Trường Đại Học FPT

Lớp IA1604

Student Grading Management Sub-System database

DBI202-ASSIGNMENT

Sinh viên: Châu Ngọc Đức

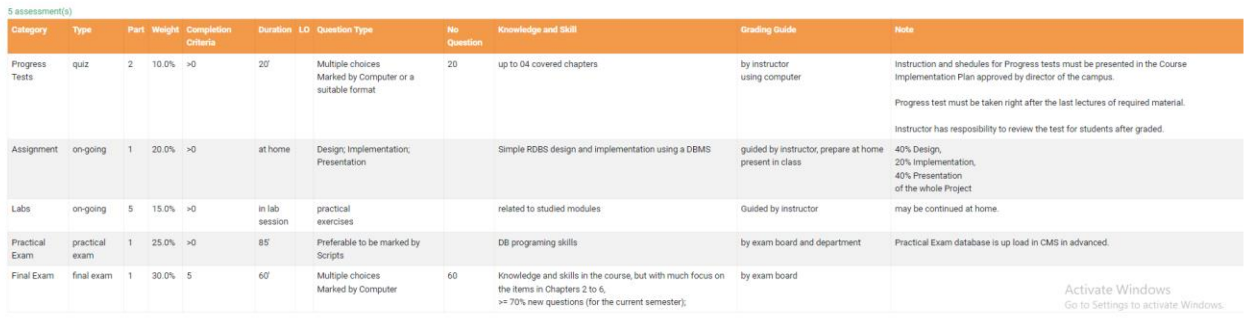
Mssv: HE161106

Lớp: IA1604

Student Grading Management Sub-System

DBI202-Assignment

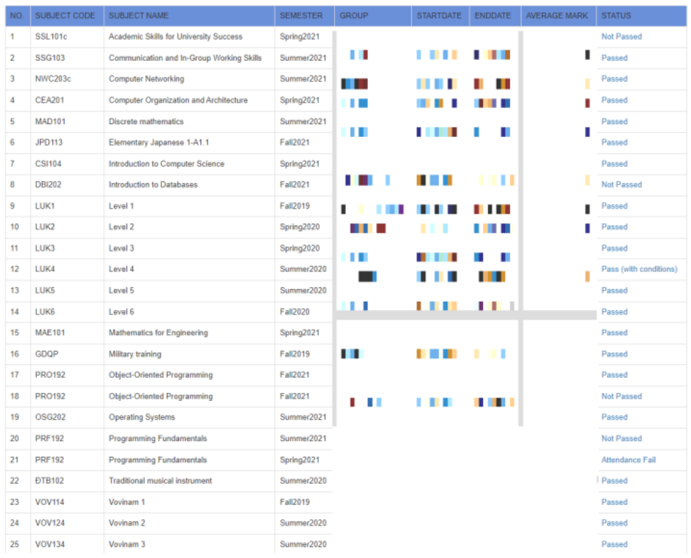
This is a Student Grading Management Sub-System database,for each subject that attended by the student, the lecture will give score to the assessment to each of their assessment. Below figure shows an Example of the assessments for course DBI202.



5 assessment(s)

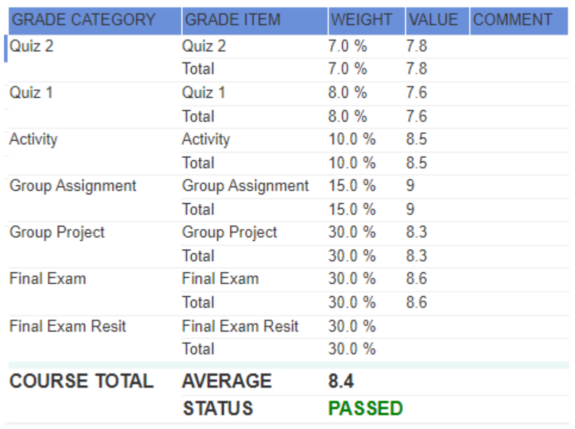
Category	Type	Part	Weight	Completion Criteria	Duration	LO	Question Type	No Question	Knowledge and Skill	Grading Guide	Note
Progress Tests	quiz	2	10.0%	>0	20'		Multiple choices Marked by Computer or a suitable format	20	up to 04 covered chapters	by instructor using computer	Instruction and shedules for Progress tests must be presented in the Course Implementation Plan approved by director of the campus. Progress test must be taken right after the last lectures of required material. Instructor has resposibility to review the test for students after graded.
Assignment	on-going	1	20.0%	>0	at home		Design; Implementation; Presentation		Simple RDBS design and implementation using a DBMS	guided by instructor, prepare at home present in class	40% Design, 20% Implementation, 40% Presentation of the whole Project
Labs	on-going	5	15.0%	>0	in lab session		practical exercises		related to studied modules	Guided by instructor	may be continued at home.
Practical Exam	practical exam	1	25.0%	>0	85'		Preferable to be marked by Scripts		DB programing skills	by exam board and department	Practical Exam database is up load in CMS in advanced.
Final Exam	final exam	1	30.0%	5	60'		Multiple choices Marked by Computer	60	Knowledge and skills in the course, but with much focus on the items in Chapters 2 to 6, >= 70% new questions (for the current semester);	by exam board	

Students can check their results at the end of semester as following example:



NO.	SUBJECT CODE	SUBJECT NAME	SEMESTER	GROUP	STARTDATE	ENDDATE	AVERAGE MARK	STATUS
1	SSL101c	Academic Skills for University Success	Spring2021	[redacted]	[redacted]	[redacted]	[redacted]	Not Passed
2	SSG103	Communication and In-Group Working Skills	Summer2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
3	NWC203c	Computer Networking	Summer2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
4	CEA201	Computer Organization and Architecture	Spring2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
5	MAD101	Discrete mathematics	Summer2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
6	JPD113	Elementary Japanese 1-A1.1	Fall2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
7	CSI104	Introduction to Computer Science	Spring2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
8	DBI202	Introduction to Databases	Fall2021	[redacted]	[redacted]	[redacted]	[redacted]	Not Passed
9	LUK1	Level 1	Fall2019	[redacted]	[redacted]	[redacted]	[redacted]	Passed
10	LUK2	Level 2	Spring2020	[redacted]	[redacted]	[redacted]	[redacted]	Passed
11	LUK3	Level 3	Spring2020	[redacted]	[redacted]	[redacted]	[redacted]	Passed
12	LUK4	Level 4	Summer2020	[redacted]	[redacted]	[redacted]	[redacted]	Pass (with conditions)
13	LUK5	Level 5	Summer2020	[redacted]	[redacted]	[redacted]	[redacted]	Passed
14	LUK6	Level 6	Fall2020	[redacted]	[redacted]	[redacted]	[redacted]	Passed
15	MAE101	Mathematics for Engineering	Spring2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
16	GDQP	Military training	Fall2019	[redacted]	[redacted]	[redacted]	[redacted]	Passed
17	PRO192	Object-Oriented Programming	Fall2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
18	PRO192	Object-Oriented Programming	Fall2021	[redacted]	[redacted]	[redacted]	[redacted]	Not Passed
19	OSG202	Operating Systems	Summer2021	[redacted]	[redacted]	[redacted]	[redacted]	Passed
20	PRF192	Programming Fundamentals	Summer2021					Not Passed
21	PRF192	Programming Fundamentals	Spring2021					Attendance Fail
22	ĐTB102	Traditional musical instrument	Summer2020					Passed
23	VOV114	Vovinam 1	Fall2019					Passed
24	VOV124	Vovinam 2	Summer2020					Passed
25	VOV134	Vovinam 3	Summer2020					Passed

Each Subject code, student can check their detailed result of as below example:



GRADE CATEGORY	GRADE ITEM	WEIGHT	VALUE	COMMENT
Quiz 2	Quiz 2	7.0 %	7.8	
	Total	7.0 %	7.8	
Quiz 1	Quiz 1	8.0 %	7.6	
	Total	8.0 %	7.6	
Activity	Activity	10.0 %	8.5	
	Total	10.0 %	8.5	
Group Assignment	Group Assignment	15.0 %	9	
	Total	15.0 %	9	
Group Project	Group Project	30.0 %	8.3	
	Total	30.0 %	8.3	
Final Exam	Final Exam	30.0 %	8.6	
	Total	30.0 %	8.6	
Final Exam Resit	Final Exam Resit	30.0 %		
	Total	30.0 %		
COURSE TOTAL	**AVERAGE**	**8.4**		
	STATUS	**PASSED**		

DATABASE REQUIREMEMTS

The database must consist of at least six tables that have been populated with data. The database is to support queries that would typically be submitted to the system for the topical area that you have chosen. You must do the following:

Self-investigation for the requirement of the system. Listed them all as form of reports, business rules.

