|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра цифровой трансформации (ЦТ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Разработка баз данных»

**Практическое занятие № 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студенты группы | *ИКБО-50-23 Враженко Д.О.*  *.* | (подпись) | |  |
| Преподаватель | *Мажей Я.В.* | (подпись) | |  |
| Отчет представлен | «\_\_\_»\_\_\_\_\_\_\_\_2025 г. | |  | |

Москва 2025 г.

**Цель работы**:

Целью данной практической работы является формирование и закрепление у студентов фундаментальных навыков работы с реляционными базами данных на примере СУБД Postgres Pro. По завершении работы студент должен уметь:

* Сформировать практический навык определения структуры базы данных с использованием языка определения данных DDL (Data Definition Language).
* Научиться применять ограничения целостности (PRIMARY KEY, FOREIGN KEY, CHECK, UNIQUE, NOT NULL) для реализации бизнес правил и обеспечения консистентности (согласованности) данных, основываясь на теоретических положениях реляционной модели.
* Освоить составление SQL-запросов на выборку данных с использованием расширенного синтаксиса инструкции SELECT, включая выражения в списке выборки, псевдонимы и фильтрацию дубликатов с помощью DISTINCT.
* Развить умение применять разнообразные условия фильтрации записей в предложении WHERE, охватывая логические операции, проверку принадлежности диапазону и множеству, сравнение с шаблоном и корректную проверку на NULL.
* Получить базовые навыки агрегации данных с использованием GROUP BY и агрегатных функций, а также научиться корректно фильтровать агрегированные результаты с помощью предложения HAVING.

**Постановка задачи**:

Для выполнения практической работы необходимо последовательно выполнить следующие шаги, **основываясь на логической модели данных**, которая была спроектирована в рамках курса «Проектирование баз данных» **в предыдущем семестре**:

1. На основе логической модели данных, созданной в прошлом семестре, письменно описать не менее **5 различных бизнес-правил** и не менее **3 ограничений целостности** для таблиц. Выбор бизнес-правил и ограничений целостности производится на усмотрение студента. Результаты представить в виде таблицы.
2. С использованием DDL-оператора CREATE TABLE создать **все необходимые таблицы** (*согласно созданной в прошлом семестре логической модели данных*) в СУБД Postgres Pro, корректно реализовав все описанные **ограничения целостности**.
3. Заполнить созданные таблицы согласованными тестовыми данными (**не менее 5-7 записей на таблицу**, где это применимо) с помощью оператора INSERT INTO.
4. Составить и выполнить не менее **6 SQL-запросов** к таблицам, иллюстрирующих использование различных операторов SELECT и WHERE, согласно перечню, указанному в задании (*см. Ход выполнения работы*). В составленных запросах должны быть **использованы все приведённые примеры –** .
5. Составить и выполнить **по два SQL-запроса** к таблицам для демонстрации работы предложений ORDER BY, GROUP BY и HAVING.
6. Каждый SQL-запрос **сопроводить комментарием**, объясняющим его назначение и логику работы.

**ХОД ВЫПОЛНЕНИЯ РАБОТЫ**

**1. Анализ и описание ограничений целостности**

Таблица 1 — Описание ограничений для таблицы employee (сотрудник)

|  |  |  |  |
| --- | --- | --- | --- |
| **Название столбца** | **Тип данных** | **Ограничение** | **Обоснование (Бизнес-правило)** |
| id | SERIAL | NOT NULL PRIMARY KEY | Уникальный идентификатор сотрудника, генерируется автоматически. |
| id\_position | INTEGER | NOT NULL | Ссылка на должность является обязательной и не может быть пустой. |
| last\_name | VARCHAR(30) | NOT NULL | Фамилия сотрудника является обязательной и не может быть пустой. |
| first\_name | VARCHAR(30) | NOT NULL | Имя сотрудника является обязательным и не может быть пустым. |
| phone\_number | VARCHAR(10) | NOT NULL | Номер телефона сотрудника является обязательным и не может быть пустым. |
| registration\_address | TEXT | NOT NULL | Адрес регистрации сотрудника является обязательным и не может быть пустым. |
| employment\_date | DATE | NOT NULL | Дата назначения сотрудника на должность не может быть пустой и является обязательной. |
| contract\_end\_date | DATE | NOT NULL | Дата окончания контракта с сотрудником не может быть пустой и является обязательной. |

