



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);

```

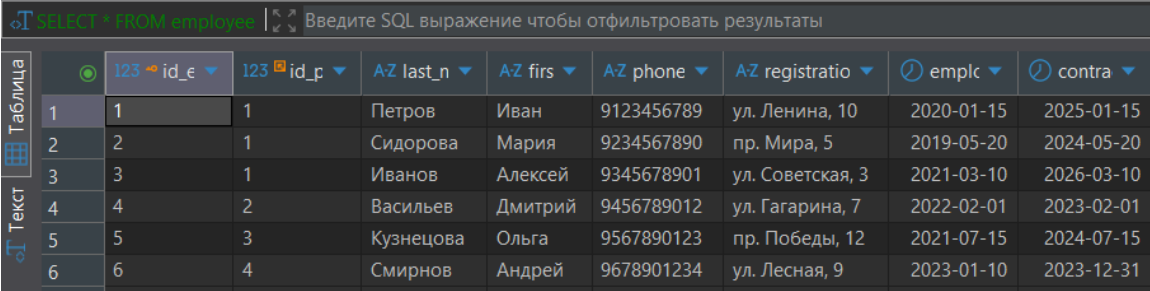
4. Составление запросов на выборку (часть 1)

4.1 Элементы списка выборки – SELECT

4.1.1 Выбор всех столбцов (*)

Листинг 3 — Выбор всех столбцов

```
SELECT * FROM employee;
```



| | 123 id_ε | 123 id_p | A-Z last_n | A-Z first_n | A-Z phone | A-Z registratio | emplc | contra |
|---|----------|----------|------------|-------------|------------|------------------|------------|------------|
| 1 | 1 | 1 | Петров | Иван | 9123456789 | ул. Ленина, 10 | 2020-01-15 | 2025-01-15 |
| 2 | 2 | 1 | Сидорова | Мария | 9234567890 | пр. Мира, 5 | 2019-05-20 | 2024-05-20 |
| 3 | 3 | 1 | Иванов | Алексей | 9345678901 | ул. Советская, 3 | 2021-03-10 | 2026-03-10 |
| 4 | 4 | 2 | Васильев | Дмитрий | 9456789012 | ул. Гагарина, 7 | 2022-02-01 | 2023-02-01 |
| 5 | 5 | 3 | Кузнецова | Ольга | 9567890123 | пр. Победы, 12 | 2021-07-15 | 2024-07-15 |
| 6 | 6 | 4 | Смирнов | Андрей | 9678901234 | ул. Лесная, 9 | 2023-01-10 | 2023-12-31 |

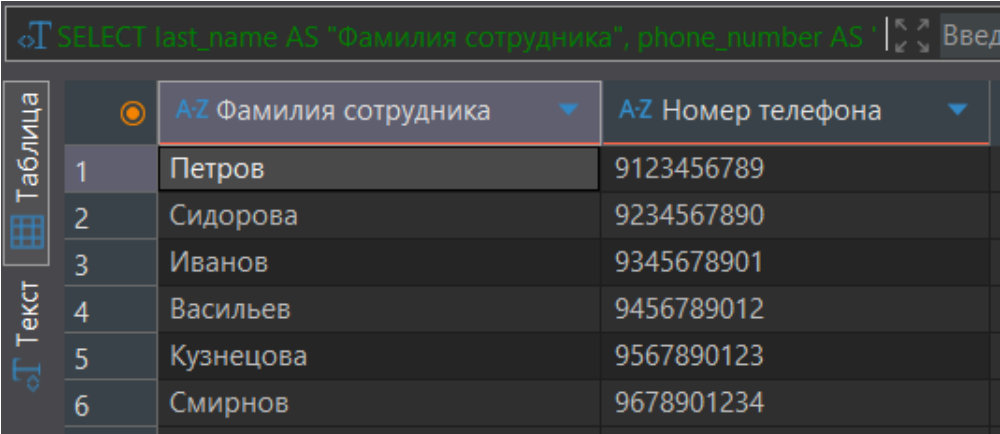
Рисунок 1 — Выбор всех столбцов

Листинг 4 — Использование псевдонима