· Using UML, Chen's notation to create an Entity Relationship (ER) model for your relational database. All entity types, their attributes and relationships must be clearly shown. You will also be required to show all cardinality and participation constraints. You should use some enhanced ER features in your conceptual model where it makes sense to do so.

· Map the EER model devised in part (1) into a relational data model. It must be normalised up to at least 3rd Normal Form.

· Using appropriate SQL commands create a set of database tables in MS SQL Server 2008+. You should also show all constraints used in the creation of the tables.

· Populate the database with a small amount of data. The data should be meaningful but does not need to be extensive. The following sites may be useful for quickly generating data:

♣ http://www.databasetestdata.com/ ♣ http://www.generatedata.com/

• Your database must contain one view, one trigger, on store procedure and an index (describe why).

• Create 10 sample queries that demonstrate the expressiveness of your database system. Your queries must demonstrate different aspects of the system.

Final Report

You must submit a brief final report which must include the following:

a) A brief description of the database including any assumptions made during the design (THIS IS VERY IMPORTANT TO CLERIFY THE ASSUMTIONS in form of business rules).

b) An ERD (Entity Relationship Diagram) that fully describes the database (giving descriptions on your work would be appreciated).

c) The relational schema derived from the ERD that is at least in 3NF (Any detail of the process would be appreciated).

d) The set of database statements used to create the tables used in your database. You do NOT need to include all the data and insert statements.

e) 10 queries that demonstrate the usefulness of the database. Also state why and when each query would be used. The following must be demonstrated by at least one of your queries:

- A query that uses ORDER BY
- A query that uses INNER JOINS
- A query that uses aggregate functions
- A query that uses the GROUP BY and HAVING clauses
- A query that uses a sub-query as a relation
- A query that uses a sub-query in the WHERE clause
- A query that uses partial matching in the WHERE clause
- A query that uses a self-JOIN

f) The trigger, store procedure, and the index should be added (explain why you make it) Demonstration You will be required to briefly demonstrate your system in one of the laboratory sessions prior to submission of the report.

Process

1. Define Entites

Student

- **StudentID**
- **TerritoryID** (where does this student lives)
- Student frist name
- Student last name
- BirthDay
- Sex
- Major
- Contact Mail
- Image (image to identify the student)

Lecture

- **LectureID**
- **TerritoryID** (where does this lecture lives)
- Report to (who the lecture report to)
- Lecture first name
- Lecture last name
- BirthDay
- Sex
- Worked since (when lecture started working in this school)
- Contact mail
- Report to (who this lecture will report to)

Territory

- **TerritoryID**

- Territory description

Course (as known as 'Subject')

- **CourseID**
- Course_Name
- Number of sessions
- Department
- Semester
- Number of credits

Class

- **ClassID**
- **LectureID**

Piece of Work (things must do if student take the course)

- **CourseID**
- **Category**
- Start date
- End date

Output

- **OutputID**
- **StudentID**
- **Category**
- Submission status

Assessments

- **AssessmentID**
- **Category**
- Type (Type of piece of work)
- Part (How many parts of value point need to pass the course)
- Weight (this assessment account for ?% of the total point value)
- Completion criteria (prerequisite condition to pass the assessment)
- Duraton (Time needed to do assessment)
- Number of questions

2. Convert ERD into relational mapping

2.1. Define cardinality constraints/partial constraints

- 1-1: look for partial constraints if it's total or partial
- n/n-1: move the primary key (PK) of the 1-sided table into the n-sided table and set it as foreign key (FK)
- n-n: create a new table that its PKs set is the combination of its relative tables' PKs

2.2. Look for relationships between entities

- Student and Course: 1 student can enrolls in many course, vice versa, the relationships is N-N
    - ➜ so the table called StudentGroup has been created and its PKs are the PKs set of Student and Course table

StudentGroup

- **GroupID**
- **StudentID**
- **CourseID**
- Enroll Date (dd/mm/yyyy that this student enroll in this course)

- Lecture and class: 1 lecture can teaches/instructes in many class, but 1 class can only be teached by 1 lecture

➜ so the relationship here is 1-N

- Class and StudentGroup: 1 group can be assigned into many class, 1 class can also has many groups

➜ so the relationship here is N-N

➜ so the table called Studied-in is created

Studied-in

- ClassID
- GroupID
- StudentID
- CourseID
- Enroll Date (dd/mm/yyyy that this student enroll in this course)

- Territory and Student/Lecture: as known as "Person", each person has their own territory that they live,

which include region, address, city, distinct,... ect.

But 1 territory is the place for many people lives in ➜ so the relationships here is 1-N

- Course and Piece of works (as known as to-do-Works): in each course include many pieces of works like assignments, exams,..etc.

But 1 piece of work is belong to 1 course

(the assignment of the Database course cannot be same as the assignment of Data structure and algorithms course

, the exam about character of japanese cannot be the same as madarin)

➜ so the so the relationships here is 1-N

- Student and Piece of works: if a student wants to pass this course and get the certificate, he/she will have to do all the Piece of works,

and all Piece of works can be done by many students (many student can take final exam, can do the same assignment)

➜ so the relationship here is N-N

➜ the new table called "process" is created

Process (The progress of student when doing Piece of Work)

- StudentID
- CourseID
- Category
- Start date
- End date

- Piece of works and Assessment: each Piece of works has their own assessment, assessment is the conditition to specify if the student pass the course or not.

but each assessment is belong to 1 course (the assessment of madarin course and japanese can't be the same)

➜ so the relationships here is 1-N

- Process and Output: each Student do a piece of work will has an output, each student has many output in many category, but 1 output in 1 category can only be possessed by 1 student

➜ so the relationships here is 1-N

Lecture and Output: The lecture will give score to students based on their output, many lectures can give marks to many student and vice versa (for example, the pratical exam and final exam can't be graded by 1 lecture)

➜ so the relationship here is N-N

➜ the new table called "grading" is created

Grading

- LectureID
- OutputID
- submission status

- Grading and Assessment: After lectures grading scores for students, scores then will be compared with assessment conditions to find out whether the student pass the course or not, 1 gradding score will have to compare to many assessment conditions and 1 assessment conditions will be compared to many gradding score