**2. Создание структуры данных**

Листинг 1 — Создание всех таблиц

|  |
| --- |
| **CREATE** **TABLE** position (  id\_position serial **NOT** **NULL** **PRIMARY** **KEY**,  position\_name **varchar**(30) **NOT** **NULL**,  access\_category **varchar**(30) **NOT** **NULL**,  salary **decimal** **NOT** **NULL**  );  **CREATE** **TABLE** department (  id\_department serial **NOT** **NULL** **PRIMARY** **KEY**,  department\_name **varchar**(30) **NOT** **NULL**,  department\_head **varchar**(30) **NOT** **NULL**  );  **CREATE** **TABLE** employee (  id\_employee serial **NOT** **NULL** **PRIMARY** **KEY**,  id\_position **integer** **NOT** **NULL**,  last\_name **varchar**(30) **NOT** **NULL**,  first\_name **varchar**(30) **NOT** **NULL**,  phone\_number **varchar**(10) **NOT** **NULL**,  registration\_address **text** **NOT** **NULL**,  employment\_date **date** **NOT** **NULL**,  contract\_end\_date **date** **NOT** **NULL**  );  **CREATE** **TABLE** termination\_request (  id\_termination\_request serial **NOT** **NULL** **PRIMARY** **KEY**,  id\_department **integer** **NOT** **NULL**,  id\_employee **integer** **NOT** **NULL**,  request\_date **date** **NOT** **NULL**,  reason **text** **NOT** **NULL**,  status **varchar**(50) **NOT** **NULL**,  termination\_date **date** **NOT** **NULL**  );  **CREATE** **TABLE** termination\_type (  id\_termination\_type serial **NOT** **NULL** **PRIMARY** **KEY**,  type\_name **varchar**(50) **NOT** **NULL**,  description **text** **NOT** **NULL**  );  **CREATE** **TABLE** request\_termination\_type (  id\_request\_termination\_type serial **NOT** **NULL** **PRIMARY** **KEY**,  id\_termination\_request **integer** **NOT** **NULL**,  id\_termination\_type **integer** **NOT** **NULL**  );  **CREATE** **TABLE** document (  id\_document serial **NOT** **NULL** **PRIMARY** **KEY**,  document\_type **varchar**(30) **NOT** **NULL**,  creation\_date **date** **NOT** **NULL**,  id\_termination\_request **integer** **NOT** **NULL**,  content **text** **NOT** **NULL**  );  **CREATE** **TABLE** payment (  id\_payment serial **NOT** **NULL** **PRIMARY** **KEY**,  id\_termination\_request **integer** **NOT** **NULL**,  amount **decimal** **NOT** **NULL**,  payment\_date **date** **NOT** **NULL**,  comment **text** **NOT** **NULL**  );  **CREATE** **TABLE** inventory (  id\_inventory serial **NOT** **NULL** **PRIMARY** **KEY**,  item\_name **varchar**(30) **NOT** **NULL**,  category **varchar**(30) **NOT** **NULL**,  status **varchar**(20) **NOT** **NULL**  );  **CREATE** **TABLE** employee\_inventory (  id\_employee\_inventory serial **NOT** **NULL** **PRIMARY** **KEY**,  id\_employee **integer** **NOT** **NULL**,  id\_inventory **integer** **NOT** **NULL**  );  **ALTER** **TABLE** request\_termination\_type **ADD** **CONSTRAINT** request\_termination\_type\_id\_termination\_request\_fk **FOREIGN** **KEY** (id\_termination\_request) **REFERENCES** termination\_request(id\_termination\_request);  **ALTER** **TABLE** request\_termination\_type **ADD** **CONSTRAINT** request\_termination\_type\_id\_termination\_type\_fk **FOREIGN** **KEY** (id\_termination\_type) **REFERENCES** termination\_type(id\_termination\_type);  **ALTER** **TABLE** employee\_inventory **ADD** **CONSTRAINT** employee\_inventory\_id\_inventory\_fk **FOREIGN** **KEY** (id\_inventory) **REFERENCES** inventory(id\_inventory);  **ALTER** **TABLE** employee\_inventory **ADD** **CONSTRAINT** employee\_inventory\_id\_employee\_fk **FOREIGN** **KEY** (id\_employee) **REFERENCES** employee(id\_employee);  **ALTER** **TABLE** payment **ADD** **CONSTRAINT** payment\_id\_termination\_request\_fk **FOREIGN** **KEY** (id\_termination\_request) **REFERENCES** termination\_request(id\_termination\_request);  **ALTER** **TABLE** document **ADD** **CONSTRAINT** document\_id\_termination\_request\_fk **FOREIGN** **KEY** (id\_termination\_request) **REFERENCES** termination\_request(id\_termination\_request);  **ALTER** **TABLE** termination\_request **ADD** **CONSTRAINT** termination\_request\_id\_department\_fk **FOREIGN** **KEY** (id\_department) **REFERENCES** department(id\_department);  **ALTER** **TABLE** termination\_request **ADD** **CONSTRAINT** termination\_request\_id\_employee\_fk **FOREIGN** **KEY** (id\_employee) **REFERENCES** employee(id\_employee);  **ALTER** **TABLE** employee **ADD** **CONSTRAINT** employee\_id\_position\_fk **FOREIGN** **KEY** (id\_position) **REFERENCES** position(id\_position); |

