МИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»  
(Самарский университет)   
  
  
Факультет информатики  
Кафедра программных систем  
  
Дисциплина  
**Базы данных  
  
  
  
ОТЧЕТ**по лабораторной работе №3  
  
Вариант №14

Студент: Михайлова К.В  
Группа: 6314-020302D  
  
Преподаватель: Повова-Коварцева Д. А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**Вариант № 14**

Предметная область: Бухгалтерия (расчет стипендии).

Основные предметно-значимые сущности: Студенты, Группы студентов, Результаты сдачи сессии.

Основные предметно-значимые атрибуты сущностей:

* студенты – фамилия, имя, отчество;
* группы студентов – название или номер группы;
* результаты сдачи сессии – студент, семестр, название категории (не сдал, сдал на 3, сдал на 4-5, сдал на 5).

Основные требования к функциям системы:

* назначить размер стипендии студентов за последнюю сессию в соответствие с действующими правилами;
* вывести группы, в которых студенты не получают стипендии;
* выбрать всех студентов, обучающихся по целевым договорам, сдавших на 3 или вообще не сдавших последнюю сессию, по группам;
* выбрать всех студентов, сдавших последнюю сессию на 5, по группам;
* подсчитать сумму стипендий студентов по курсам;
* вывести размер назначенной стипендии студентов, обучающихся по целевым договорам, по предприятиям;
* подсчитать сумму стипендий по группам.

Тема работы: обработка данных в реляционных БД с помощью структурированного языка запросов SQL.

Цель работы: приобретение навыков использования языка SQL. Выборка данных из таблиц, добавление, удаление, редактирование информации.

Содержание работы:

1. Изучение основ использования языка SQL: выборка данных из одной таблицы, выборка данных из связанных таблиц, сортировка и группировка результатов запроса, использование агрегатных функций и специальных предикатов.
2. Подготовка ответов на вопросы к лабораторной работе.
3. Подготовка отчета о проделанной работе.

Последовательность выполнения работы:

1. Придумайте и напишите SQL запросы, которые будут необходимы для предметной области (в соответствии с вариантом задания):
   1. Запрос на выборку избранных полей таблицы, с использованием синонима (алиаса) и сортировкой записей (ORDER BY).
   2. Запрос с использованием сортировки (ORDER BY) и группировки (GROUP BY).
   3. Запрос с использованием предложения DISTINCT.
   4. Запрос с использованием операций сравнения.
   5. Запросы для предикатов: IN, BETWEEN, LIKE, IS NULL.
   6. Запросы с использованием агрегатных функций (COUNT, SUM, AVG, MAX, MIN ), производящие обобщенную групповую обработку значений полей (используя ключевые фразы GROUP BY и HAVING).
   7. Запрос на выборку данных из двух связанных таблиц. Выбрать несколько полей, по которым сортируется вывод.
   8. Многотабличный запрос с использованием внутреннего и внешнего соединения.
   9. Многотабличный запрос с использованием оператора UNION.
2. Напишите SQL запросы, предусмотренной вариантом задания.
3. Создайте SQL команды для модификации данных (INSERT, UPDATE, DELETE).
4. Оформите отчет о выполнении лабораторной работы.

Выполнение работы:

1. На рисунке 1 представлен результат выполнения запроса с использованием алиаса, также продемонстрирована работа ORDER BY.

SELECT Название AS Категория, Размер\_стипендии AS Стипендия

FROM Бухгалтерия.dbo.Категория

ORDER BY Размер\_стипендии DESC;

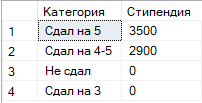


Рисунок 1 – пример выполнения

2. На рисунке 2 предоставлен результат выполнения запроса, который включает в себя объединение 3 таблиц( студент, результат сдачи, категория), продемонстрирована работа с GROUP BY и ORDER BY. Для сортировки от большего к меньшему произведен запрос с помощью агрегатной функции SUM.