➜ so the relationship here is N-N

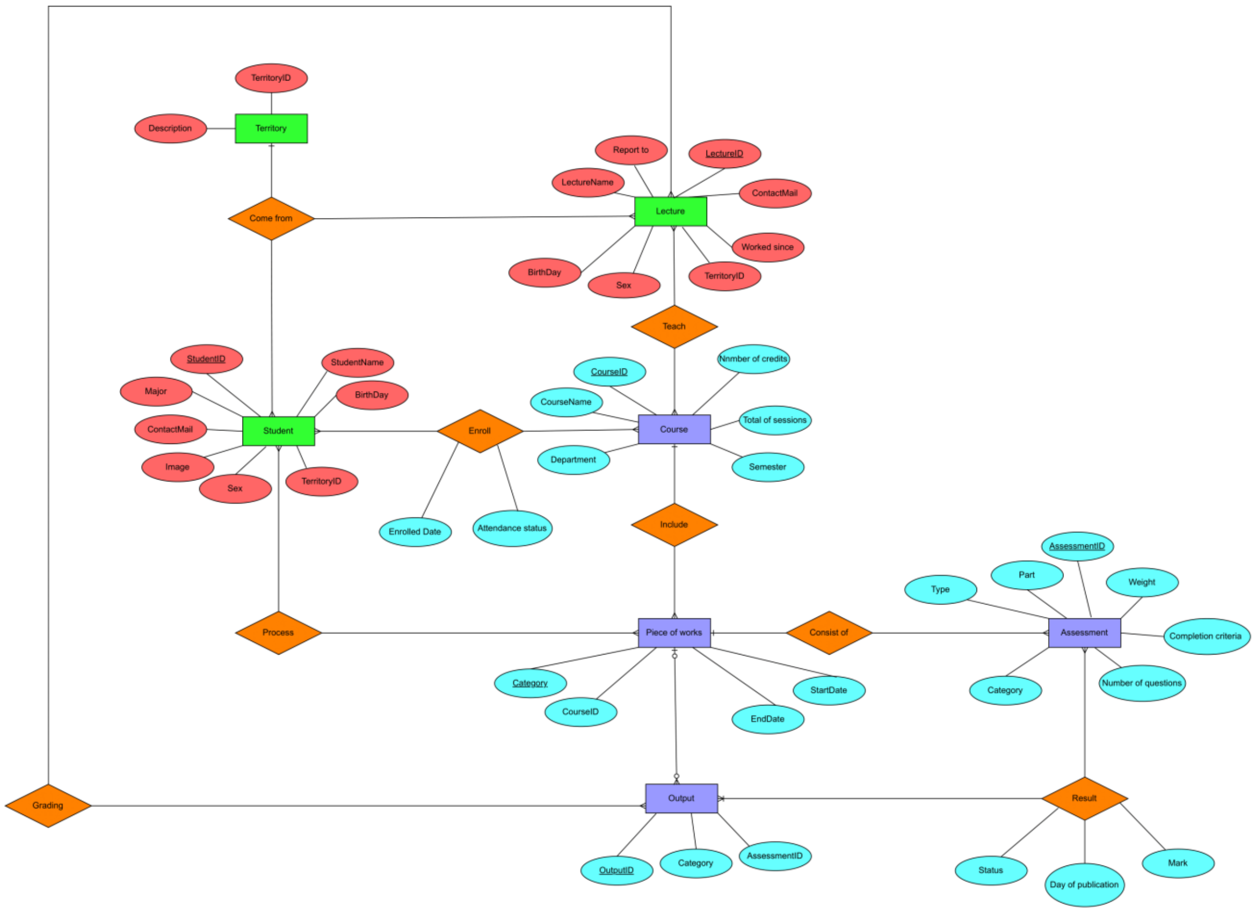
➜ the new table called "result" is created

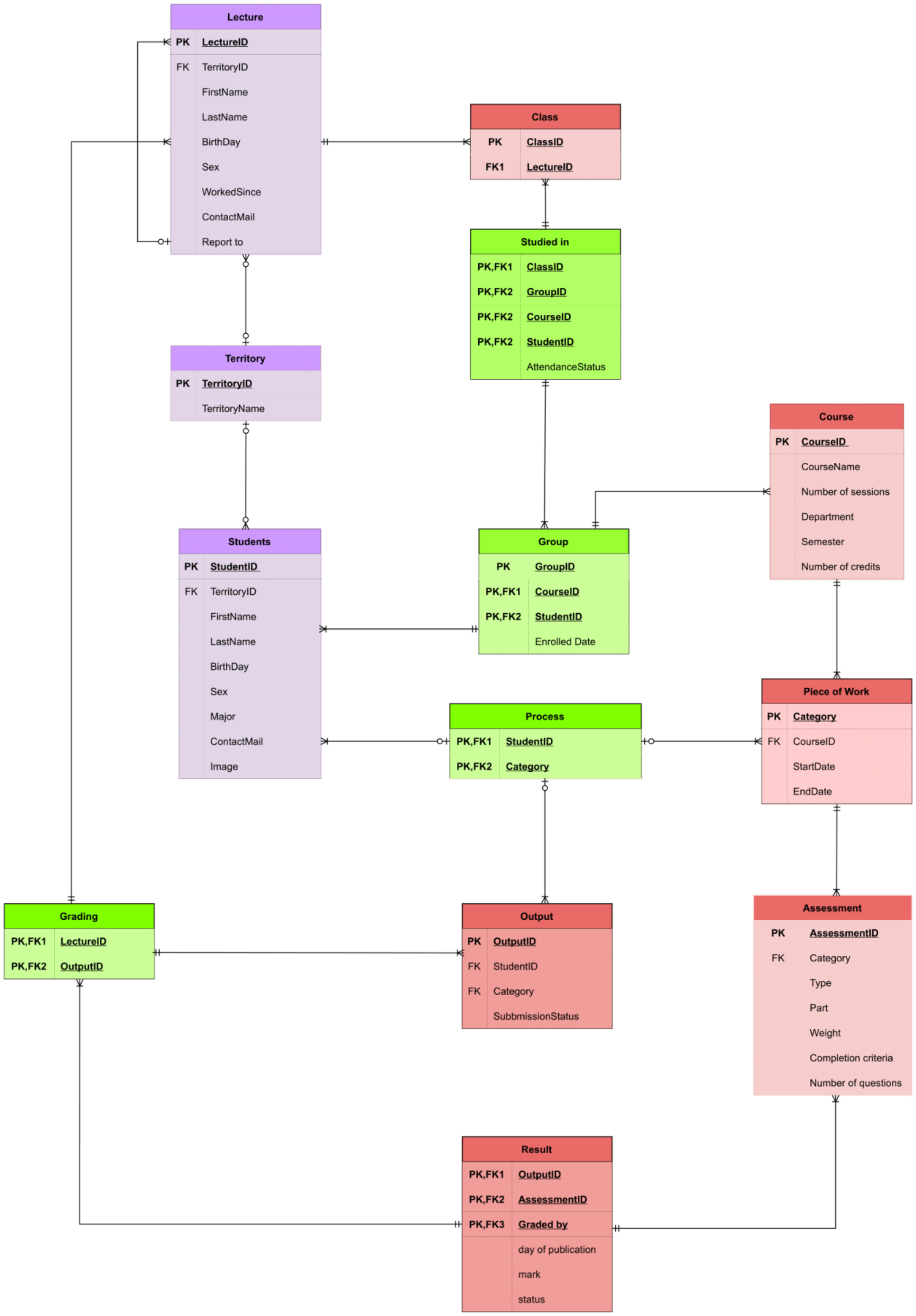
Result

- OutputID
- AssessmentID
- Day of publication
- Mark

- Status

3. Implement ERD and relational mapping




Lecture
PK LectureID
FK TerritoryID
FirstName
LastName
BirthDay
Sex
WorkedSince
ContactMail
Report to
Class
PK ClassID
FK1 LectureID
Studied in
PK,FK1 ClassID
PK,FK2 GroupID
PK,FK2 CourseID
PK,FK2 StudentID
AttendanceStatus
Territory
PK TerritoryID
TerritoryName
Course
PK CourseID
CourseName
Number of sessions
Department
Semester
Number of credits
Students
PK StudentID
FK TerritoryID
FirstName
LastName
BirthDay
Sex
Major
ContactMail
Image
Group
PK GroupID
PK,FK1 CourseID
PK,FK2 StudentID
Enrolled Date
Process
PK,FK1 StudentID
PK,FK2 Category
Piece of Work
PK Category
FK CourseID
StartDate
EndDate
Grading
PK,FK1 LectureID
PK,FK2 OutputID
Output
PK OutputID
FK StudentID
FK Category
SubbmissionStatus
Assessment
PK AssessmentID
FK Category
Type
Part
Weight
Completion criteria
Number of questions
Result
PK,FK1 OutputID
PK,FK2 AssessmentID
PK,FK3 Graded by
day of publication
mark
status

