**Using MSSQL to answer the following seven (17-23) practical SQL questions using the same university data from Assignment 1 and 2. You will need to convert the MySQL Data to Microsoft SQL to the questions. You may use SQLLINES.COM at** [**http://www.sqlines.com/online**](http://www.sqlines.com/online)**.**

# Problem #14 – Aggregates that are grouped and subsetted (using a GROUP BY clause and a HAVING clause)

Retrieve the class name, minimum GPA, maximum GPA, average GPA, and average GPA plus 10% for each class but only for classes with an average GPA less than 3.5.

|  |
| --- |
| **SELECT** C.CrsDesc **AS** Class, **MIN**(E.EnrGrade) **AS** MINGPA, **MAX**(E.EnrGrade) **AS** MAXGPA, **AVG**(E.EnrGrade) **AS** AVGGPA, **AVG**(E.EnrGrade)\*1.1 **AS** GPA10  **FROM** Enrollment **AS** E, Course **AS** C, Offering **AS** O  **WHERE** O.CourseNo = C.CourseNo  **AND** E.OfferNo = O.OfferNo  **GROUP** **BY** C.CrsDesc  **HAVING** **AVG**(E.EnrGrade) < 3.5  **ORDER** **BY** **AVG**(E.EnrGrade); |

# Problem #15 – Aggregates of a subset of rows that are grouped and subsetted (using a WHERE clause, a GROUP BY clause, and a HAVING clause)

Retrieve the class name, minimum GPA, maximum GPA, average GPA, and average GPA plus 10% for each class but only for non-IS majors and only for classes with an average GPA greater than 3 for non-IS majors.

|  |
| --- |
| **SELECT** C.CrsDesc **AS** Class, **MIN**(E.EnrGrade) **AS** MINGPA, **MAX**(E.EnrGrade) **AS** MAXGPA, **AVG**(E.EnrGrade) **AS** AVGGPA, **AVG**(E.EnrGrade)\*1.1 **AS** GPA10  **FROM** Enrollment **AS** E, Offering **AS** O, Course **AS** C, Student **AS** S  **WHERE** StdMajor != 'IS'  **AND** O.OfferNo = E.OfferNo  **AND** E.StdNo = S.StdNo  **AND** O.CourseNo = C.CourseNo  **GROUP** **BY** C.CrsDesc  **HAVING** **AVG**(E.EnrGrade) > 3; |

# Problem #16 – Cartesian Products, how many rows expected

Perform a Cartesian Product between tables Student, Offering, Enrollment, Course, and Faculty How many columns are expected?

How many rows are expected?

|  |
| --- |
| 34 columns - add all the columns from each table  222, 222 rows - multiply the number of rows from each table  **SELECT** \* **FROM** Student, Offering, Enrollment, Course, Faculty; |

# Problem #17 – Cartesian Products, figuring out which rows match

Perform a Cartesian Product between tables Student, Offering, Enrollment, Course, and Faculty

Retrieve only the columns which are needed to show matching based on the relationship between the five tables and order in such a way as to tell the matching records.

|  |
| --- |
| **SELECT** C.CourseNo, F.FacNo, O.CourseNo, O.FacNo, S.StdNo, E.StdNo **FROM** Student **AS** S, Offering **AS** O, Enrollment **AS** E, Course **AS** C, Faculty **AS** F  **ORDER** **BY** E.StdNo; |

# Problem #18 – Turning a Cartesian Product into an Inner Join by adding a WHERE clause to the Cross Product Syntax

Start with a Cartesian Product between tables Student, Offering, Enrollment, Course, and Faculty

Retrieve only the columns which are needed to show matching based on the relationship between the five tables and order in such a way as to tell the matching records

Add a WHERE clause to turn the Cartesian Product into an Inner Join.

|  |
| --- |
| **SELECT** C.CourseNo, F.FacNo, O.CourseNo, O.FacNo, S.StdNo, E.StdNo, E.OfferNo, O.OfferNo **FROM** Student **AS** S, Offering **AS** O, Enrollment **AS** E, Course **AS** C, Faculty **AS** F  **WHERE** O.CourseNo = C.CourseNo  **AND** E.StdNo = S.StdNo  **AND** F.FacNo = O.FacNo  **AND** E.OfferNo = O.OfferNo  **ORDER** **BY** E.StdNo; |

# Problem #19 – Converting an Inner Join from Cross Product Syntax to Join Operator Syntax

Start with the Inner Join using Cross Product Syntax for the tables: Student, Offering, Enrollment, Course, and Faculty Convert to Join Operator Syntax.

|  |
| --- |
| **SELECT** C.CourseNo, F.FacNo, O.CourseNo, O.FacNo, S.StdNo, E.StdNo, E.OfferNo, O.OfferNo  **FROM** Course **AS** C  **INNER** **JOIN** Offering **AS** O  **ON** O.CourseNo = C.CourseNo  **INNER** **JOIN** Faculty **AS** F  **ON** F.FacNo = O.FacNo  **INNER** **JOIN** Enrollment **AS** E  **ON** E.OfferNo = O.OfferNo  **INNER** **JOIN** Student **AS** S  **ON** S.StdNo = E.StdNo; |

# Problem #20 – Combining Inner Join and WHERE, GROUP BY, and HAVING clauses

List the course number, offer number, and average grade of students enrolled in fall 2010 IS course offerings in which more than one student is enrolled.

|  |
| --- |
| **SELECT** E.OfferNo, O.CourseNo, **AVG**(E.EnrGrade) **AS** AvgGrade  **FROM** Offering **AS** O  **INNER** **JOIN** Enrollment **AS** E  **ON** O.OfferNo = E.OfferNo  **WHERE** O.OffTerm = 'FALL' **AND** O.OffYear = 2010 **AND** O.CourseNo **LIKE** 'IS%'  **GROUP** **BY** E.OfferNo, E.OfferNo, O.CourseNo  **HAVING** E.OfferNo > 1; |