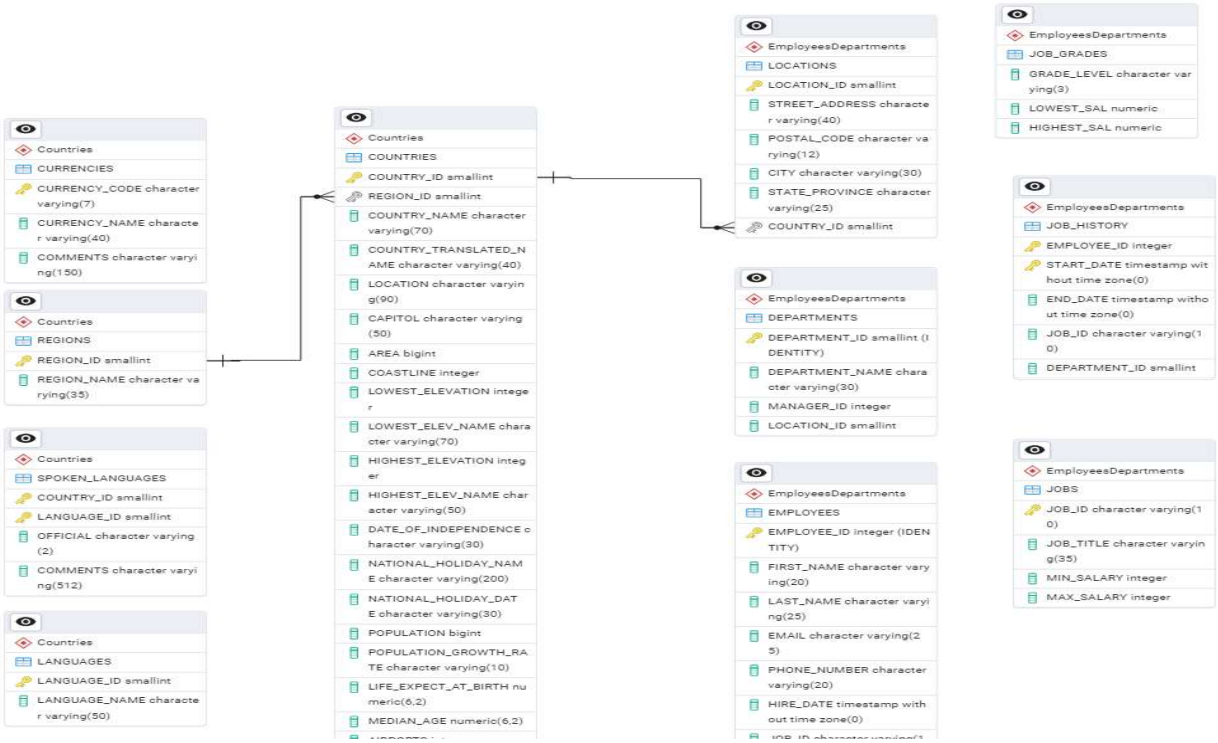

Рисунок 6.2 – Диаграмма «сущность-связь» базы данных HR

Как вы можете заметить, ещё не все требуемые связи на диаграмме отражены.

3. Экспортируйте снимок диаграммы путем нажатия на следующую кнопку (рисунок 6.3):

Рисунок 6.3 – Экспорт диаграммы

4. Самостоятельно посмотрите другие возможности этой панели.
5. Сохраните проект диаграммы (рисунок 6.4).

Рисунок 6.4 – Сохранение диаграммы

6. Закройте вкладку с диаграммой.

NB! Правкой БД через диаграмму следует пользоваться с осторожностью. Если инструмент построения диаграмм БД применяется для разработки в многопользовательской среде, помните, что изменения сохраняются у того, кто сохраняет их последним!