4. Initialize datatypes to attributes

Lecture		
Attributes	Data type	Null-allowed
LectureID	Nvarchar(25)	no
TerritoryID	int	no
LectureFirstName	Nvarchar(30)	no
LectureLastName	Nvarchar(30)	no
BirthDay	Date	yes
Sex	Bit	no
WorkedSince	Date	no
Report to	Date	yes
Department	Varchar(70)	no
ContacMail	Varchar(70)	no

Student		
Attributes	Data type	Null-allowed
StudentID	Nvarchar(20)	no
TerritoryID	int	no
StudentFirstName	Nvarchar(30)	no
StudentLastName	Nvarchar(30)	no
BirthDay	Date	yes

Sex	Bit	no
Major	Varchar(55)	no
ContacMail	Date	yes
Department	Varchar(70)	no
Image	Varchar(255)	no

Territory		
Attributes	Data type	Null-allowed
TerritoryID	int IDENTITY(1,1)	no
TerritoryName	Nvarchar(30)	no

Course		
Attributes	Data type	Null-allowed
CourseID	varchar (15)	no
CourseName	int	no
Number of sessions	Nvarchar(30)	no
Department	Nvarchar(30)	no
Number of credits	int	no

StudentGroup		
Attributes	Data type	Null-allowed
GroupID	varchar (15)	no
StudentID	varchar (25)	no
CourseID	varchar (15)	no
EnrolledDate	Date	no

Studied-in		
Attributes	Data type	Null-allowed
ClassID	varchar (80)	no
StudentID	varchar (25)	no
CourseID	varchar (15)	no
GroupID	varchar (15)	no
Attendance status	bit	no

Class		
Attributes	Data type	Null-allowed
ClassID	varchar (80)	no
LectureID	varchar (25)	no

PieceOfWork		
Attributes	Data type	Null-allowed
Category	varchar (80)	no
CourseID	varchar (25)	no

StartDate	date	no
EndDate	date	no

Process		
Attributes	Data type	Null-allowed
StudentID	varchar (25)	no
Category	varchar (40)	no

Assessment		
Attributes	Data type	Null-allowed
AssessmentID	int IDENTITY(1,1)	no
Category	varchar (40)	no
Type	varchar (15)	no
Part	int	no
Weight	Decimal(5,2)	no
Completion Criteria	int	no
No Question	int	yes

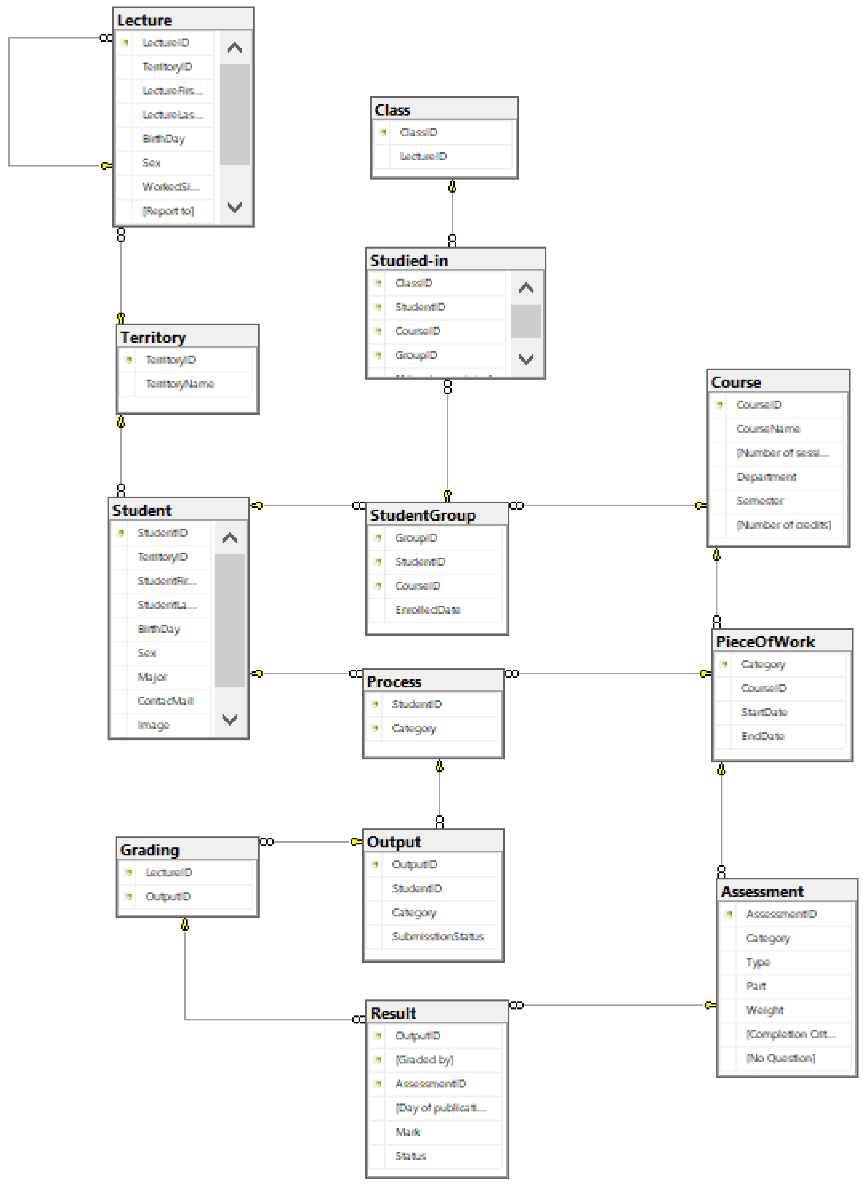
Output		
Attributes	Data type	Null-allowed
OutputID	int IDENTITY(1,1)	no
StudentID	varchar (25)	no

Category	varchar (40)	no
SubmisstionStatus	int	no

Grading		
Attributes	Data type	Null-allowed
LectureID	Varchar(25)	no
OutputID	int	no

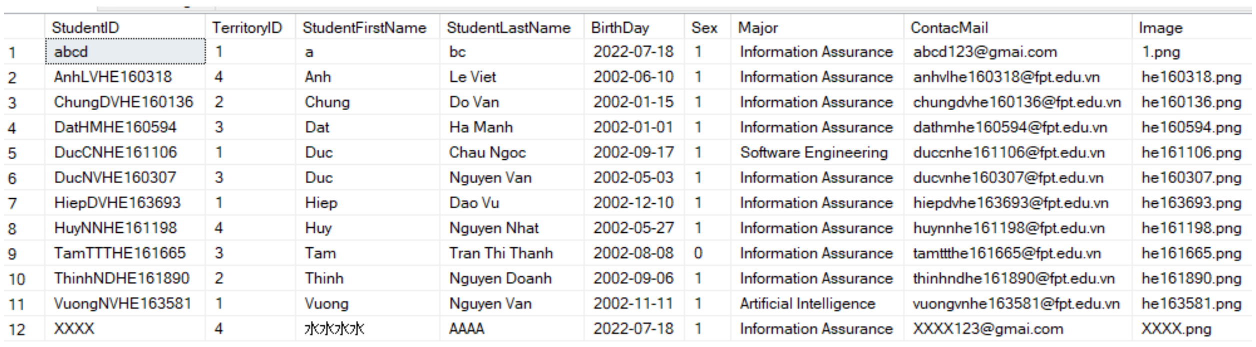
Result		
Attributes	Data type	Null-allowed
OutputID	int IDENTITY(1,1)	no
Graded by	varchar (25)	no
AssessmentID	int	no
Day of publication	date	no
Mark	float	yes
Status	bit	yes

