**Лабораторная работа № 6. Многотабличные SELECT-запросы**

|  |  |
| --- | --- |
| 1.На основе таблиц **AUDITORIUM\_ TYPE** и **AUDITORIUM** сформировать перечень кодов аудиторий (столбец **AUDITORUM.AUDITORIUM**) и соответствующих им наименований типов аудиторий (столбец **AUDITORIUM\_ TYPE.AUDITORIUM\_ TYPENAME**). Примечание: использовать соединение таблиц INNER JOIN. | SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME  FROM AUDITORIUM  INNER JOIN  AUDITORIUM\_TYPE  ON AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE; |
| 2.На основе таблиц **AUDITORIUM\_TYPE** и **AUDITORIUM** сформировать перечень кодов аудиторий (столбец **AUDITORIUM.AUDITORIUM**) и соответствующих им наименований типов аудиторий (столбец **AUDITORIUM\_ TYPE.AUDITO-RIUM\_TYPENAME**).  При этом следует выбрать только те аудитории, в наименовании которых присутствует подстрока **компьютер**.  Примечание: использовать соединение таблиц INNER JOIN и предикат LIKE. | SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME  FROM AUDITORIUM  INNER JOIN  AUDITORIUM\_TYPE  ON AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE  WHERE AUDITORIUM\_TYPENAME LIKE '%компьютер%'; |
| 3.Написать два SELECT-запроса, формирующих результирующие наборы аналогичные запросам из заданий 1 и 2, но без применения INNER JOIN. | SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME  FROM AUDITORIUM, AUDITORIUM\_TYPE  WHERE AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE;  SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME  FROM AUDITORIUM, AUDITORIUM\_TYPE  WHERE AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE  AND AUDITORIUM\_TYPENAME LIKE '%компьютер%'; |
| 4.На основе таблиц **PRORGESS**, **STUDENT**, **GROUPS**, **SUBJECT**, **PULPIT** и **FACULTY** сформировать перечень студентов, получивших экзаменационные оценки (столбец **PROGRESS.NOTE**) от 6 до 8.  Результирующий набор должен содержать столбцы: **Факультет**, **Кафедра**, **Специальность**, **Дисциплина**, **Имя Студента**, **Оценка**. В столбце **Оценка** должны быть записаны экзаменационные оценки прописью: **шесть**, **семь**, **восемь**. | SELECT FACULTY.FACULTY\_NAME [Факультет], PULPIT.PULPIT\_NAME [Кафедра],  PROFESSION.PROFESSION\_NAME [Специальность], PROGRESS.SUBJECT [Дисциплина],  STUDENT.NAME [Имя студента], PROGRESS.NOTE [Оценка],  CASE  WHEN (PROGRESS.NOTE = 6) then 'шесть'  WHEN (PROGRESS.NOTE = 7) then 'семь'  WHEN (PROGRESS.NOTE = 8) then 'восемь'  END [ОценкаС]  FROM  PROGRESS  INNER JOIN STUDENT  ON PROGRESS.IDSTUDENT = STUDENT.IDSTUDENT,  FACULTY  INNER JOIN PULPIT  ON PULPIT.FACULTY = FACULTY.FACULTY  INNER JOIN PROFESSION  ON FACULTY.FACULTY = PROFESSION.FACULTY  WHERE PROGRESS.NOTE BETWEEN 6 AND 8  ORDER BY FACULTY.FACULTY,  PULPIT.PULPIT,  PROFESSION.PROFESSION,  STUDENT.NAME ASC,  PROGRESS.NOTE DESC; |
| 5.Переписать запрос, реализующий задание 4 таким образом, чтобы в результирующем наборе сортировка по экзаменационным оценкам была следующей: сначала выводились строки с оценкой **7**, затем строки с оценкой **8** и далее строки с оценкой **6**.  Примечание: использовать выражение CASE в секции ORDER BY. | SELECT FACULTY.FACULTY\_NAME [Факультет], PULPIT.PULPIT\_NAME [Кафедра],  PROFESSION.PROFESSION\_NAME [Специальность], PROGRESS.SUBJECT [Дисциплина],  STUDENT.NAME [Имя студента], PROGRESS.NOTE [Оценка],  CASE  WHEN (PROGRESS.NOTE = 6) then 'шесть'  WHEN (PROGRESS.NOTE = 7) then 'семь'  WHEN (PROGRESS.NOTE = 8) then 'восемь'  END [ОценкаС]  FROM  PROGRESS  INNER JOIN STUDENT  ON PROGRESS.IDSTUDENT = STUDENT.IDSTUDENT,  FACULTY  INNER JOIN PULPIT  ON PULPIT.FACULTY = FACULTY.FACULTY  INNER JOIN PROFESSION  ON FACULTY.FACULTY = PROFESSION.FACULTY  WHERE PROGRESS.NOTE BETWEEN 6 AND 8  ORDER BY (CASE  WHEN (PROGRESS.NOTE = 6) THEN 3  WHEN (PROGRESS.NOTE = 8) THEN 2  WHEN (PROGRESS.NOTE = 7) THEN 1  END  ); |