SELECT Студент.ID\_студента AS Студент,SUM(Размер\_стипендии) AS Богатство

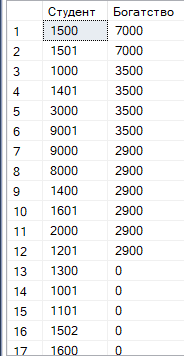
FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Студент.ID\_студента

ORDER BY Богатство DESC;

  
Рисунок 2 – пример выполнения

3. На рисунке 3 представлен результат работы запроса с использованием DISTINCT. Благодаря этому исключены повторы фамилий, повторяющиеся были выведены лишь один раз.

SELECT DISTINCT Фамилия

FROM Бухгалтерия.dbo.Студент

  
Рисунок 3 – пример выполнения

4. На рисунке 4 представлен результат работы с таблицей с использованием операции сравнения.

SELECT\* FROM Бухгалтерия.dbo.Студент

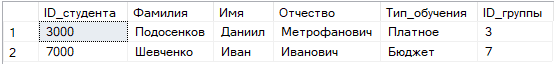
WHERE ID\_группы > 8

  
Рисунок 4 – пример выполнения

5. На рисунке 5 представлен результат работы с запросом с использованием предиката IN. Благодаря этому были выведены студенты с определенным ID группы.

SELECT\* FROM Бухгалтерия.dbo.Студент

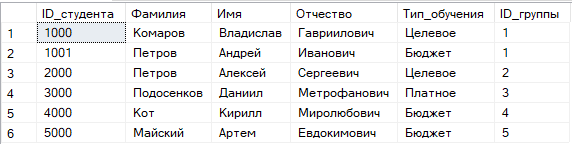
WHERE (ID\_группы) IN (3,7);

  
Рисунок 5 – пример выполнения

6. На рисунке 6 представлен результат работы с запросом с использованием предиката BETWEEN. Были выведены студенты ID группы которых, находится между 1 и 5.

SELECT\* FROM Бухгалтерия.dbo.Студент

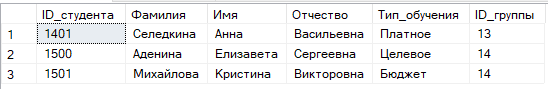
WHERE (ID\_группы) BETWEEN 1 AND 5;

  
Рисунок 6 – пример выполнения

7. На рисунке 7 представлен результат работы с запросом с использованием предиката LIKE. Были выведены студенты, отчество которых заканчивается на «на».

SELECT\* FROM Бухгалтерия.dbo.Студент

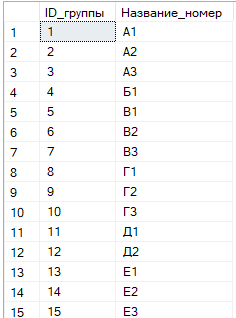
WHERE (Отчество) LIKE '%на' ;

  
Рисунок 7 – пример выполнения

8. На рисунке 8 представлен результат работы с запросом с использованием предиката IS NOT NULL. Были выведены строки, не содержащие NULL значения.

SELECT\* FROM Бухгалтерия.dbo.Группа

WHERE (ID\_группы) IS NOT NULL;

  
Рисунок 8 – пример выполнения

9. На рисунке 9 представлен результат выполнения запроса, с использованием агрегатной функции AVG, которая посчитала среднюю стипендию. Далее студенты были сгруппированы по фамилии и с помощью HAVING были выведены только те, средние значения, которые больше 1000. Для этого запроса были объединены 3 таблицы(студент, категория, результат сдачи).

SELECT Студент.Фамилия AS Студент,AVG(Размер\_стипендии) AS Средняя

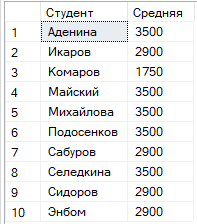
FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Студент.Фамилия

HAVING(AVG(Размер\_стипендии)>1000)

  
Рисунок 9 – пример выполнения

10. На рисунке 10 продемонстрирована работа с агрегатной функцией COUT, которая подсчитала сколько раз была получена стипендия студентами разных групп. С помощью GROUP BY студенты были сгруппированы по группам, а с помощью HAVING выведены группы, чьи студенты, получали стипендию более 1 раза.