**3. Заполнение таблиц данными (DML – Data Manipulation Language)**

Листинг 2 — Заполнение всех таблиц

|  |
| --- |
| **INSERT** **INTO** position (position\_name, access\_category, salary) **VALUES**  ('Менеджер', 'Администрация', 50000.00),  ('Повар', 'Кухня', 35000.00),  ('Кассир', 'Обслуживание', 30000.00),  ('Уборщик', 'Обслуживание', 25000.00);  **INSERT** **INTO** department (department\_name, department\_head) **VALUES**  ('Кухня', 'Иван Петров'),  ('Обслуживание', 'Мария Сидорова'),  ('Администрация', 'Алексей Иванов');  **INSERT** **INTO** employee (id\_position, last\_name, first\_name, phone\_number, registration\_address, employment\_date, contract\_end\_date) **VALUES**  (1, 'Петров', 'Иван', '9123456789', 'ул. Ленина, 10', '2020-01-15', '2025-01-15'),  (1, 'Сидорова', 'Мария', '9234567890', 'пр. Мира, 5', '2019-05-20', '2024-05-20'),  (1, 'Иванов', 'Алексей', '9345678901', 'ул. Советская, 3', '2021-03-10', '2026-03-10'),  (2, 'Васильев', 'Дмитрий', '9456789012', 'ул. Гагарина, 7', '2022-02-01', '2023-02-01'),  (3, 'Кузнецова', 'Ольга', '9567890123', 'пр. Победы, 12', '2021-07-15', '2024-07-15'),  (4, 'Смирнов', 'Андрей', '9678901234', 'ул. Лесная, 9', '2023-01-10', '2023-12-31');  **INSERT** **INTO** termination\_type (type\_name, description) **VALUES**  ('По собственному желанию', 'Увольнение по инициативе сотрудника'),  ('Сокращение штата', 'Увольнение в связи с сокращением численности персонала'),  ('Нарушение дисциплины', 'Увольнение за нарушение трудовой дисциплины');  **INSERT** **INTO** termination\_request (id\_department, id\_employee, request\_date, reason, status, termination\_date) **VALUES**  (3, 6, '2023-11-01', 'Окончание контракта', 'Завершено', '2023-12-31'),  (2, 5, '2023-10-15', 'Переход на другую работу', 'В процессе', '2024-01-01');  **INSERT** **INTO** request\_termination\_type (id\_termination\_request, id\_termination\_type) **VALUES**  (1, 1),  (2, 3);  **INSERT** **INTO** document (document\_type, creation\_date, id\_termination\_request, content) **VALUES**  ('Заявление на увольнение', '2023-10-30', 1, 'Заявление на увольнение по собственному желанию от Смирнова А.'),  ('Приказ об увольнении', '2023-12-31', 1, 'Приказ №123 об увольнении Смирнова А.А.');  **INSERT** **INTO** payment (id\_termination\_request, amount, payment\_date, comment) **VALUES**  (1, 25000.00, '2023-12-31', 'Окончательный расчет по увольнению');  **INSERT** **INTO** inventory (item\_name, category, status) **VALUES**  ('Нож поварской', 'Кухонное оборудование', 'В использовании'),  ('Кассовый аппарат', 'Кассовое оборудование', 'На складе'),  ('Фартук', 'Спецодежда', 'В использовании'),  ('Стол кухонный', 'Мебель', 'На складе');  **INSERT** **INTO** employee\_inventory (id\_employee, id\_inventory) **VALUES**  (4, 1),  (5, 2),  (4, 3),  (6, 4); |