Задание 7. Добавление данных в таблицу

1. Раскройте меню для генерирования скрипта по вставке данных в таблицу Countries.CURRENCIES (рисунок 7.1):

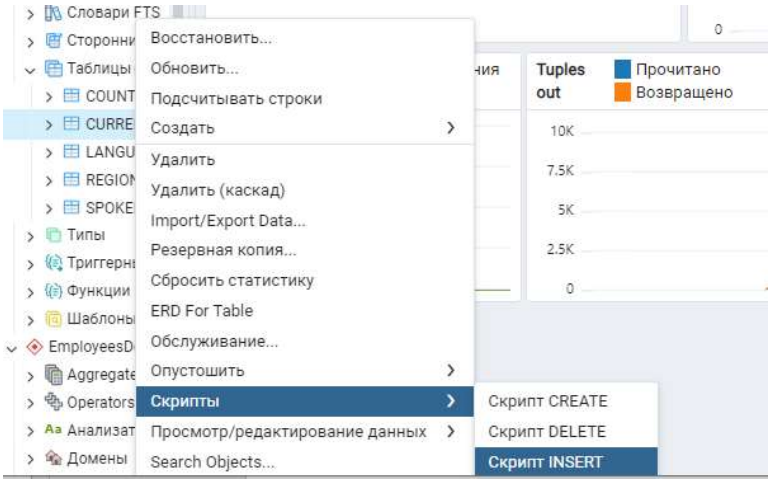

Рисунок 7.1 – Обозреватель объектов. Добавление данных в таблицу.

Отредактируйте код следующим образом (рисунок 7.2):

```
INSERT INTO "Countries"."CURRENCIES"("COMMENTS","CURRENCY_NAME", "CURRENCY_CODE")
VALUES('EuroZone','Euro','EUR');
```

Рисунок 7.2 – Добавление данных в таблицу, DML оператор INSERT.

NB! Обратите внимание на экранирование значений всех атрибутов в кавычки, так как все они имеют символьный тип.

2. Проверьте добавленные данные (рисунок 7.3):

```
select *
from "Countries"."CURRENCIES";
```

Рисунок 7.3 – Проверка, что данные добавлены.

3. В Object Explorer раскройте узел БД HR до таблицы "EmployeesDepartments"."EMPLOYEES". Перед добавлением значений, вспомните дополнительный функционал для полей, который определялся в более ранних лабораторных, раскрыв свойства этой таблицы (рисунок 7.4)::

Рисунок 7.4 – Обозреватель объектов. Свойства столбцов таблицы EMPLOYEES

Таблица EMPLOYEES содержит столбец идентификации, т.е.специальный столбец, который автоматически генерируется из неявной последовательности. Если команда INSERT выполняется для таблицы со столбцом идентификации, и значение для этого столбца не указано явно, вставляется значение, сгенерированное неявной последовательностью.

Свойства ALWAYS и BY DEFAULT в определении столбца устанавливают, как явно указанные пользователем значения обрабатываются командами INSERT и UPDATE. В команде INSERT, в случае выбора ALWAYS, пользовательское значение принимается, только если в этой команде указано OVERRIDING SYSTEM VALUE. С предложением BY DEFAULT пользовательскому значению отдаётся предпочтение. Таким образом, поведение с BY DEFAULT похоже на поведение с использованием значений по умолчанию, когда такие значения могут быть переопределены пользовательскими. Поэтому изменим свойство для генерации последовательности атрибута EMPLOYEE_ID с ALWAYS на BY DEFAULT, для добавления подготовленных тестовых данных (рисунок 7.5):