SELECT Студент.ID\_группы AS Студент,COUNT(Размер\_стипендии) AS Получали

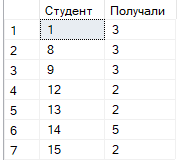
FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Студент.ID\_группы

HAVING(COUNT(Размер\_стипендии)>1)

  
Рисунок 10 – пример выполнения

11. На рисунке 11 продемонстрирована работа с агрегатной функцией MAX. Студенты сгруппированы по ID студента и выведены максимальные стипендии для определенных студентов.

SELECT Студент.ID\_студента AS Студент,MAX(Размер\_стипендии) AS Богатство

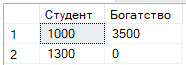
FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Студент.ID\_студента

HAVING Студент.ID\_студента IN ('1300','1000')

  
Рисунок 11 – пример выполнения

12. На рисунке 12 продемонстрирована работа с агрегатной функцией MIN. Студенты сгруппированы по ID студента и выведены минимальные стипендии для определенных студентов. Для этого были объединены три таблицы (студент, категория, результат сдачи) с помощью внешнего объединения.

SELECT Студент.ID\_студента AS Студент,MIN(Размер\_стипендии) AS Богатство

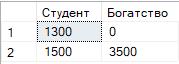
FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Студент.ID\_студента

HAVING Студент.ID\_студента IN ('1300','1500')

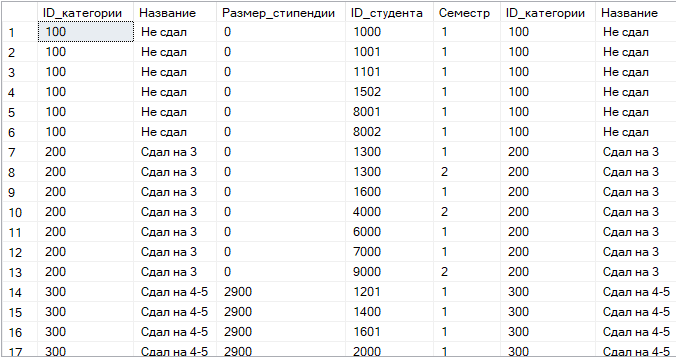
  
Рисунок 12 – пример выполнения

13. На рисунке 13 предоставлен результат выполнения внутреннего объединения двух таблиц из базы данных (категория, результат сдачи).

SELECT\*

FROM Бухгалтерия.dbo.Категория

FULL OUTER JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

  
Рисунок 13 – пример выполнения

14. На рисунке 14 предоставлен результат объединения двух таблиц с помощью UNION

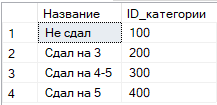
SELECT Название, ID\_категории

FROM Бухгалтерия.dbo.Результат\_сдачи

UNION

SELECT Название,ID\_категории

FROM Бухгалтерия.dbo.Категория

  
Рисунок 14 – пример выполнения

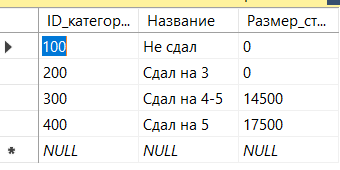
15. На рисунке 15 предоставлен запрос, который вносит новые данные в определенные таблицу.

  
Рисунок 15 – запрос на выполнение

16. На рисунке 16 предоставлен результат выполнения изменения с помощью UPDATE. Здесь была изменена стипендия, её увеличили в 5 раз.

UPDATE Бухгалтерия.dbo.Категория

SET Размер\_стипендии=Размер\_стипендии\*5

  
Рисунок 16 – пример выполнения

17. На рисунке 17 представлен результат выполнения DELETE. Из таблицы студент, был удален студент с ID равным 1000

DELETE FROM Бухгалтерия.dbo.Студент

WHERE ID\_студента='1000'

  
Рисунок 17 – результат выполнения

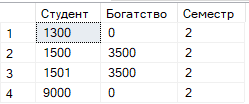
18. На рисунке 18 пример выполнения запроса, при котором назначается размер стипендии студентов за последнюю сессию.  
SELECT Студент.ID\_студента AS Студент, Категория.Размер\_стипендии AS Богатство, Результат\_сдачи.Семестр

FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

WHERE Семестр>1

  
Рисунок 18 – результат выполнения

19. На рисунке 19 пример выполнения того, как с помощью запроса можно вывести группы, в которых студенты не получают стипендию.