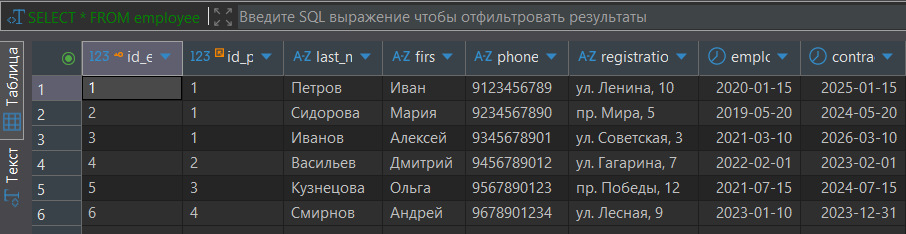
**4. Составление запросов на выборку (часть 1)**

**4.1 Элементы списка выборки – SELECT**

**4.1.1 Выбор всех столбцов (\*)**

Листинг 3 — Выбор всех столбцов

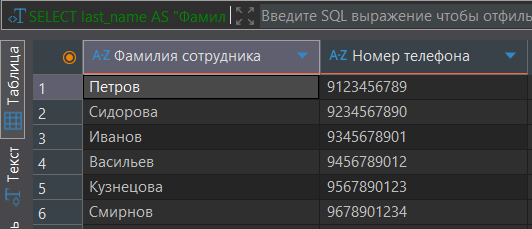
|  |
| --- |
| **SELECT** \* **FROM** employee; |

Рисунок 1 — Выбор всех столбцов

**4.1.2 Выбор конкретных полей и использование псевдонима (AS)**

Листинг 4 — Использование псевдонима

|  |
| --- |
| **SELECT**  last\_name **AS** *"Фамилия сотрудника"*,  phone\_number **AS** *"Номер телефона"*  **FROM**  employee; |

Рисунок 2 — Использование псевдонима

**4.1.3 Выражение в списке выборки**

Листинг 5 — Использование выражения

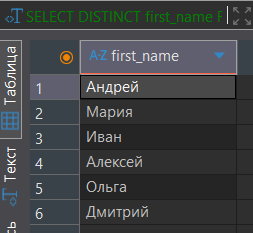
|  |
| --- |
| -- Пример, взятый из методического материала  **SELECT**  **name**,  price,  quantity\_in\_stock,  price \* quantity\_in\_stock **AS** *total\_value*  **FROM**  medicines; |

В имеющейся базе данных нет данных, которые можно между собой умножать или складывать, поэтому нет возможности привести пример выполнения данной команды.

