Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Вариант № 834929

**Лабораторная работа №2**

По дисциплине

Базы Данных

Выполнил студент группы P3121:

Фам Данг Чунг Нгиа

Преподаватель:

Карапетян Эрик Акопович

Санкт-Петербург 2024 г.

1. **Текст задания**

Составить запросы на языке SQL (пункты 1-7).

1. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
   Таблицы: Н\_ОЦЕНКИ, Н\_ВЕДОМОСТИ.  
   Вывести атрибуты: Н\_ОЦЕНКИ.ПРИМЕЧАНИЕ, Н\_ВЕДОМОСТИ.ЧЛВК\_ИД.  
   Фильтры (AND):  
   a) Н\_ОЦЕНКИ.ПРИМЕЧАНИЕ > хорошо.  
   b) Н\_ВЕДОМОСТИ.ИД < 1426978.  
   Вид соединения: INNER JOIN.
2. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
   Таблицы: Н\_ЛЮДИ, Н\_ВЕДОМОСТИ, Н\_СЕССИЯ.  
   Вывести атрибуты: Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ИД.  
   Фильтры (AND):  
   a) Н\_ЛЮДИ.ИМЯ < Александр.  
   b) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 142390.  
   c) Н\_СЕССИЯ.ИД > 14369.  
   Вид соединения: INNER JOIN.
3. Составить запрос, который ответит на вопрос, есть ли среди студентов вечерней формы обучения те, кто не имеет отчества.
4. В таблице Н\_ГРУППЫ\_ПЛАНОВ найти номера планов, по которым обучается (обучалось) ровно 2 групп на заочной форме обучения.  
   Для реализации использовать подзапрос.
5. Выведите таблицу со средними оценками студентов группы 4100 (Номер, ФИО, Ср\_оценка), у которых средняя оценка равна средней оценк(е|и) в группе 3100.
6. Получить список студентов, зачисленных ровно первого сентября 2012 года на первый курс заочной формы обучения. В результат включить:  
   номер группы;  
   номер, фамилию, имя и отчество студента;  
   номер и состояние пункта приказа;  
   Для реализации использовать подзапрос с EXISTS.
7. Вывести список студентов, имеющих одинаковые отчества, но не совпадающие ид.
8. **Реализацию запросов на SQL**

-- Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:

-- Таблицы: Н\_ОЦЕНКИ, Н\_ВЕДОМОСТИ.

-- Вывести атрибуты: Н\_ОЦЕНКИ.ПРИМЕЧАНИЕ, Н\_ВЕДОМОСТИ.ЧЛВК\_ИД.

-- Фильтры (AND):

-- a) Н\_ОЦЕНКИ.ПРИМЕЧАНИЕ > хорошо.

-- b) Н\_ВЕДОМОСТИ.ИД < 1426978.

-- Вид соединения: INNER JOIN.

SELECT "Н\_ОЦЕНКИ"."ПРИМЕЧАНИЕ", "Н\_ВЕДОМОСТИ"."ЧЛВК\_ИД"

FROM "Н\_ОЦЕНКИ"

INNER JOIN "Н\_ВЕДОМОСТИ" ON "Н\_ОЦЕНКИ"."КОД" = "Н\_ВЕДОМОСТИ"."ОЦЕНКА"

WHERE "Н\_ОЦЕНКИ"."ПРИМЕЧАНИЕ" > 'хорошо'

AND "Н\_ВЕДОМОСТИ"."ИД" < 1426978;

--------------------------------------------------------------------------------

-- Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:

-- Таблицы: Н\_ЛЮДИ, Н\_ВЕДОМОСТИ, Н\_СЕССИЯ.

-- Вывести атрибуты: Н\_ЛЮДИ.ФАМИЛИЯ, Н\_ВЕДОМОСТИ.ДАТА, Н\_СЕССИЯ.ИД.

-- Фильтры (AND):

-- a) Н\_ЛЮДИ.ИМЯ < Александр.

-- b) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 142390.

-- c) Н\_СЕССИЯ.ИД > 14369.

-- Вид соединения: INNER JOIN.

SELECT "Н\_ЛЮДИ"."ФАМИЛИЯ", "Н\_ВЕДОМОСТИ"."ДАТА", "Н\_СЕССИЯ"."ИД"

FROM "Н\_ЛЮДИ"

INNER JOIN "Н\_ВЕДОМОСТИ" ON "Н\_ЛЮДИ"."ИД" = "Н\_ВЕДОМОСТИ"."ЧЛВК\_ИД"

INNER JOIN "Н\_СЕССИЯ" ON "Н\_ЛЮДИ"."ИД" = "Н\_СЕССИЯ"."ЧЛВК\_ИД"

WHERE "Н\_ЛЮДИ"."ИМЯ" < 'Александр'

AND "Н\_ВЕДОМОСТИ"."ЧЛВК\_ИД" > 142390

AND "Н\_СЕССИЯ"."ИД" > 14369;

--------------------------------------------------------------------------------

--  Составить запрос, который ответит на вопрос, есть ли среди студентов вечерней формы обучения те, кто не имеет отчества.

