Лабораторная работа №10 Функции в SQL

Цель работы:

Освоить использование встроенных функций PostgreSQL для работы с числами, строками и агрегирования данных. Изучить применение условной конструкции CASE для обработки данных в запросах.

Программное обеспечение: PostgreSQL, DBeaver
Время: 2 академических часа

Подготовка базы данных

Создание таблиц с разнообразными типами данных

```
SET search_path TO ваш_логин;

-- Таблица сотрудников компании
CREATE TABLE employees (
    employee_id SERIAL PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    email VARCHAR(100) UNIQUE,
    department VARCHAR(50),
    position VARCHAR(50),
    salary DECIMAL(10,2),
    bonus_percent DECIMAL(5,2) DEFAULT 0,
    hire_date DATE NOT NULL,
    birth_date DATE,
    phone VARCHAR(20),
    address TEXT,
    performance_score INTEGER CHECK (performance_score BETWEEN 1 AND
100),
    years_experience INTEGER
);

-- Таблица проектов
```

```sql
CREATE TABLE projects (
    project_id SERIAL PRIMARY KEY,
    project_name VARCHAR(100) NOT NULL,
    start_date DATE NOT NULL,
    end_date DATE,
    budget DECIMAL(12,2),
    status VARCHAR(20) DEFAULT 'active'
);

-- Таблица назначений сотрудников на проекты
CREATE TABLE project_assignments (
    assignment_id SERIAL PRIMARY KEY,
    employee_id INTEGER REFERENCES employees(employee_id),
    project_id INTEGER REFERENCES projects(project_id),
    role VARCHAR(50),
    hours_worked DECIMAL(6,2),
    hourly_rate DECIMAL(8,2),
    assignment_date DATE DEFAULT CURRENT_DATE
);

-- Таблица продаж
CREATE TABLE sales (
    sale_id SERIAL PRIMARY KEY,
    employee_id INTEGER REFERENCES employees(employee_id),
    sale_date DATE NOT NULL,
    product_name VARCHAR(100),
    quantity INTEGER,
    unit_price DECIMAL(10,2),
    discount_percent DECIMAL(5,2) DEFAULT 0,
    region VARCHAR(50)
);
```

