|  |  |  |
| --- | --- | --- |
| Julie Sweany | | |
|  | Your Answers | |
| **1** | Result Grid |  | |
| SQL Text | -- Unit 9.1 - Julie Sweany  -- List students whose scholarship amount is less than the average scholarship amount awarded to students at this school (but greater than zero). Sort by last name.  USE College;  SELECT  LastName AS Last,  FirstName AS First,  Major.Name AS Major,  FORMAT(Scholarship, 2) AS Scholarship  FROM  Student  JOIN  Major ON Major.ID = MajorID  WHERE  Scholarship > 0  AND Scholarship < (SELECT  AVG(Scholarship)  FROM  Student  WHERE  Scholarship > 0)  ORDER BY LastName; | |
| **2** | Result Grid |  | |
| SQL Text | -- 9.2 - Julie Sweany  -- Output last and first names and indicate student vs. faculty in a column named Type for those in the Nursing department. Order by last name.  USE College;  SELECT  LastName AS Last, FirstName AS First, 'Student' AS Type  FROM  Student  WHERE  MajorID = 5  UNION SELECT  LastName AS Last, FirstName AS First, 'Faculty' AS Type  FROM  Faculty  WHERE  DepartmentID = 5  ORDER BY Last; | |
| **3** | Scholarships view | | | |
| Screen shot of the result grid from SELECT \* FROM StudentScholarships | | | |
| Screen shot of your Schema window showing the **scholarships** view with its columns expanded | | | |
| Text of your CREATE VIEW SQL  -- 9.3 - Julie Sweany  -- Create and use view to output list of all students receiving scholarships.  USE College;  CREATE OR REPLACE VIEW StudentScholarships AS  SELECT  LastName, FirstName, Email, Scholarship  FROM  Student  WHERE  Scholarship > 0; | | | |
| **4** | Student Flatten View | | | |
| Screen shot of the result grid from SELECT \* FROM StudentFlatten | | | |
| Screen shot of your Schema window showing the view with its columns expanded | | | |
| Text of your CREATE VIEW SQL  -- 9.4 - Julie Sweany  -- Create view to display ID, Name, Major, Email, Sex Date of Birth, Enrollment Date, Scholarship, and Major ID for all students. Order by last name.  USE College;  CREATE OR REPLACE VIEW StudentFlatten AS  SELECT  Student.ID,  CONCAT(LastName, ', ', FirstName) AS Name,  Major.Name AS Major,  Email,  Sex,  DateOfBirth AS 'Date of Birth',  EnrolledDate AS Enrolled,  Scholarship,  MajorID  FROM  Student  LEFT OUTER JOIN  Major ON MajorID = Major.ID  ORDER BY Name; | | | |
|  | Discussion of how this view might be useful  The view is useful because it can save the time involved with joining the two tables (to get the name of the major’s ID) and concatenating the students’ names. It’s likely that these things would be called for fairly often, so it makes sense to create the view so that the work does not need to be redone. | | | |
| **5** | Nursing view | | | |
| Screen shot of the result grid Nursing Students with a scholarship | | | |
| Screen shot of your Schema window showing the view with its columns expanded | | | |
| Text of your CREATE VIEW SQL  -- 9.5 - Julie Sweany  -- Create a view that is based on the StudentFlatten view that returns only students majoring in nursing.  USE College;  CREATE OR REPLACE VIEW NursingStudents AS  SELECT  \*  FROM  studentflatten  WHERE  MajorID = 5; | | | |
|  | Discussion of why you shouldn’t select on the name of the major.  To select on the name of the major, you need to add single quotes around the word major (‘major’) or else an error results. It is better and less error-prone to work with the simpler MajorID number. It is probably also more efficient to use the WHERE clause in the Student table rather than in the joined Major table. | | | |

|  |  |
| --- | --- |
| **6** | Do It Yourself View |
| Screen shot of the result grid from SELECT \* FROM your view |
| Brief explanation of the purpose of your view. Tell me what it’s for, not how it works.  The view outputs all departments (regardless of whether they have faculty, courses, or sections associated with them) and lists the department name, course name, section, semester, instructor and instructor email address. Results are sorted by department, and then by course. |
| Text of your CREATE VIEW SQL  -- 9.6 - Julie Sweany  -- Output all Departments, even those without faculty, courses, or sections, and list department names, course names, sections, semester, instructor, and instructors' email addresses. Sort by Department, then by Course.  USE College;  CREATE OR REPLACE VIEW AllDepartmentCourses AS  SELECT  Department.Name AS Department,  Course.Name AS Course,  Section.Name AS Section,  Semester.Name AS Semester,  CONCAT(LastName, ', ', FirstName) AS Instructor,  Email AS 'Instructor\'s Email'  FROM  Faculty  RIGHT JOIN  Department ON Department.ID = Faculty.DepartmentID  LEFT JOIN  Course ON Department.ID = Course.DepartmentID  LEFT JOIN  Section ON CourseID = Course.ID  LEFT JOIN  Semester ON SemesterID = Semester.ID  ORDER BY Department , Course; |