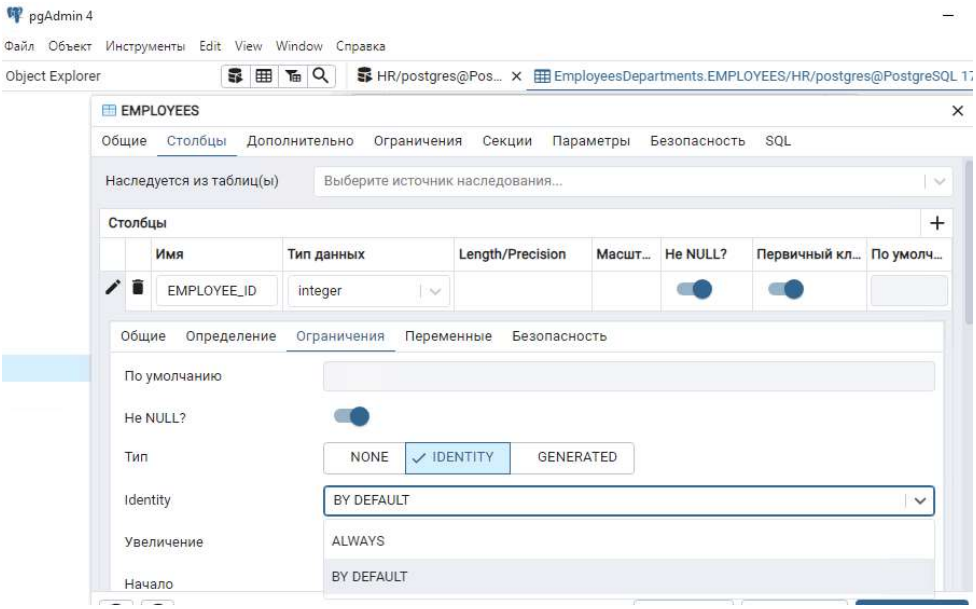

Рисунок 7.5 – Таблица EMPLOYEES, последовательность.

4. Добавьте запись в таблицу "EmployeesDepartments"."EMPLOYEES" с помощью графических средств pgAdmin (рисунок 7.6):

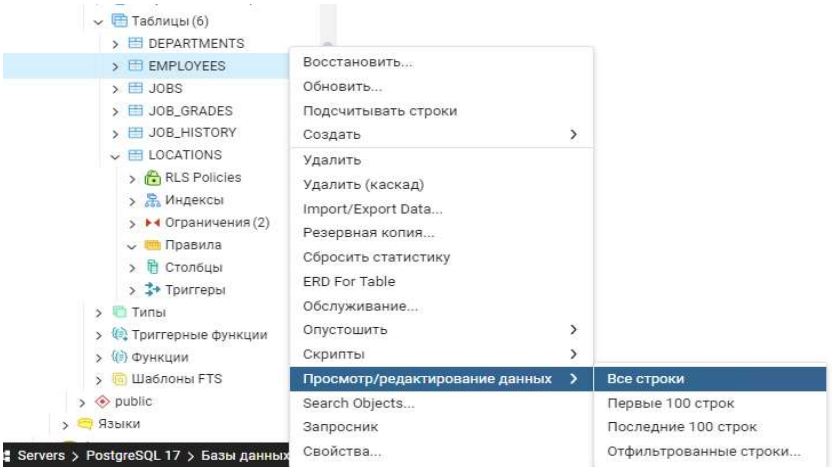

Рисунок 7.6 – Обозреватель объектов. Просмотр данных в таблице

Справа от окна Object Explorer должны появиться все строки данных таблицы. Но так как таблица не содержит никаких данных, то вы увидите пустую сетку для ввода первой записи. Добавьте строку нажатием на кнопку «Add Row» (рисунок 7.7):