5. Generate datas (Script file is on github)



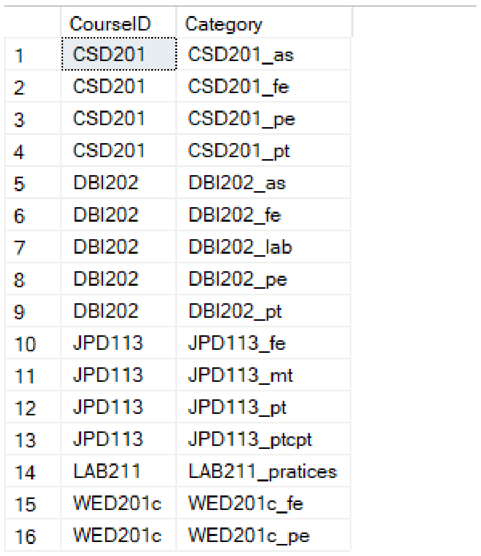
6. Queries

```
--- A query that uses ORDER BY
--SORT STUDENT ASCENDING
SELECT * FROM Student ORDER BY StudentID
```



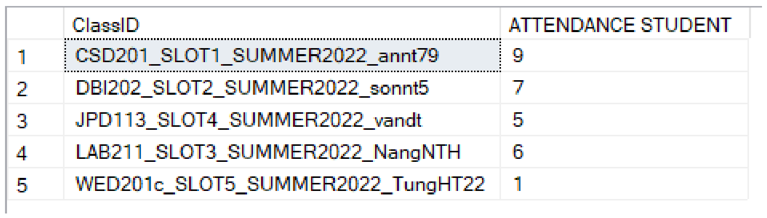
	StudentID	TerritoryID	StudentFirstName	StudentLastName	BirthDay	Sex	Major	ContacMail	Image
1	abcd	1	a	bc	2022-07-18	1	Information Assurance	abcd123@gmai.com	1.png
2	AnhLVHE160318	4	Anh	Le Viet	2002-06-10	1	Information Assurance	anhvlhe160318@fpt.edu.vn	he160318.png
3	ChungDVHE160136	2	Chung	Do Van	2002-01-15	1	Information Assurance	chungdvhe160136@fpt.edu.vn	he160136.png
4	DatHMHE160594	3	Dat	Ha Manh	2002-01-01	1	Information Assurance	dathmhe160594@fpt.edu.vn	he160594.png
5	DucCNHE161106	1	Duc	Chau Ngoc	2002-09-17	1	Software Engineering	duccnhe161106@fpt.edu.vn	he161106.png
6	DucNVHE160307	3	Duc	Nguyen Van	2002-05-03	1	Information Assurance	ducvnhe160307@fpt.edu.vn	he160307.png
7	HiepDVHE163693	1	Hiep	Dao Vu	2002-12-10	1	Information Assurance	hiepdvhe163693@fpt.edu.vn	he163693.png
8	HuyNNHE161198	4	Huy	Nguyen Nhat	2002-05-27	1	Information Assurance	huynnhe161198@fpt.edu.vn	he161198.png
9	TamTTTHE161665	3	Tam	Tran Thi Thanh	2002-08-08	0	Information Assurance	tamttthe161665@fpt.edu.vn	he161665.png
10	ThinhNDHE161890	2	Thinh	Nguyen Doanh	2002-09-06	1	Information Assurance	thinhndhe161890@fpt.edu.vn	he161890.png
11	VuongNVHE163581	1	Vuong	Nguyen Van	2002-11-11	1	Artificial Intelligence	vuongvnhe163581@fpt.edu.vn	he163581.png
12	XXXX	4	水水水水	AAAA	2022-07-18	1	Information Assurance	XXXX123@gmai.com	XXXX.png

```
--- A query that uses INNER JOINS
SELECT C.CourseID, P.Category FROM Course C INNER JOIN
PieceOfWork P
ON C.CourseID = P.CourseID
```



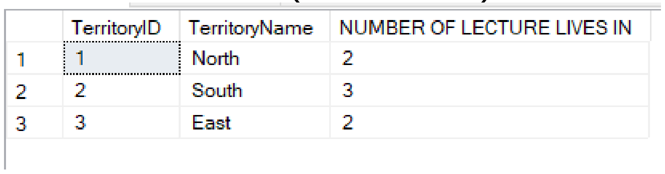
	CourseID	Category
1	CSD201	CSD201_as
2	CSD201	CSD201_fe
3	CSD201	CSD201_pe
4	CSD201	CSD201_pt
5	DBI202	DBI202_as
6	DBI202	DBI202_fe
7	DBI202	DBI202_lab
8	DBI202	DBI202_pe
9	DBI202	DBI202_pt
10	JPD113	JPD113_fe
11	JPD113	JPD113_mt
12	JPD113	JPD113_pt
13	JPD113	JPD113_ptcpt
14	LAB211	LAB211_pratices
15	WED201c	WED201c_fe
16	WED201c	WED201c_pe

```
--A query that uses aggregate functions
SELECT C.ClassID,COUNT(StudentID) AS [ATTENDANCE STUDENT]
FROM Class C INNER JOIN [Studied-in] SI
ON C.ClassID = SI.ClassID AND SI.[Attendance status] = 1
GROUP BY C.ClassID
```



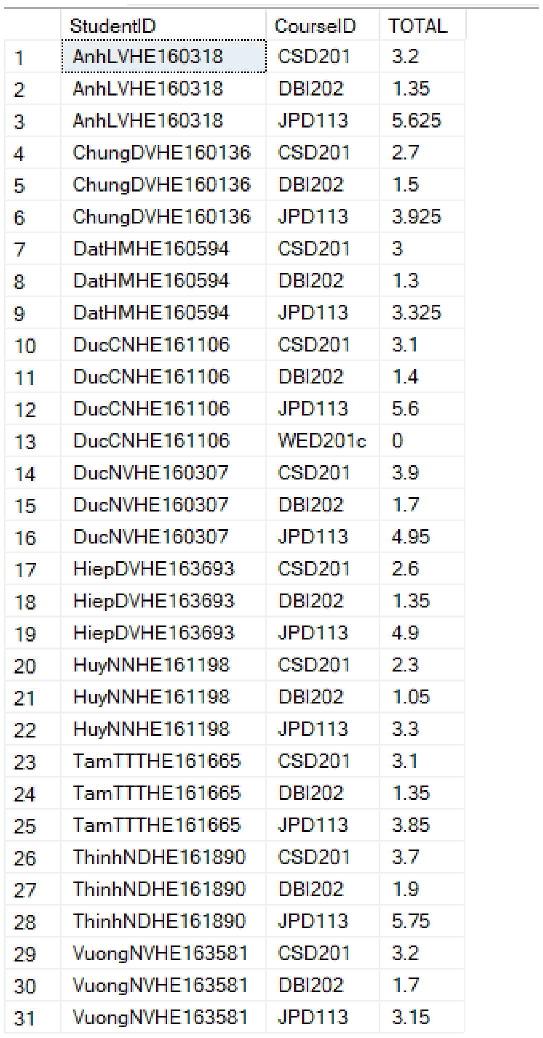
	ClassID	ATTENDANCE STUDENT
1	CSD201_SLOT1_SUMMER2022_annt79	9
2	DBI202_SLOT2_SUMMER2022_sonnt5	7
3	JPD113_SLOT4_SUMMER2022_vandt	5
4	LAB211_SLOT3_SUMMER2022_NangNTH	6
5	WED201c_SLOT5_SUMMER2022_TungHT22	1

```
--A query that uses the GROUP BY and HAVING clauses
--The difference between the having and
--where clause in SQL is that the where
--clause cannot be used with aggregates(max, min,count,avg,sum), but the
having clause can.
--NUMBER OF LECTURE LIVES IN EACH TERRITORY THAT GREATER
THAN 1
SELECT T.TerritoryID,T.TerritoryName, COUNT(l.LectureID) AS
[NUMBER OF LECTURE LIVES IN] FROM Territory T INNER JOIN
Lecture L
ON T.TerritoryID = L.TerritoryID
GROUP BY T.TerritoryID,T.TerritoryName
HAVING COUNT(l.LectureID) > 1
```