SELECT Студент.ID\_студента AS Студент, Категория.Размер\_стипендии AS Богатство, Результат\_сдачи.Семестр, Студент.ID\_группы AS Группа

FROM Бухгалтерия.dbo.Категория

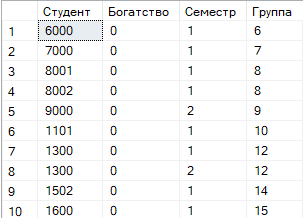
JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

JOIN Бухгалтерия.dbo.Группа ON Группа.ID\_группы = Студент.ID\_группы

WHERE Категория.Размер\_стипендии=0

ORDER BY Студент.ID\_группы

  
Рисунок 19 – результат выполнения

20. На рисунке 20 пример выбраны все студенты, обучающиеся на целевому договору и сдавшие на 3 или не сдавшие последнюю сессию, по группам.

SELECT Группа.ID\_группы, Категория.Размер\_стипендии AS Получает, Студент.Тип\_обучения AS Обучается, Результат\_сдачи.Семестр AS Семестр

FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

JOIN Бухгалтерия.dbo.Группа ON Группа.ID\_группы = Студент.ID\_группы

WHERE Тип\_обучения='Целевое' AND Категория.ID\_категории IN (100,200) AND Семестр=2

ORDER BY Группа.ID\_группы

  
Рисунок 20 – результат выполнения

21. На рисунке 21 пример выполнения работы, при которой были выбраны студенты, которые сдали последнюю сессию на 5, по группам.

SELECT Студент.ID\_студента AS Студент, Категория.Размер\_стипендии AS Богатство, Результат\_сдачи.Семестр, Студент.ID\_группы AS Группа

FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

JOIN Бухгалтерия.dbo.Группа ON Группа.ID\_группы = Студент.ID\_группы

WHERE Категория.Размер\_стипендии=3500 AND Результат\_сдачи.Семестр=2

ORDER BY Студент.ID\_группы

  
Рисунок 21 – результат выполнения

22. На рисунке 22 выведены размеры стипендий студентов, обучающихся по целевым договорам.

SELECT Группа.ID\_группы, Категория.Размер\_стипендии AS Получает, Студент.Тип\_обучения AS Обучается, Результат\_сдачи.Семестр AS Семестр

FROM Бухгалтерия.dbo.Категория

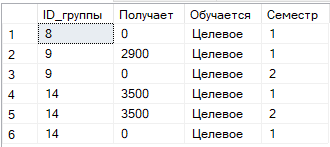
JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

JOIN Бухгалтерия.dbo.Группа ON Группа.ID\_группы = Студент.ID\_группы

WHERE Тип\_обучения='Целевое'

ORDER BY Группа.ID\_группы

  
Рисунок 22 – результат выполнения

23. На рисунке 23 пример того, как посчитать сумму стипендий студентов по группам.

SELECT Студент.ID\_группы AS Студент,SUM(Размер\_стипендии) AS Богатство

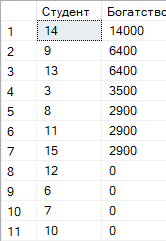
FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Студент.ID\_группы

ORDER BY Богатство DESC;

  
Рисунок 23 – результат выполнения

24. На рисунке 23 пример того, как посчитать сумму стипендий студентов по семестрам.

SELECT Результат\_сдачи.Семестр AS Семестр,SUM(Размер\_стипендии) AS Богатство

FROM Бухгалтерия.dbo.Категория

JOIN Бухгалтерия.dbo.Результат\_сдачи ON Категория.ID\_категории=Результат\_сдачи.ID\_категории

JOIN Бухгалтерия.dbo.Студент ON Результат\_сдачи.ID\_студента = Студент.ID\_студента

GROUP BY Результат\_сдачи.Семестр

ORDER BY Богатство DESC;

  
Рисунок 24 – результат выполнения