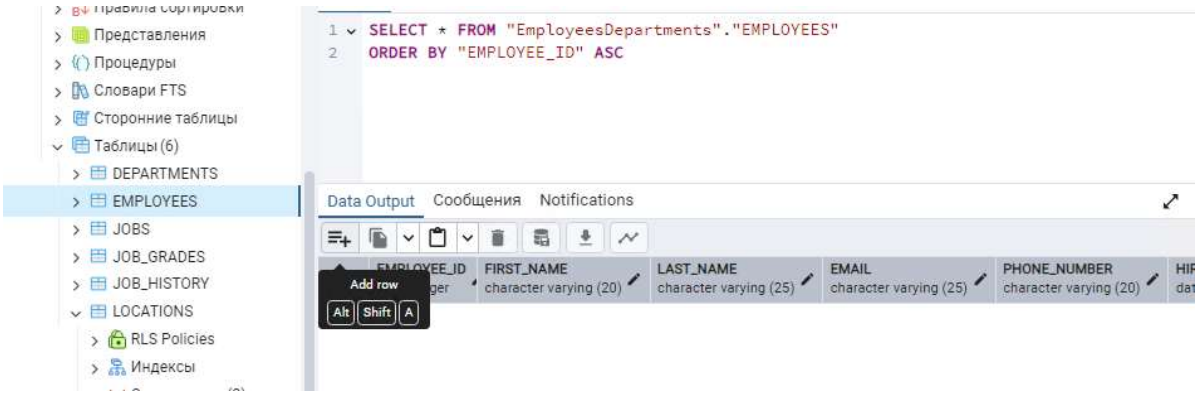

Рисунок 7.7 – Просмотр данных в таблице EMPLOYEES

Для попытки сохранения записи нажмите на кнопку «Save Data Changes» (рисунок 7.8):

Рисунок 7.8 – Сохранение данных в таблице EMPLOYEES

5. При ручном добавлении записей ориентируйтесь на всплывающие подсказки и ограничения (рисунок 7.9 и рисунок 7.10):

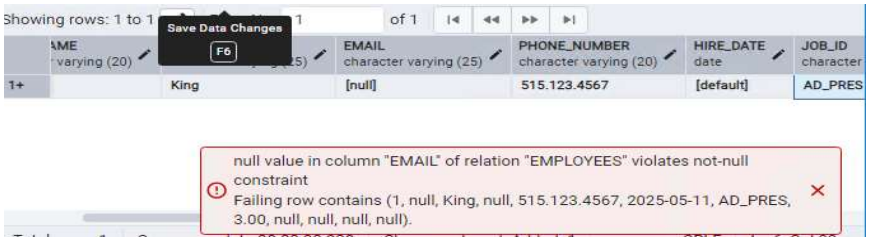

Рисунок 7.9 – Сохранение данных в таблице EMPLOYEES, получение ошибки

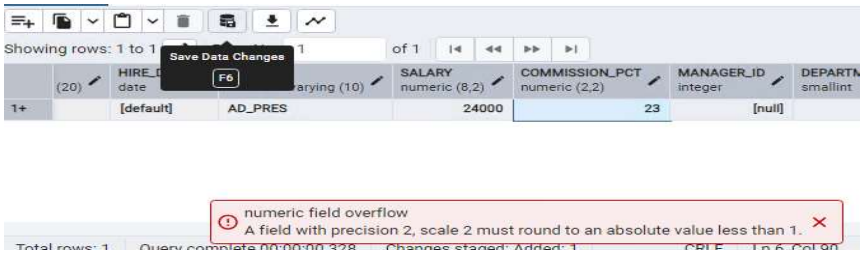

Рисунок 7.10 – Сохранение данных в таблице EMPLOYEES, получение ошибки

6. Введите строку правильно с учетом всех подсказок Пример строки (рисунок 7.11):

	EMPLOYEE_ID [PK] integer	FIRST_NAME character varying (20)	LAST_NAME character varying (25)	EMAIL character varying (25)	PHONE_NUMBER character varying (20)	HIRE_DATE date
1	1	Steven	King	SKING	515.123.4567	2025-05-11

JOB_ID character varying (10)	SALARY numeric (8,2)	COMMISSION_PCT numeric (2,2)	MANAGER_ID integer	DEPARTMENT_ID smallint	BONUS character varying (5)
AD_PRES	24000.00	0.23	[null]	90	[null]

Рисунок 7.11 – Сохранение данных в таблице EMPLOYEES, получение ошибки

Не забудьте сохранить введенные данные и закройте окно для ввода.