SELECT DISTINCT 'ЕСТЬ' AS "ОТВЕТ"

FROM "Н\_УЧЕНИКИ"

INNER JOIN "Н\_ЛЮДИ" ON "Н\_УЧЕНИКИ"."ЧЛВК\_ИД" = "Н\_ЛЮДИ"."ИД"

INNER JOIN "Н\_ПЛАНЫ" ON "Н\_УЧЕНИКИ"."ПЛАН\_ИД" = "Н\_ПЛАНЫ"."ИД"

INNER JOIN "Н\_ФОРМЫ\_ОБУЧЕНИЯ" ON "Н\_ПЛАНЫ"."ФО\_ИД" = "Н\_ФОРМЫ\_ОБУЧЕНИЯ"."ИД"

WHERE "Н\_ФОРМЫ\_ОБУЧЕНИЯ"."ИМЯ\_В\_РОД\_ПАДЕЖЕ" = 'вечерней'

AND "Н\_ЛЮДИ"."ОТЧЕСТВО" = '.';

--------------------------------------------------------------------------------

-- В таблице Н\_ГРУППЫ\_ПЛАНОВ найти номера планов, по которым обучается (обучалось) ровно 2 групп на заочной форме обучения.

-- Для реализации использовать подзапрос.

SELECT "Н\_ПЛАНЫ"."НОМЕР"

FROM "Н\_ПЛАНЫ"

WHERE (

    SELECT COUNT(\*)

    FROM "Н\_ГРУППЫ\_ПЛАНОВ"

    WHERE "Н\_ПЛАНЫ"."ПЛАН\_ИД" = "Н\_ГРУППЫ\_ПЛАНОВ"."ПЛАН\_ИД"

    AND "Н\_ПЛАНЫ"."ПЛАН\_ИД" IN (

        SELECT "Н\_ПЛАНЫ"."ПЛАН\_ИД"

        FROM "Н\_ПЛАНЫ"

        INNER JOIN "Н\_ФОРМЫ\_ОБУЧЕНИЯ" ON "Н\_ПЛАНЫ"."ФО\_ИД" = "Н\_ФОРМЫ\_ОБУЧЕНИЯ"."ИД"

        WHERE "Н\_ФОРМЫ\_ОБУЧЕНИЯ"."ИМЯ\_В\_ПРЕД\_ПАДЕЖЕ" = 'заочной'

    )

) = 2;

--------------------------------------------------------------------------------

-- Выведите таблицу со средними оценками студентов группы 4100 (Номер, ФИО, Ср\_оценка), у которых средняя оценка равна средней оценк(е|и) в группе 3100.

SELECT

    "Номер",

    "ФИО",

    "Ср\_оценка"

FROM (

    SELECT

        ROW\_NUMBER() OVER (ORDER BY "Н\_ЛЮДИ"."ФАМИЛИЯ" ASC) AS "Номер",

        CONCAT("Н\_ЛЮДИ"."ФАМИЛИЯ", ' ', "Н\_ЛЮДИ"."ИМЯ", ' ', "Н\_ЛЮДИ"."ОТЧЕСТВО") AS "ФИО",

        ROUND(AVG(CAST("Н\_ВЕДОМОСТИ"."ОЦЕНКА" AS numeric)), 2) AS "Ср\_оценка"

    FROM

        "Н\_УЧЕНИКИ"

    INNER JOIN

        "Н\_ЛЮДИ" ON "Н\_ЛЮДИ"."ИД" = "Н\_УЧЕНИКИ"."ЧЛВК\_ИД"

    INNER JOIN

        "Н\_ВЕДОМОСТИ" ON "Н\_ВЕДОМОСТИ"."ЧЛВК\_ИД" = "Н\_УЧЕНИКИ"."ЧЛВК\_ИД"

    WHERE

        "Н\_УЧЕНИКИ"."ГРУППА" = '4100'

        AND "Н\_ВЕДОМОСТИ"."ОЦЕНКА" IN ('2', '3', '4', '5')

    GROUP BY

        "Н\_УЧЕНИКИ"."ЧЛВК\_ИД", "Н\_ЛЮДИ"."ФАМИЛИЯ", "Н\_ЛЮДИ"."ИМЯ", "Н\_ЛЮДИ"."ОТЧЕСТВО"

) AS sup

WHERE

    "Ср\_оценка" IN (

        SELECT ROUND(AVG(CAST("Н\_ВЕДОМОСТИ"."ОЦЕНКА" AS numeric)), 2)

        FROM "Н\_ВЕДОМОСТИ"