| 6.На основе таблиц **PULPIT** и **TEACHER** получить полный перечень кафедр (столбец **PULPIT.PULPIT\_ NAME)** и преподавателей (столбец **TEACHER.TEA-CHER\_NAME**) на этих кафедрах. Результирующий набор должен содержать два столбца: **Кафедра** и **Преподаватель**. Если на кафедре нет преподавателей, то в столбце **Преподаватель** должна быть выведена строка **\*\*\*.**  Примечание: использовать соединение таблиц LEFT OUTER JOIN и функцию **isnull**. | SELECT PULPIT.PULPIT\_NAME [Кафедра], ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [Преподаватель]  FROM PULPIT  LEFT OUTER JOIN TEACHER  ON PULPIT.PULPIT = TEACHER.PULPIT; |
| **7.** В запросе, реализующем пункт 6, поменять порядок таблиц в выражении LEFT OUTER JOIN. Объяснить: почему в столбце **Кафедра** не может быть значения NULL.  Переписать запрос таким образом, чтобы получился аналогичный результат, но применялось соединение таблиц RIGHT OUTER JOIN. | SELECT PULPIT.PULPIT\_NAME [Кафедра], ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [Преподаватель]  FROM TEACHER  LEFT OUTER JOIN PULPIT  ON PULPIT.PULPIT = TEACHER.PULPIT;  SELECT PULPIT.PULPIT\_NAME [Кафедра], ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [Преподаватель]  FROM PULPIT  RIGHT OUTER JOIN TEACHER  ON PULPIT.PULPIT = TEACHER.PULPIT; |
| 8.Показать на примере, что соединение FULL OUTER JOIN двух таблиц:  − является коммутативной операцией;  − является объединением LEFT OUTER JOIN и RIGHT OUTER JOIN соединений этих таблиц;  − включает соединение INNER JOIN этих таблиц.  Примечание: создать две таблицы, заполнить их данными. Разработать SELECT-запросы, реализующие задания.  Создать три новых запроса:  − запрос, результат которого содержит данные левой (в операции FULL OUTER JOIN) таблицы и не содержит данные правой;  − запрос, результат которого содержит данные правой таблицы и не содержащие данные левой;  − запрос, результат которого содержит данные правой таблицы и левой таблиц;  Примечание: использовать в запросах выражение IS NULL и IS NOT NULL. | CREATE DATABASE TEST;  USE TEST;  CREATE TABLE T1  (  T1F1 NVARCHAR(50),  T1F2 INT  );  CREATE TABLE T2  (  T2F1 NVARCHAR(50),  T2F2 INT  );  INSERT INTO T1  VALUES ('111', 111),  ('222', 222),  ('333', 333),  ('444', 444),  ('555', 555);  INSERT INTO T2  VALUES ('111', 111),  ('222', 22),  ('333', 33),  ('44', 444),  ('555', 55);  SELECT \* FROM T1  FULL OUTER JOIN T2  ON T1F1 = T2F1;    SELECT \* FROM T2  FULL OUTER JOIN T1  ON T1F1 = T2F1;    SELECT \* FROM T1  FULL OUTER JOIN T2  ON T1F2 = T2F2;  SELECT \* FROM T1  LEFT OUTER JOIN T2  ON T1F2 = T2F2;  SELECT \* FROM T1  RIGHT OUTER JOIN T2  ON T1F2 = T2F2;  SELECT \* FROM T1  INNER JOIN T2  ON T1F2 = T2F2;    SELECT T1F1, T1F2  FROM T1  FULL OUTER JOIN T2  ON T1F2 = T2F2;  SELECT T2F1, T2F2  FROM T1  FULL OUTER JOIN T2  ON T1F2 = T2F2;  SELECT \*  FROM T1  FULL OUTER JOIN T2  ON T1F2 = T2F2;    SELECT ISNULL(T1F1, '\*\*\*'), ISNULL(T1F2, '\*\*\*') … |
| 9.Разработать SELECT-запрос на основе CROSS JOIN-соединения таблиц **AUDITORIUM\_TYPE** и **AUDITORIUM**, формирующего результат, аналогичный результату, полученному при выполнении запроса в задании 1. | USE Stash\_UNIVER;  SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME  FROM AUDITORIUM  CROSS JOIN  AUDITORIUM\_TYPE  WHERE AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE; |
| 10. Разработать и выполнить аналогичные запросы для базы данных **X\_MyBASE**. | USE STASH\_MyBase;  SELECT Студент.Фамилия, Оценка.Оценка  FROM Студент  INNER JOIN  Оценка  ON Студент.ID\_Student = Оценка.ID\_Student;  USE STASH\_MyBase;  SELECT Студент.Фамилия, Оценка.Оценка  FROM Студент  INNER JOIN  Оценка  ON Студент.ID\_Student = Оценка.ID\_Student  WHERE Фамилия LIKE 'A%';  SELECT Студент.Фамилия, Оценка.Оценка, Предмет.Объём\_ЛР  FROM Студент  INNER JOIN  Оценка  ON Студент.ID\_Student = Оценка.ID\_Student  INNER JOIN  Предмет  ON Оценка.ID\_Subject = Предмет.ID\_Subject;  SELECT Студент.Фамилия, Оценка.Оценка, Предмет.Объём\_ЛР,  CASE  WHEN Объём\_ЛР > 10 THEN 'сложно'  WHEN Объём\_ЛР < 10 THEN 'изи'  ELSE 'OK'  END [LVL]  FROM Студент  INNER JOIN  Оценка  ON Студент.ID\_Student = Оценка.ID\_Student  INNER JOIN  Предмет  ON Оценка.ID\_Subject = Предмет.ID\_Subject  ORDER BY Студент.Фамилия;  SELECT Предмет.Название\_предмета, ISNULL(Оценка.Оценка, 0)  FROM Предмет  LEFT OUTER JOIN Оценка  ON Предмет.ID\_Subject = Оценка.ID\_Subject; |