7. Теперь откройте окно Query Editor (Редактор запросов) и напишите следующий код (рисунок 7.12):

```
INSERT INTO "EmployeesDepartments"."EMPLOYEES"("FIRST_NAME", "LAST_NAME", "EMAIL", "PHONE_NUMBER",
"JOB_ID", "SALARY", "COMMISSION_PCT", "MANAGER_ID", "DEPARTMENT_ID")
VALUES('Steven','King','SKING','515.123.4567','AD_PRES',24000,null,null,90);
```

Рисунок 7.12 – Добавление данных в таблицу EMPLOYEES

Обратите внимание, что не все столбцы таблицы перечислены. Отсутствующие столбцы в операторе INSERT заполняются значением по умолчанию или значением NULL, если оно допустимо для данного столбца.

Проверьте, что запись появилась в таблице (рисунок 7.13).

```
SELECT * FROM "EmployeesDepartments"."EMPLOYEES";
```

Рисунок 7.13 – Проверка, что данные добавлены.

Задание 8. Ограничения для столбцов

1. Добавьте ограничения к таблицам "Countries"."SPOKEN_LANGUAGES" (рисунок 8.1) и "EmployeesDepartments"."EMPLOYEES" (рисунок 8.2). Для этого в окне запросов введите и выполните следующий код (рисунок 8.1):

```
ALTER TABLE "Countries"."SPOKEN_LANGUAGES"
ADD CONSTRAINT "CTRY_NUM_FK1" FOREIGN KEY ("COUNTRY_ID")
REFERENCES "Countries"."COUNTRIES" ("COUNTRY_ID");
```

Рисунок 8.1 – Добавление ограничения к таблице SPOKEN_LANGUAGES

```
ALTER TABLE "EmployeesDepartments"."EMPLOYEES"
ADD CONSTRAINT "EMP_SALARY_MIN"
CHECK ("SALARY" > 0);
```

Рисунок 8.2 – Добавление ограничения к таблице EMPLOYEES

Таким образом, вы добавили:

- внешний ключ с именем CTRY_NUM_FK1 к таблице SPOKEN_LANGUAGES, он определяет ссылку значений столбца COUNTRY_ID таблицы SPOKEN_LANGUAGES на столбец COUNTRY_ID таблицы COUNTRIES.
- проверочное ограничение CHECK для столбца SALARY таблицы "EmployeesDepartments"."EMPLOYEES", которое позволяет вводить только положительное число в данный столбец.

2. Обновите обозреватель объектов и посмотрите на ограничения, они отображаются в разделе «Ограничения» (рисунок 8.4):

Рисунок 8.3 – Обозреватель объектов.

3. Добавьте еще одно ограничение. Ограничение добавляется для таблицы JOB_HISTORY. Для этого в свойствах таблицы в разделе «Ограничения» добавьте проверочное CHECK с условием END_DATE > START_DATE (рисунок 8.4):

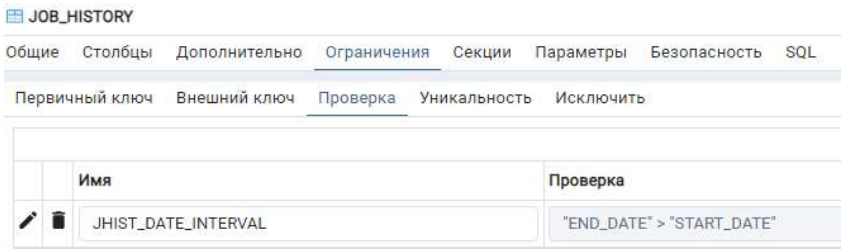

Рисунок 8.4 – Добавление ограничения к таблице JOB_HISTORY

Название ограничения при желании можно не вводить, оно сгенерируется автоматически.