**4.1.4 Удаление дубликатов (DISTINCT)**

Листинг 6 — Удаление дубликатов

|  |
| --- |
| **SELECT** **DISTINCT**  first\_name  **FROM**  employee; |

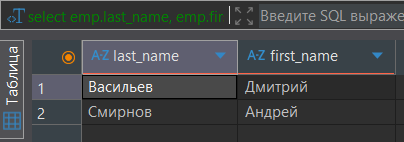
Рисунок 3 — Удаление дубликатов

**4.2 Условия фильтрации – WHERE**

**4.2.1 Простое условие и логические связки (AND, OR)**

Листинг 7 — Простое условие и логические связки (AND, OR)

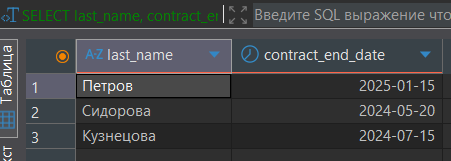
|  |
| --- |
| **SELECT**  *emp*.last\_name,  *emp*.first\_name  **FROM**  employee **AS** *emp*  **JOIN**  position **AS** *pos*  **ON**  *emp*.id\_position = *pos*.id\_position  **WHERE**  *emp*.contract\_end\_date < '2024-01-01' **AND** *pos*.salary < 40000.00; |

Рисунок 4 — Простое условие и логические связки (AND, OR)

**4.2.2 Проверка принадлежности диапазону (BETWEEN)**

Листинг 8 — Проверка принадлежности диапазону (BETWEEN)

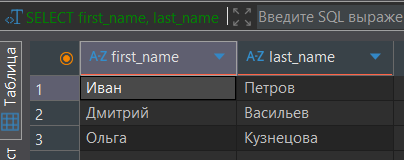
|  |
| --- |
| **SELECT**  last\_name,  contract\_end\_date  **FROM**  employee  **WHERE**  contract\_end\_date **BETWEEN** '2024-01-01' **AND** '2026-01-01'; |

Рисунок 5 — Проверка принадлежности диапазону (BETWEEN)

**4.2.3 Проверка вхождения в множество (IN)**

Листинг 9 — Проверка вхождения в множество (IN)

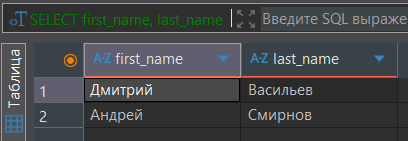
|  |
| --- |
| **SELECT**  first\_name,  last\_name  **FROM**  employee  **WHERE**  **first\_name IN ('Иван', 'Антон', 'Дмитрий', 'Ольга');** |

Рисунок 6 — Проверка вхождения в множество (IN)

**4.2.4 Сравнение с шаблоном (LIKE)**

Листинг 10 — Сравнение с шаблоном (LIKE)

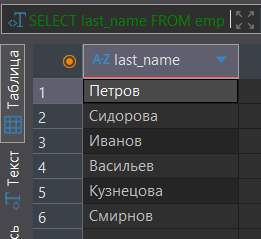
|  |
| --- |
| **SELECT**  first\_name,  last\_name  **FROM**  employee  **WHERE**  **first\_name LIKE '%р\_й%';** |

Рисунок 7 — Сравнение с шаблонов (LIKE)

**4.2.5 Проверка на NULL**

Листинг 11 — Проверка на NULL

|  |
| --- |
| **SELECT**  last\_name  **FROM**  employee  **WHERE**  **id\_employee is NOT NULL;** |