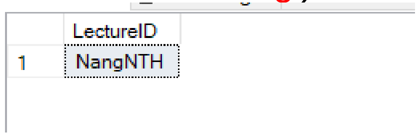
	TerritoryID	TerritoryName	NUMBER OF LECTURE LIVES IN
1	1	North	2
2	2	South	3
3	3	East	2

```
--A query that uses a sub-query as a relation
--COUNT TOTAL MARK OF STUDENT IN EACH COURSE
SELECT StudentID,CourseID,[TOTAL]
FROM(
SELECT O.StudentID,POW.CourseID, SUM(ISNULL(R.Mark *
A.Weight/100,0)) AS [TOTAL] FROM Result R INNER JOIN Grading G
ON R.OutputID = G.OutputID INNER JOIN Output O
ON O.OutputID = G.OutputID INNER JOIN Assessment A
ON O.Category = A.Category AND O.Category NOT LIKE 'LAB%' INNER
JOIN PieceOfWork POW
ON POW.Category = O.Category GROUP BY
O.StudentID,POW.CourseID) AS A
ORDER BY StudentID
```



	StudentID	CourseID	TOTAL
1	AnhLVHE160318	CSD201	3.2
2	AnhLVHE160318	DBI202	1.35
3	AnhLVHE160318	JPD113	5.625
4	ChungDVHE160136	CSD201	2.7
5	ChungDVHE160136	DBI202	1.5
6	ChungDVHE160136	JPD113	3.925
7	DatHMHE160594	CSD201	3
8	DatHMHE160594	DBI202	1.3
9	DatHMHE160594	JPD113	3.325
10	DucCNHE161106	CSD201	3.1
11	DucCNHE161106	DBI202	1.4
12	DucCNHE161106	JPD113	5.6
13	DucCNHE161106	WED201c	0
14	DucNVHE160307	CSD201	3.9
15	DucNVHE160307	DBI202	1.7
16	DucNVHE160307	JPD113	4.95
17	HiepDVHE163693	CSD201	2.6
18	HiepDVHE163693	DBI202	1.35
19	HiepDVHE163693	JPD113	4.9
20	HuyNNHE161198	CSD201	2.3
21	HuyNNHE161198	DBI202	1.05
22	HuyNNHE161198	JPD113	3.3
23	TamTTHE161665	CSD201	3.1
24	TamTTHE161665	DBI202	1.35
25	TamTTHE161665	JPD113	3.85
26	ThinhNDHE161890	CSD201	3.7
27	ThinhNDHE161890	DBI202	1.9
28	ThinhNDHE161890	JPD113	5.75
29	VuongNVHE163581	CSD201	3.2
30	VuongNVHE163581	DBI202	1.7
31	VuongNVHE163581	JPD113	3.15

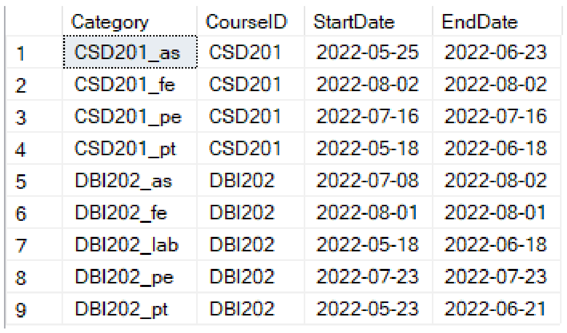
```
--A query that uses a sub-query in the WHERE clause
--FIND ALL LECTURE THAT THEIR SUPERVISOR'S DEPARTMENT IS
'Machine learning'
SELECT L1.LectureID
FROM Lecture L1
WHERE L1.[Report to] IN (SELECT L2.LectureID
                                    FROM Lecture L2
                                        WHERE L2.Department =
'Machine learning')
```



	LectureID
1	NangNTH

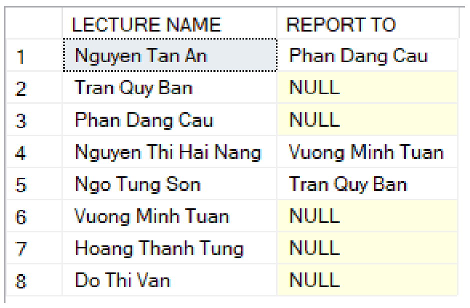
```
--A query that uses partial matching in the WHERE clause
```

```
--SELECT CATEGORIES THAT IN CSD OR DBI COURSE
SELECT *
FROM PieceOfWork
WHERE Category LIKE 'CSD201%' OR Category LIKE 'DBI202%'
```



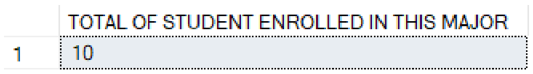
	Category	CourseID	StartDate	EndDate
1	CSD201_as	CSD201	2022-05-25	2022-06-23
2	CSD201_fe	CSD201	2022-08-02	2022-08-02
3	CSD201_pe	CSD201	2022-07-16	2022-07-16
4	CSD201_pt	CSD201	2022-05-18	2022-06-18
5	DBI202_as	DBI202	2022-07-08	2022-08-02
6	DBI202_fe	DBI202	2022-08-01	2022-08-01
7	DBI202_lab	DBI202	2022-05-18	2022-06-18
8	DBI202_pe	DBI202	2022-07-23	2022-07-23
9	DBI202_pt	DBI202	2022-05-23	2022-06-21

```
--A query that uses a self-JOIN
--The SELF-JOIN is a special kind of joins
--that allow you to join a table to itself using
--either LEFT JOIN or INNER JOIN clause.
--You use self-join to create a result set that joins the rows with the other
rows within the same table.
--FIND ALL LECTRUE FULLNAME AND THEIR SUPERVISOR, IF DONT
HAVE THEN DISPLAY NULL
SELECT L1.LectureLastName+'    '+L1.LectureFirstName AS [LECTURE
NAME],L2.LectureLastName+'    '+L2.LectureFirstName AS [REPORT
TO] FROM Lecture L1 LEFT JOIN Lecture L2
ON L1.[Report to] = L2.LectureID
```

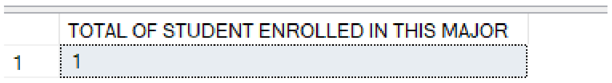


	LECTURE NAME	REPORT TO
1	Nguyen Tan An	Phan Dang Cau
2	Tran Quy Ban	NULL
3	Phan Dang Cau	NULL
4	Nguyen Thi Hai Nang	Vuong Minh Tuan
5	Ngo Tung Son	Tran Quy Ban
6	Vuong Minh Tuan	NULL
7	Hoang Thanh Tung	NULL
8	Do Thi Van	NULL