```

SELECT
    last_name AS "Фамилия сотрудника",
    phone_number AS "Номер телефона"
FROM
    employee;

```



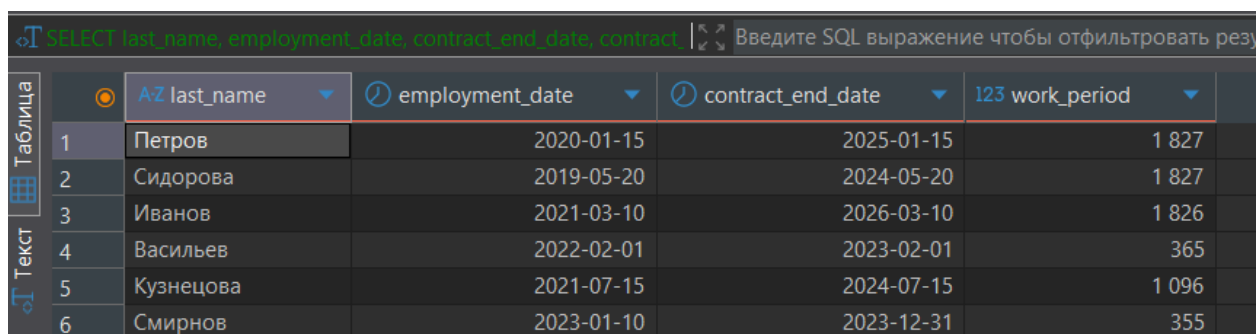
| | A-Z Фамилия сотрудника | A-Z Номер телефона |
|---|------------------------|--------------------|
| 1 | Петров | 9123456789 |
| 2 | Сидорова | 9234567890 |
| 3 | Иванов | 9345678901 |
| 4 | Васильев | 9456789012 |
| 5 | Кузнецова | 9567890123 |
| 6 | Смирнов | 9678901234 |

Рисунок 2 — Использование псевдонима

4.1.3 Выражение в списке выборки

Листинг 5 — Использование выражения

```
SELECT
    last_name,
    employment_date,
    contract_end_date,
    contract_end_date - employment_date AS work_period
FROM
    employee;
```



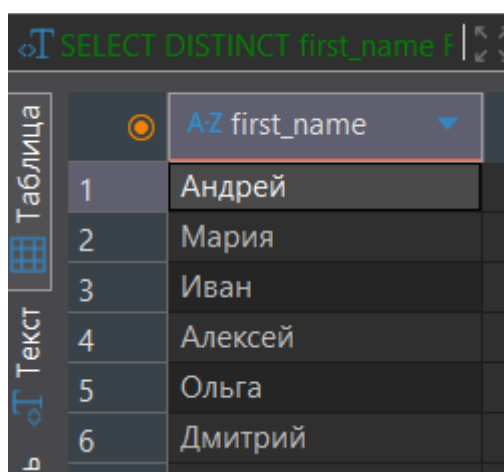
| | AZ last_name | employment_date | contract_end_date | 123 work_period |
|---|--------------|-----------------|-------------------|-----------------|
| 1 | Петров | 2020-01-15 | 2025-01-15 | 1 827 |
| 2 | Сидорова | 2019-05-20 | 2024-05-20 | 1 827 |
| 3 | Иванов | 2021-03-10 | 2026-03-10 | 1 826 |
| 4 | Васильев | 2022-02-01 | 2023-02-01 | 365 |
| 5 | Кузнецова | 2021-07-15 | 2024-07-15 | 1 096 |
| 6 | Смирнов | 2023-01-10 | 2023-12-31 | 355 |

Рисунок 3 — Использование выражения

4.1.4 Удаление дубликатов (DISTINCT)

Листинг 6 — Удаление дубликатов

```
SELECT DISTINCT
    first_name
FROM
    employee;
```



| | AZ first_name |
|---|---------------|
| 1 | Андрей |
| 2 | Мария |
| 3 | Иван |
| 4 | Алексей |
| 5 | Ольга |
| 6 | Дмитрий |

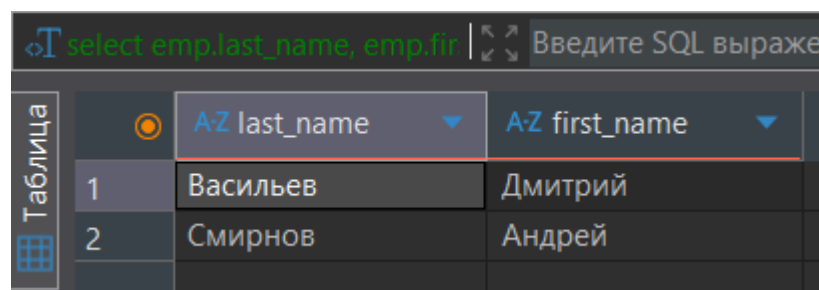
Рисунок 4 — Удаление дубликатов

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;
```