Заполнение таблиц тестовыми данными

```sql
-- Сотрудники
INSERT INTO employees (first_name, last_name, email, department,
position, salary, bonus_percent, hire_date, birth_date, phone, address,
performance_score, years_experience) VALUES
('Иван', 'Иванов', 'ivanov@company.com', 'IT', 'Разработчик', 85000.50,
15.00, '2020-03-15', '1990-05-20', '+7(912)345-67-89', 'ул. Ленина, д.
10, кв. 25', 85, 5),
```

```
('Мария', 'Петрова-Сидорова', 'petrova@company.com', 'Маркетинг',
'Менеджер', 75000.00, 12.50, '2021-06-10', '1992-08-12', '+7(912)987-65-
43', 'пр. Мира, д. 15', 92, 3),
('Алексей', 'Смирнов', 'smirnov@company.com', 'IT', 'Тимлид', 120000.75,
20.00, '2018-11-05', '1985-12-03', '+7(912)111-22-33', 'ул. Центральная,
д. 5', 95, 10),
('Ольга', 'Кузнецова', 'kuznetsova@company.com', 'Финансы', 'Аналитик',
68000.25, 10.00, '2022-01-20', '1995-03-25', '+7(912)444-55-66', 'ул.
Садовая, д. 8', 78, 2),
('Дмитрий', 'Попов', 'popov@company.com', 'Продажи', 'Директор',
150000.00, 25.00, '2015-09-12', '1980-11-08', '+7(912)777-88-99', 'пр.
Победы, д. 20', 88, 15),
('Елена', 'Васильева', 'vasilieva@company.com', 'HR', 'Рекрутер',
55000.80, 8.00, '2019-04-18', '1991-07-30', '+7(912)123-45-67', 'ул.
Молодежная, д. 3', 91, 4),
('Сергей', 'Новиков', 'novikov@company.com', 'IT', 'Тестировщик',
65000.00, 10.00, '2023-02-22', '1993-09-14', '+7(912)876-54-32', 'ул.
Школьная, д. 7', 82, 1),
('Анна', 'Морозова', 'morozova@company.com', 'Финансы', 'Бухгалтер',
72000.40, 12.00, '2020-12-01', '1988-01-18', '+7(912)234-56-78', 'ул.
Лесная, д. 12', 87, 6),
('Павел', 'Волков', 'volkov@company.com', 'Продажи', 'Менеджер',
95000.00, 18.00, '2017-05-30', '1982-04-12', '+7(912)345-67-89', 'пр.
Строителей, д. 30', 94, 8),
('Юлия', 'Зайцева', 'zaytseva@company.com', 'Маркетинг', 'Дизайнер',
62000.90, 10.50, '2021-08-15', '1994-06-28', '+7(912)456-78-90', 'ул.
Речная, д. 9', 89, 3);

-- Проекты
INSERT INTO projects (project_name, start_date, end_date, budget,
status) VALUES
('Разработка CRM-системы', '2023-01-15', '2023-12-31', 500000.00,
'completed'),
('Мобильное приложение', '2023-06-01', '2024-03-31', 350000.00,
'active'),
('Ребрендинг компании', '2023-09-10', '2024-01-31', 200000.00,
'active'),
('Автоматизация отчетности', '2023-03-01', '2023-08-30', 180000.00,
'completed'),
('Внедрение ERP-системы', '2024-01-15', '2024-12-31', 800000.00,
'planned');
```

```sql
-- Назначения на проекты
INSERT INTO project_assignments (employee_id, project_id, role,
hours_worked, hourly_rate, assignment_date) VALUES
(1, 1, 'Backend разработчик', 320.5, 1500.00, '2023-01-20'),
(1, 2, 'API разработчик', 120.0, 1600.00, '2023-07-01'),
(2, 3, 'Руководитель проекта', 200.0, 1800.00, '2023-09-15'),
(3, 1, 'Архитектор', 450.0, 2000.00, '2023-01-15'),
(3, 4, 'Консультант', 80.0, 2200.00, '2023-03-10'),
(4, 4, 'Аналитик', 250.0, 1200.00, '2023-03-05'),
(5, 2, 'Product Owner', 180.0, 2500.00, '2023-06-15'),
(6, 1, 'Тестировщик', 150.0, 1000.00, '2023-02-01'),
(7, 2, 'QA инженер', 95.5, 1100.00, '2023-07-10'),
(8, 3, 'Финансовый контролер', 120.0, 1300.00, '2023-09-20'),
(9, 1, 'Менеджер', 300.0, 1700.00, '2023-01-25'),
(10, 2, 'Дизайнер UI/UX', 210.0, 1400.00, '2023-06-05');

-- Продажи
INSERT INTO sales (employee_id, sale_date, product_name, quantity,
unit_price, discount_percent, region) VALUES
(5, '2024-01-10', 'Ноутбук Pro', 2, 75000.00, 5.00, 'Центральный'),
(5, '2024-01-15', 'Монитор 27"', 5, 25000.00, 10.00, 'Центральный'),
(9, '2024-01-20', 'Смартфон', 10, 45000.00, 7.50, 'Восточный'),
(5, '2024-01-25', 'Планшет', 3, 35000.00, 0.00, 'Центральный'),
(9, '2024-02-05', 'Ноутбук Basic', 4, 40000.00, 15.00, 'Западный'),
(5, '2024-02-10', 'Монитор 24"', 8, 18000.00, 12.50, 'Центральный'),
(9, '2024-02-15', 'Смартфон', 6, 38000.00, 5.00, 'Восточный'),
(5, '2024-02-20', 'Принтер', 2, 22000.00, 8.00, 'Центральный'),
(9, '2024-02-25', 'Планшет', 5, 32000.00, 10.00, 'Западный'),
(5, '2024-03-01', 'Ноутбук Pro', 1, 78000.00, 3.00, 'Центральный');
```

Конструкция CASE

Синтаксис конструкции CASE

Конструкция `CASE` позволяет выполнять условную логику в SQL-запросах. Существует две формы:

Простая форма:

```
CASE выражение
    WHEN значение1 THEN результат1
    WHEN значение2 THEN результат2
    ...
    ELSE результат_по_умолчанию
END
```

Примеры использования CASE

Определение стажа работы:

```
SELECT
    first_name,
    last_name,
    hire_date,
    CASE
        WHEN hire_date > CURRENT_DATE - INTERVAL '1 year' THEN 'Новичок'
        WHEN hire_date > CURRENT_DATE - INTERVAL '3 years' THEN
'Опытный'
        ELSE 'Ветеран'
    END as experience_level
FROM employees;
```

Задание для студентов: 10 запросов с использованием функций

Запрос 1: Простые строковые функции

Напишите запрос, который выводит полное имя сотрудника в формате "Фамилия И." (например, "Иванов И.") и email, преобразованный в нижний регистр.

Запрос 2: Математические вычисления

Вычислите общую зарплату сотрудника за год с учетом бонуса (salary *12* (1 + bonus_percent/100)). Округлите результат до 2 знаков после запятой и выведите вместе с именем сотрудника.

Запрос 3: Работа с датами

Найдите стаж работы сотрудников в полных годах (разница между текущей датой и hire_date). Выведите имя сотрудника и стаж.

