# LAB – Simple SQL statements

**Question 1:**

Write SQL statements to create a database containing the following tables. Note: need to add appropriate primary and foreign keys.

1. **Table: Departments**

**Name Type Size**

DeptID varchar 4

Name Nvarchar 50

NoOfStudents int

**Constraint:** Name - not null.

1. **Table: Students**

**Name Type Size**

StudentID varchar 4

LastName Nvarchar 30

FirstName Nvarchar 10

Sex varchar 1

DateOfBirth Date

PlaceOfBirth Nvarchar 30

DeptID Varchar 4

Scholarship float

AverageScore Numeric(4,2)

Constraint: Sex should be ‘F’ or ‘M’

1. **Table: Courses**

**Name Type Size**

CourseID varchar 4

Name Nvarchar 35

Credits tinyint

1. **Table: Results**

**Name Type Size**

StudentID varchar 4

CourseID varchar 4

Year int

Semester int

Mark float 1

Grade varchar 6

Write SQL statements to insert data to database as follow:

Table Departments:

|  |  |  |
| --- | --- | --- |
| **DeptID** | **Name** | **NoOfStudents** |
| IS | Information Systems |  |
| NC | Network and Communication |  |
| SE | Software Engineering |  |
| CE | Computer Engineering |  |
| CS | Computer Science |  |

Table Students:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **StudentID** | **LastName** | **FirstName** | **Sex** | **DateOfBirth** | **PlaceOfBirth** | **DeptID** | **Scholarship** | **AverageScore** |
| S001 | Lê | Kim Lan | F | 23/02/1990 | Hà nội | IS | 130000 |  |
| S002 | Trần | Minh Chánh | M | 24/12/1992 | Bình Định | NC | 150000 |  |
| S003 | Lê | An Tuyết | F | 21/02/1991 | Hải phòng | IS | 170000 |  |
| S004 | Trần | Anh Tuấn | M | 20/12/1993 | TpHCM | NC | 80000 |  |
| S005 | Trần | Thị Mai | F | 12/08/1991 | TpHCM | SE | 0 |  |
| S006 | Lê | Thị Thu Thủy | F | 02/01/1991 | An Giang | IS | 0 |  |
| S007 | Nguyễn | Kim Thư | F | 02/02/1990 | Hà Nội | SE | 180000 |  |
| S008 | Lê | Văn Long | M | 08/12/1992 | TpHCM | IS | 190000 |  |

Table Courses:

|  |  |  |
| --- | --- | --- |
| **CourseID** | **Name** | **Credits** |
| DS01 | Database Systems | 3 |
| AI01 | Artificial Intelligence | 3 |
| CN01 | Computer Network | 3 |
| CG01 | Computer Graphics | 4 |
| DSA1 | Data Structures and Algorithms | 4 |

Table Results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **StudentID** | **CourseID** | **Year** | **Semester** | **Mark** | **Grade** |
| S001 | DS01 | 2017 | 1 | 3 |  |
| S001 | DS01 | 2017 | 2 | 6 |  |
| S001 | AI01 | 2017 | 1 | 4.5 |  |
| S001 | AI01 | 2017 | 2 | 6 |  |
| S001 | CN01 | 2017 | 3 | 5 |  |
| S002 | DS01 | 2016 | 1 | 4.5 |  |
| S002 | DS01 | 2017 | 1 | 7 |  |
| S002 | CN01 | 2016 | 3 | 10 |  |
| S002 | DSA1 | 2016 | 3 | 9 |  |
| S003 | DS01 | 2017 | 1 | 2 |  |
| S003 | DS01 | 2017 | 3 | 5 |  |
| S003 | CN01 | 2017 | 2 | 2.5 |  |
| S003 | CN01 | 2017 | 3 | 4 |  |
| S004 | DS01 | 2017 | 3 | 4.5 |  |
| S004 | DSA1 | 2018 | 1 | 10 |  |
| S005 | DS01 | 2017 | 2 | 7 |  |
| S005 | CN01 | 2017 | 2 | 2.5 |  |
| S005 | CN01 | 2018 | 1 | 5 |  |
| S006 | AI01 | 2018 | 1 | 6 |  |
| S006 | CN01 | 2018 | 2 | 10 |  |

Write SQL statements to do bellow task.

1. Update NoOfStudents of each department in Departments table where NoOfStudents is the total number of students of each departments. Note that for department that has no student, the NoOfStudents should be 0.
2. Update AverageScore for each student so that for each course, we take only his/her highest Mark and the AverageScore of the student is calculated as the average mark of all the courses that the student joins.
3. Update Grade in table Results so that:

* Grade = ‘Passed’ if 5<= Mark <= 10
* Grade = ‘Failed’ if 0<= Mark < 5

1. List (StudentID, Fullname, DateOfBirth, PlaceOfBirth, DeptID, Scholarship) of all students having Scholarship not greater than 160000, in descending order of Scholarship. Note that FullName is the concatenation of LastName and FirstName. For example, if LastName = ‘Lê’ and FirstName = ‘Kim Lan’, then Fullname should be ‘Kim Lan Lê’.
2. List (DeptID, DepartmentName, StudentID, LastName, FirstName) of all departments (KHOA) so that we see also departments which have no students.
3. List (StudentID, LastName, FirstName, NumberOfCourses) of all students, show the results in ascending order of NumberOfCourses where NumberOfCourses is the total number of courses studied by each student.
4. List (DeptID, DepartmentName, NumberOfFemaleStudents, NumberOfMaleStudents) of all departments.
5. Show the list of students which are not in the department ‘Information Systems’ but having Mark of Database Systems greater than at least one student of department ‘Information Systems’.
6. List (CourseID, CourseName, BestStudentFullName) where BestStudentFullName is the name of the student who has the highest mark for this course.