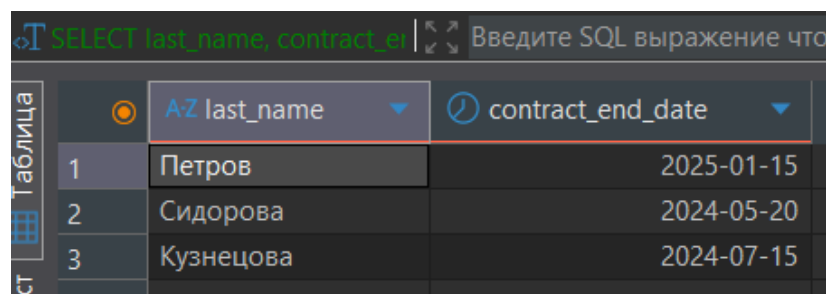
| Таблица | AZ last_name | AZ first_name |
|---------|--------------|---------------|
| 1 | Васильев | Дмитрий |
| 2 | Смирнов | Андрей |

Рисунок 5 — Простое условие и логические связки

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';
```



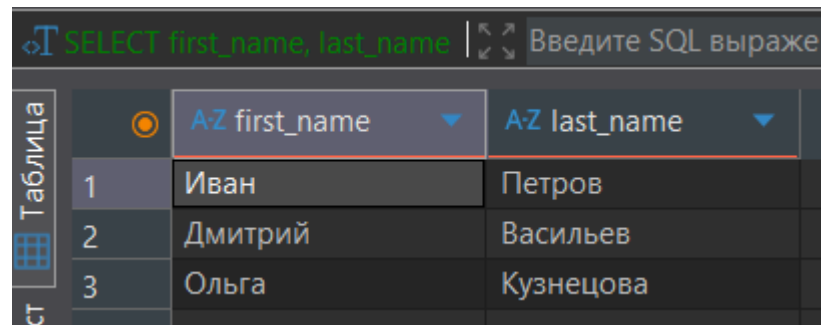
| Таблица | AZ last_name | contract_end_date |
|---------|--------------|-------------------|
| 1 | Петров | 2025-01-15 |
| 2 | Сидорова | 2024-05-20 |
| 3 | Кузнецова | 2024-07-15 |

Рисунок 6 — Проверка принадлежности диапазону

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');
```



The screenshot shows a SQL query result in a dark-themed interface. The query is `SELECT first_name, last_name`. The result is displayed as a table with two columns: `A-Z first_name` and `A-Z last_name`. The table contains three rows of data.

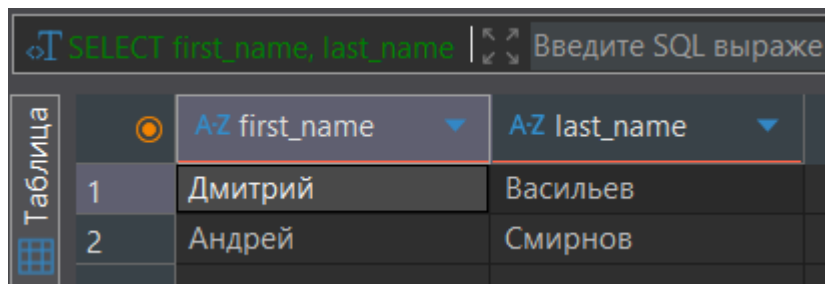
| Таблица | | A-Z first_name | A-Z last_name |
|---------|--|----------------|---------------|
| 1 | | Иван | Петров |
| 2 | | Дмитрий | Васильев |
| 3 | | Ольга | Кузнецова |

Рисунок 7 — 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 '%п_й%';
```



The screenshot shows a SQL query result in a dark-themed interface. The query is `SELECT first_name, last_name`. The result is displayed as a table with two columns: `A-Z first_name` and `A-Z last_name`. The table contains two rows of data.

| Таблица | | A-Z first_name | A-Z last_name |
|---------|--|----------------|---------------|
| 1 | | Дмитрий | Васильев |
| 2 | | Андрей | Смирнов |

Рисунок 8 — Сравнение с шаблоном (LIKE)

4.2.5 Проверка на NULL

Листинг 11 — Проверка на NULL

```
SELECT
last_name
FROM
```

```
employee
WHERE
id_employee is NOT NULL;
```

| | A-Z last_name |
|---|---------------|
| 1 | Петров |
| 2 | Сидорова |
| 3 | Иванов |
| 4 | Васильев |
| 5 | Кузнецова |
| 6 | Смирнов |

Рисунок 9 — 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;
```

| | A-Z position_name | 123 salary |
|---|-------------------|------------|
| 1 | Уборщик | 25 000 |
| 2 | Кассир | 30 000 |
| 3 | Повар | 35 000 |
| 4 | Менеджер | 50 000 |

Рисунок 10 — 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;
```

employee 1 X

SELECT employment_date, co | Введите SQL выражение чтобы отфи

| Таблица | employment_date | contract_end_date |
|---------|-----------------|-------------------|
| 1 | 2021-03-10 | 2026-03-10 |
| 2 | 2020-01-15 | 2025-01-15 |
| 3 | 2021-07-15 | 2024-07-15 |
| 4 | 2019-05-20 | 2024-05-20 |
| 5 | 2023-01-10 | 2023-12-31 |
| 6 | 2022-02-01 | 2023-02-01 |

Текст

Рисунок 11 — 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;
```