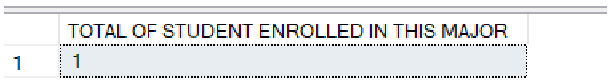
```
--STORED PROCEDUCE
--COUNT NUMBER OF STUDENT ENROLLED IN EACH MAJOR
GO
	CREATE PROC Count_Num_of_Major
	@Major VARCHAR(55)
	AS
	BEGIN
		SELECT COUNT(Major) AS [TOTAL OF STUDENT
ENROLLED IN THIS MAJOR] FROM Student WHERE Major = @Major
	END
GO
EXEC Count_Num_of_Major N'Information Assurance'
EXEC Count_Num_of_Major N'Software Engineering'
EXEC Count_Num_of_Major N'Artificial Intelligence'
SELECT * FROM Student
DROP PROC Count_Num_of_Major
```



```
   TOTAL OF STUDENT ENROLLED IN THIS MAJOR
1  10
```



```
   TOTAL OF STUDENT ENROLLED IN THIS MAJOR
1  1
```

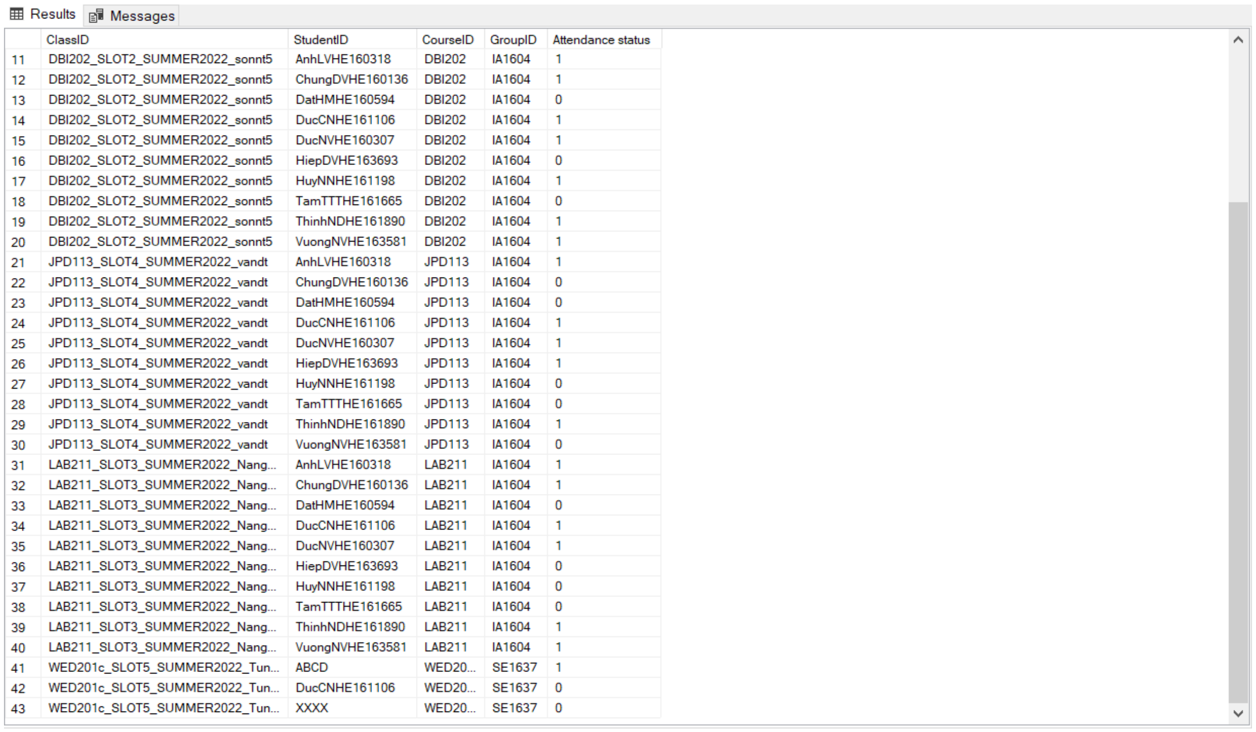


```
   TOTAL OF STUDENT ENROLLED IN THIS MAJOR
1  1
```

```
--TRIGGER
--DELETE STUDENT THAT ABSENT
GO
	ALTER TRIGGER DROP_OUT_ABSENT_STUDENT
	ON [Studied-in]
	AFTER DELETE
```

```
        AS
        BEGIN
                        DECLARE @ATTENDANCE BIT
                        SELECT @ATTENDANCE  = [Attendance status] FROM
deleted
                        IF(@ATTENDANCE = 1)
                        BEGIN
                                PRINT N'YOU CANT DELETE THIS STUDENT'
                                ROLLBACK TRAN
                        END
                        ELSE
                        BEGIN
                                PRINT N'THIS STUDENT HAS BEEN DROPPED
OUT BY LECTURE'
                        END
        END
GO
```