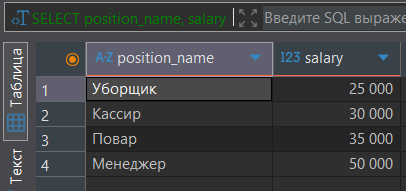
Рисунок 8 — Проверка на NULL

**5. Составление запросов на выборку (часть 2)**

**5.1 Сортировка результатов (ORDER BY)**

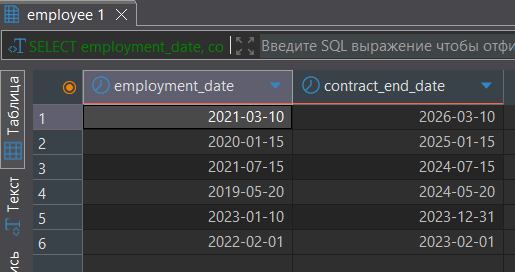
Листинг 12 — Сортировка результатов по возрастанию

|  |
| --- |
| **SELECT**  position\_name,  salary  **FROM**  position  **ORDER** **BY**  **salary ASC;** |

Рисунок 9 — Сортировка результатов по возрастанию

Листинг 13 — Сортировка результатов по убыванию

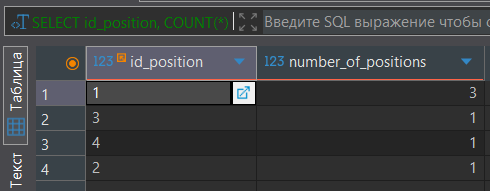
|  |
| --- |
| **SELECT**  employment\_date,  contract\_end\_date  **FROM**  employee  **ORDER** **BY**  contract\_end\_date **DESC**,  **employment\_date DESC;** |

Рисунок 10 — Сортировка результатов по убыванию

**5.2 Группировка и агрегатные функции (GROUP BY)**

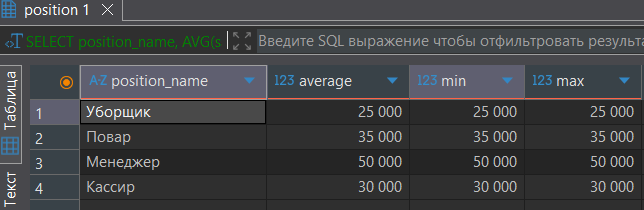
Листинг 14 — Группировка и агрегатные функции (1)

|  |
| --- |
| **SELECT**  id\_position,  **COUNT**(\*) **AS** *number\_of\_positions*  **FROM**  employee  **GROUP** **BY**  id\_position; |

Рисунок 11 — Группировка и агрегатные функции (1)

Листинг 15 — Группировка и агрегатные функции (2)

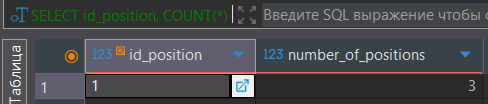
|  |
| --- |
| **SELECT**  position\_name,  **AVG**(salary) **AS** *average*,  **MIN**(salary) **AS** *min*,  **MAX**(salary) **AS** *max*  **FROM**  position  **GROUP** **BY**  **position\_name;** |

Рисунок 12 — Группировка и агрегатные функции

**5.3 Фильтрация групп (HAVING)**

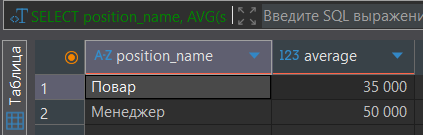
Листинг 16 — Фильтрация групп (1)

|  |
| --- |
| **SELECT**  id\_position,  **COUNT**(\*) **AS** *number\_of\_positions*  **FROM**  employee  **GROUP** **BY**  id\_position  **HAVING**  **COUNT**(\*) > 2; |

Рисунок 13 — Фильтрация групп (1)

Листинг 17 — Фильтрация групп (2)

|  |
| --- |
| **SELECT**  position\_name,  **AVG**(salary) **AS** *average*  **FROM**  position  **GROUP** **BY**  position\_name  **HAVING**  **AVG**(salary) > 34999.00; |

Рисунок 14 — Фильтрация групп (2)