SELECT id_position, COUNT(*) | Введите SQL выражение чтобы с

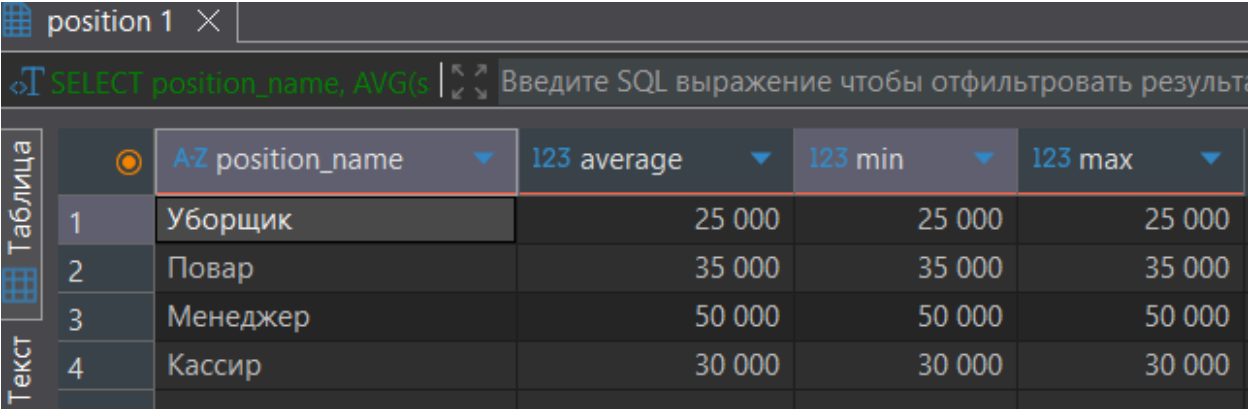
| Таблица | 123 id_position | 123 number_of_positions |
|---------|-----------------|-------------------------|
| 1 | 1 | 3 |
| 2 | 3 | 1 |
| 3 | 4 | 1 |
| 4 | 2 | 1 |

Текст

Рисунок 12 — 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;
```



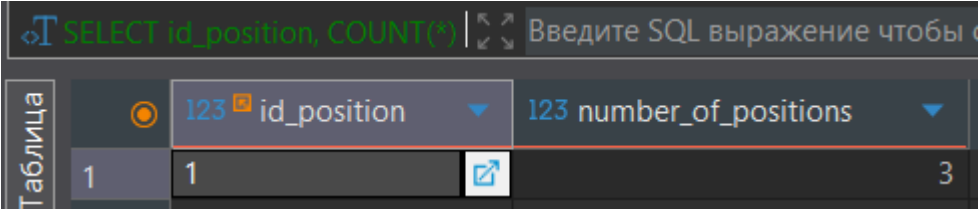
| | A-Z position_name | 123 average | 123 min | 123 max |
|---|-------------------|-------------|---------|---------|
| 1 | Уборщик | 25 000 | 25 000 | 25 000 |
| 2 | Повар | 35 000 | 35 000 | 35 000 |
| 3 | Менеджер | 50 000 | 50 000 | 50 000 |
| 4 | Кассир | 30 000 | 30 000 | 30 000 |

Рисунок 13 — Группировка и агрегатные функции (2)

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;
```

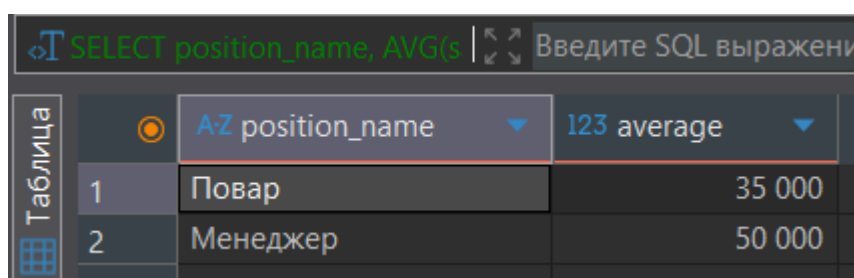


| 123 id_position | 123 number_of_positions |
|-----------------|-------------------------|
| 1 | 3 |

Рисунок 14 — 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;
```



| | AZ position_name | 123 average |
|---|------------------|-------------|
| 1 | Повар | 35 000 |
| 2 | Менеджер | 50 000 |

Рисунок 15 — Фильтрация групп (2)