Нажмите на кнопку «Сохранить». Посмотрите на ограничение в обозревателе объектов (рисунок 8.5):

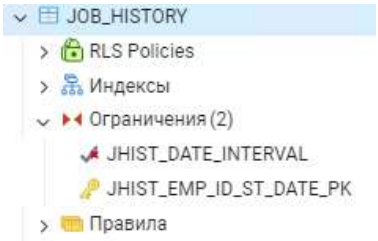

Рисунок 8.5 – Обозреватель объектов. Таблица JOB_HISTORY

Включите его, выбрав свойства этого ограничения и переключив ползунок «Не проверять?» (рисунок 8.6):

Рисунок 8.5 – Таблица JOB_HISTORY, включение ограничения

Сохраните свойства, проверьте (рисунок 8.7):

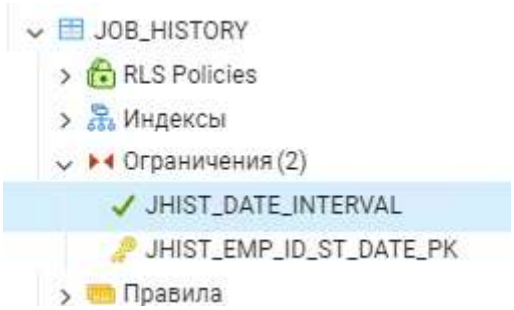

Рисунок 8.7 – Обозреватель объектов. Таблица JOB_HISTORY

4. Проверьте работоспособность ограничения. Введите и попробуйте выполнить следующие инструкции (проанализируйте результаты) (рисунок 8.8):

```
INSERT INTO "EmployeesDepartments"."JOB_HISTORY"
("EMPLOYEE_ID","START_DATE","END_DATE","JOB_ID","DEPARTMENT_ID")
VALUES(200, '06-17-1993', '09-17-1987','AD_ASST',90);

INSERT INTO "EmployeesDepartments"."JOB_HISTORY"
("EMPLOYEE_ID","START_DATE","END_DATE","JOB_ID","DEPARTMENT_ID")
VALUES(200, '09-17-1981','06-17-1993','AD_ASST',90);
```

Рисунок 8.8 – Проверка на добавление данных в таблицу JOB_HISTORY

Задание 9. Очистка таблиц и завершение создания БД HR

Завершите создание объектов базы данных и заполните её данными из подготовленных SQL файлов, содержащих команды по созданию объектов и добавлению в них данных соответственно.

1. Для корректного выполнения SQL файлов предварительно требуется очистить таблицы от данных, которые были добавлены в таблицы ранее.

Для этого произведите очистку таблицы EMPLOYEES с помощью DDL команды TRUNCATE (рисунок 9.1):

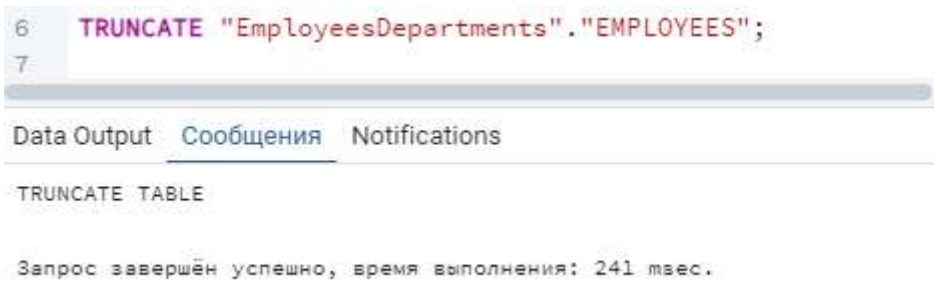

Рисунок 9.1 – Очистка таблицы EMPLOYEES

Таким же образом очистите от данных 2 другие таблицы: JOB_HISTORY, CURRENCIES.

Теперь можно выполнить скрипт по добавлению остаточных объектов базы данных. Для этого в окне запроса выполните код из файла script2_alter.sql, который находится в разделе «Литература и ПО».

Для добавления данных требуется выполнить скрипт из файла script3_insert.sql.

NB! Рекомендуется весь код выполнять последовательно, чтобы избежать ошибок.

2. Построение диаграмм базы данных

Проверьте, что все недостающие связи на ER-диаграмме появились (нажмите ПКМ на узел базы данных и выберите параметр «ERD For Database»), зафиксируйте результат в отчете.

Задание 10.

Оформить задание практическая 1 часть 1 и практическая 1 часть 2 в единый отчет и прикрепить в ответ на это задание.