Запрос 4: Агрегатная функция COUNT

Подсчитайте количество сотрудников в каждом отделе. Выведите название отдела и количество сотрудников.

Запрос 5: Агрегатные функции SUM и AVG

Для каждого проекта выведите общее количество отработанных часов и среднюю почасовую ставку.

Запрос 6: Агрегатная функция MAX/MIN

Найдите сотрудника с самой высокой и самой низкой зарплатой. Выведите их имена и зарплаты.

Запрос 7: Комбинирование строковых и математических функций

Выведите информацию о продажах: продукт, количество, цена за единицу и общая сумма продажи (quantity *unit_price* (1 - discount_percent/100)). Форматируйте общую сумму как "XXX XXX,XX руб."

Запрос 8: Группировка с HAVING

Найдите отделы, где средняя зарплата превышает 70000 рублей. Выведите название отдела и среднюю зарплату.

Запрос 9: Вложенные агрегатные функции

Для каждого региона выведите общее количество проданных товаров и среднюю скидку.

Запрос 10: Использование конструкции CASE

Выведите отчет по сотрудникам: имя, должность, зарплата, а также категория зарплаты:

- "Высокая" если зарплата > 100000
- "Средняя" если зарплата между 60000 и 100000
- "Низкая" если зарплата < 60000
  Отсортируйте по убыванию зарплаты.

Дополнительные задания

Запрос 11: Функции для работы с текстом

Найдите сотрудников, у которых в адресе есть слово "ул." или "пр.", и выведите адрес с заменой этих сокращений на полные слова "улица" и "проспект".

Запрос 12: Математические функции для анализа

Рассчитайте коэффициент эффективности сотрудника: performance_score / years_experience. Выведите только тех, у кого коэффициент больше 15.

Запрос 13: Агрегатные функции с условиями

Подсчитайте общую сумму продаж для каждого сотрудника, но только для продаж со скидкой более 5%.

Запрос 14: Работа с датами и агрегация

Найдите проекты, которые длились более 100 дней, и выведите их длительность в днях.

Запрос 15: Комплексная агрегация

Создайте отчет по отделам: количество сотрудников, средняя зарплата, суммарный бюджет на бонусы, максимальный performance_score.

Справочник функций PostgreSQL

Строковые функции:

```
CONCAT(str1, str2) или || - объединение строк
LOWER(str), UPPER(str) - регистр
LENGTH(str) - длина строки
SUBSTRING(str FROM start FOR length) - подстрока
REPLACE(str, from, to) - замена подстроки
TRIM(str) - удаление пробелов
LEFT(str, n), RIGHT(str, n) - первые/последние n символов
```

Математические функции:

```
ROUND(value, decimals) - округление
CEIL(value), FLOOR(value) - округление вверх/вниз
ABS(value) - модуль
POWER(x, y) - возведение в степень
```

```
SQRT(value) - квадратный корень
MOD(x, y) - остаток от деления
```

Функции для работы с датами:

```
CURRENT_DATE - текущая дата
AGE(date1, date2) - разница между датами
EXTRACT(field FROM date) - извлечение части даты
DATE_PART('field', date) - аналогично EXTRACT
TO_CHAR(date, format) - форматирование даты
```

Агрегатные функции:

```
COUNT(*) - количество записей
SUM(column) - сумма значений
AVG(column) - среднее значение
MAX(column), MIN(column) - максимальное/минимальное значение
GROUP BY - группировка записей
HAVING - условие для групп
```

Контрольные вопросы

Теоретические вопросы:

1. Какие группы встроенных функций существуют в SQL? Приведите примеры каждой группы.
2. В чем разница между функциями `ROUND()`, `CEIL()` и `FLOOR()`?
3. Что такое агрегатные функции и чем они отличаются от скалярных функций?
4. Какие функции используются для работы с датами и временем в PostgreSQL?
5. Как с помощью строковых функций извлечь домен из email адреса?
6. Какая функция позволяет найти разницу между двумя датами в днях?
7. Как округлить число до ближайшего целого в большую сторону?
8. В чем разница между COUNT(*) и COUNT(column_name)?
9. Как работает предложение HAVING и чем оно отличается от WHERE?
10. Можно ли использовать несколько агрегатных функций в одном запросе?
11. Что произойдет, если использовать агрегатную функцию без GROUP BY?
12. Как вычислить медиану или моду набора значений с помощью SQL-функций?

Требования к отчету

1. **SQL-запросы** для всех 15 заданий с комментариями
2. **Результаты выполнения** (первые 10 строк каждого запроса)
3. **Объяснение** использованных функций и их параметров
4. **Ответы на контрольные вопросы** (письменно)