        INNER JOIN "Н\_УЧЕНИКИ" ON "Н\_ВЕДОМОСТИ"."ЧЛВК\_ИД" = "Н\_УЧЕНИКИ"."ЧЛВК\_ИД"

        WHERE "Н\_УЧЕНИКИ"."ГРУППА" = '3100'

        AND "Н\_ВЕДОМОСТИ"."ОЦЕНКА" IN ('2', '3', '4', '5')

    );

--------------------------------------------------------------------------------

-- Получить список студентов, зачисленных ровно первого сентября 2012 года на первый курс заочной формы обучения. В результат включить:

-- номер группы;

-- номер, фамилию, имя и отчество студента;

-- номер и состояние пункта приказа;

-- Для реализации использовать подзапрос с EXISTS.

SELECT

    "Н\_УЧЕНИКИ"."ГРУППА" AS "номер группы",

    CONCAT("Н\_ЛЮДИ"."ФАМИЛИЯ", ' ', "Н\_ЛЮДИ"."ИМЯ", ' ', "Н\_ЛЮДИ"."ОТЧЕСТВО") AS "ФИО",

    "Н\_УЧЕНИКИ"."СОСТОЯНИЕ"

FROM

    "Н\_УЧЕНИКИ"

INNER JOIN

    "Н\_ЛЮДИ" ON "Н\_ЛЮДИ"."ИД" = "Н\_УЧЕНИКИ"."ЧЛВК\_ИД"

INNER JOIN

    "Н\_ПЛАНЫ" ON "Н\_ПЛАНЫ"."ПЛАН\_ИД" = "Н\_УЧЕНИКИ"."ПЛАН\_ИД"

INNER JOIN

    "Н\_ФОРМЫ\_ОБУЧЕНИЯ" ON "Н\_ПЛАНЫ"."ФО\_ИД" = "Н\_ФОРМЫ\_ОБУЧЕНИЯ"."ИД"

    AND "Н\_ФОРМЫ\_ОБУЧЕНИЯ"."ИМЯ\_В\_РОД\_ПАДЕЖЕ" = 'заочной'

INNER JOIN

    "Н\_СЕССИЯ" ON "Н\_СЕССИЯ"."ЧЛВК\_ИД" = "Н\_ЛЮДИ"."ИД"

    AND "Н\_СЕССИЯ"."СЕМЕСТР" IN ('1', '2')

WHERE EXISTS (

    SELECT \*

    FROM "Н\_УЧЕНИКИ"

    WHERE DATE("Н\_УЧЕНИКИ"."НАЧАЛО") = '2012-09-01'

    AND "Н\_УЧЕНИКИ"."СОСТОЯНИЕ" = 'утвержден'

)

GROUP BY "Н\_УЧЕНИКИ"."ГРУППА", "Н\_ЛЮДИ"."ФАМИЛИЯ", "Н\_ЛЮДИ"."ИМЯ", "Н\_ЛЮДИ"."ОТЧЕСТВО", "Н\_УЧЕНИКИ"."СОСТОЯНИЕ";

--------------------------------------------------------------------------------

-- Вывести список студентов, имеющих одинаковые отчества, но не совпадающие ид.

SELECT DISTINCT

    "Н\_ЛЮДИ"."ОТЧЕСТВО",

    "Н\_ЛЮДИ"."ИМЯ",

    "Н\_ЛЮДИ"."ФАМИЛИЯ",

    "Н\_УЧЕНИКИ"."ИД"

FROM

    "Н\_УЧЕНИКИ"

INNER JOIN

    "Н\_ЛЮДИ" ON "Н\_УЧЕНИКИ"."ЧЛВК\_ИД" = "Н\_ЛЮДИ"."ИД"

WHERE

    "Н\_ЛЮДИ"."ОТЧЕСТВО" <> '.'

    AND "Н\_ЛЮДИ"."ИМЯ" <> '.'

    AND "Н\_ЛЮДИ"."ФАМИЛИЯ" <> '.'

    AND "Н\_ЛЮДИ"."ОТЧЕСТВО" IN (

        SELECT "Н\_ЛЮДИ"."ОТЧЕСТВО"

        FROM "Н\_ЛЮДИ"

        GROUP BY "Н\_ЛЮДИ"."ОТЧЕСТВО"

        HAVING COUNT(\*) >=2

    )

ORDER BY

    "Н\_ЛЮДИ"."ОТЧЕСТВО";

1. **Выводы по работе**

При выполнении лабораторной работы я познакомился с основными функциями языка SQL и диалекта PostgreSQL. Научился писать запросы, получать, агрегировать, отсеивать и сортировать полученные данные с использованием различных синтаксических конструкций языка. В результате был освоен язык DML SQL, предназначенный для работы с данными, хранящимися внутри базы данных.