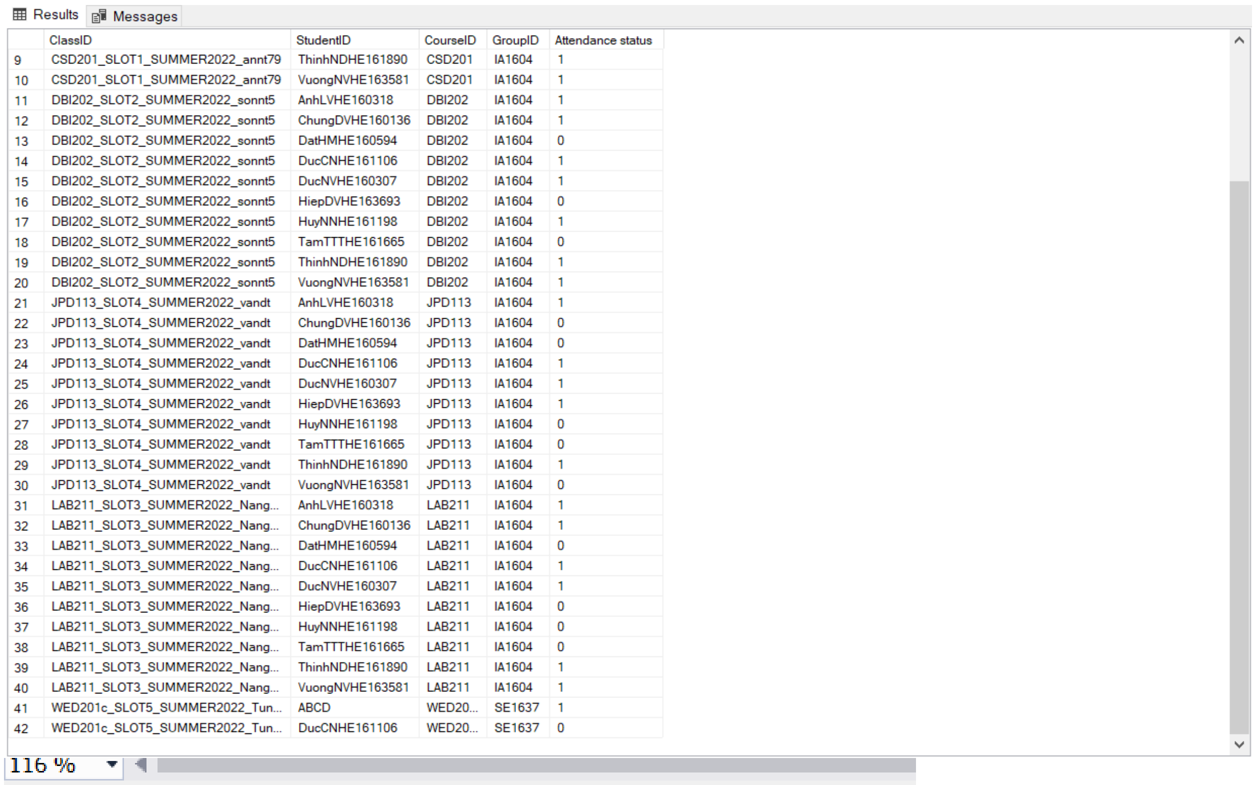
BEFORE TRIGGER



Results | Messages

	ClassID	StudentID	CourseID	GroupID	Attendance status
11	DBI202_SLOT2_SUMMER2022_sonnt5	AnhLVHE160318	DBI202	IA1604	1
12	DBI202_SLOT2_SUMMER2022_sonnt5	ChungDVHE160136	DBI202	IA1604	1
13	DBI202_SLOT2_SUMMER2022_sonnt5	DatHMHE160594	DBI202	IA1604	0
14	DBI202_SLOT2_SUMMER2022_sonnt5	DucCNHE161106	DBI202	IA1604	1
15	DBI202_SLOT2_SUMMER2022_sonnt5	DucNVHE160307	DBI202	IA1604	1
16	DBI202_SLOT2_SUMMER2022_sonnt5	HiepDVHE163693	DBI202	IA1604	0
17	DBI202_SLOT2_SUMMER2022_sonnt5	HuyNNHE161198	DBI202	IA1604	1
18	DBI202_SLOT2_SUMMER2022_sonnt5	TamTTTHE161665	DBI202	IA1604	0
19	DBI202_SLOT2_SUMMER2022_sonnt5	ThinhNDHE161890	DBI202	IA1604	1
20	DBI202_SLOT2_SUMMER2022_sonnt5	VuongNVHE163581	DBI202	IA1604	1
21	JPD113_SLOT4_SUMMER2022_vandt	AnhLVHE160318	JPD113	IA1604	1
22	JPD113_SLOT4_SUMMER2022_vandt	ChungDVHE160136	JPD113	IA1604	0
23	JPD113_SLOT4_SUMMER2022_vandt	DatHMHE160594	JPD113	IA1604	0
24	JPD113_SLOT4_SUMMER2022_vandt	DucCNHE161106	JPD113	IA1604	1
25	JPD113_SLOT4_SUMMER2022_vandt	DucNVHE160307	JPD113	IA1604	1
26	JPD113_SLOT4_SUMMER2022_vandt	HiepDVHE163693	JPD113	IA1604	1
27	JPD113_SLOT4_SUMMER2022_vandt	HuyNNHE161198	JPD113	IA1604	0
28	JPD113_SLOT4_SUMMER2022_vandt	TamTTTHE161665	JPD113	IA1604	0
29	JPD113_SLOT4_SUMMER2022_vandt	ThinhNDHE161890	JPD113	IA1604	1
30	JPD113_SLOT4_SUMMER2022_vandt	VuongNVHE163581	JPD113	IA1604	0
31	LAB211_SLOT3_SUMMER2022_Nang...	AnhLVHE160318	LAB211	IA1604	1
32	LAB211_SLOT3_SUMMER2022_Nang...	ChungDVHE160136	LAB211	IA1604	1
33	LAB211_SLOT3_SUMMER2022_Nang...	DatHMHE160594	LAB211	IA1604	0
34	LAB211_SLOT3_SUMMER2022_Nang...	DucCNHE161106	LAB211	IA1604	1
35	LAB211_SLOT3_SUMMER2022_Nang...	DucNVHE160307	LAB211	IA1604	1
36	LAB211_SLOT3_SUMMER2022_Nang...	HiepDVHE163693	LAB211	IA1604	0
37	LAB211_SLOT3_SUMMER2022_Nang...	HuyNNHE161198	LAB211	IA1604	0
38	LAB211_SLOT3_SUMMER2022_Nang...	TamTTTHE161665	LAB211	IA1604	0
39	LAB211_SLOT3_SUMMER2022_Nang...	ThinhNDHE161890	LAB211	IA1604	1
40	LAB211_SLOT3_SUMMER2022_Nang...	VuongNVHE163581	LAB211	IA1604	1
41	WED201c_SLOT5_SUMMER2022_Tun...	ABCD	WED20...	SE1637	1
42	WED201c_SLOT5_SUMMER2022_Tun...	DucCNHE161106	WED20...	SE1637	0
43	WED201c_SLOT5_SUMMER2022_Tun...	XXXX	WED20...	SE1637	0

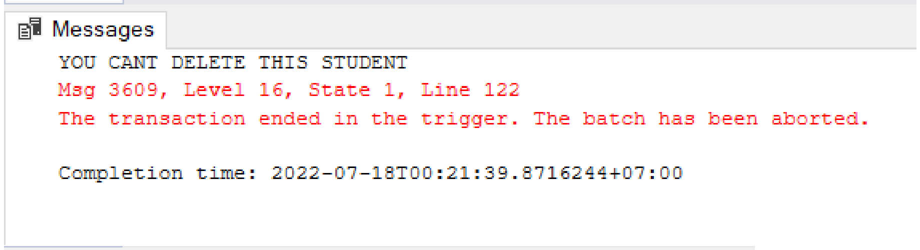
AFTER TRIGGER



Results | Messages

	ClassID	StudentID	CourseID	GroupID	Attendance status
9	CSD201_SLOT1_SUMMER2022_annt79	ThinhNDHE161890	CSD201	IA1604	1
10	CSD201_SLOT1_SUMMER2022_annt79	VuongNVHE163581	CSD201	IA1604	1
11	DBI202_SLOT2_SUMMER2022_sonnt5	AnhLVHE160318	DBI202	IA1604	1
12	DBI202_SLOT2_SUMMER2022_sonnt5	ChungDVHE160136	DBI202	IA1604	1
13	DBI202_SLOT2_SUMMER2022_sonnt5	DatHMHE160594	DBI202	IA1604	0
14	DBI202_SLOT2_SUMMER2022_sonnt5	DucCNHE161106	DBI202	IA1604	1
15	DBI202_SLOT2_SUMMER2022_sonnt5	DucNVHE160307	DBI202	IA1604	1
16	DBI202_SLOT2_SUMMER2022_sonnt5	HiepDVHE163693	DBI202	IA1604	0
17	DBI202_SLOT2_SUMMER2022_sonnt5	HuyNNHE161198	DBI202	IA1604	1
18	DBI202_SLOT2_SUMMER2022_sonnt5	TamTTTHE161665	DBI202	IA1604	0
19	DBI202_SLOT2_SUMMER2022_sonnt5	ThinhNDHE161890	DBI202	IA1604	1
20	DBI202_SLOT2_SUMMER2022_sonnt5	VuongNVHE163581	DBI202	IA1604	1
21	JPD113_SLOT4_SUMMER2022_vandt	AnhLVHE160318	JPD113	IA1604	1
22	JPD113_SLOT4_SUMMER2022_vandt	ChungDVHE160136	JPD113	IA1604	0
23	JPD113_SLOT4_SUMMER2022_vandt	DatHMHE160594	JPD113	IA1604	0
24	JPD113_SLOT4_SUMMER2022_vandt	DucCNHE161106	JPD113	IA1604	1
25	JPD113_SLOT4_SUMMER2022_vandt	DucNVHE160307	JPD113	IA1604	1
26	JPD113_SLOT4_SUMMER2022_vandt	HiepDVHE163693	JPD113	IA1604	1
27	JPD113_SLOT4_SUMMER2022_vandt	HuyNNHE161198	JPD113	IA1604	0
28	JPD113_SLOT4_SUMMER2022_vandt	TamTTTHE161665	JPD113	IA1604	0
29	JPD113_SLOT4_SUMMER2022_vandt	ThinhNDHE161890	JPD113	IA1604	1
30	JPD113_SLOT4_SUMMER2022_vandt	VuongNVHE163581	JPD113	IA1604	0
31	LAB211_SLOT3_SUMMER2022_Nang...	AnhLVHE160318	LAB211	IA1604	1
32	LAB211_SLOT3_SUMMER2022_Nang...	ChungDVHE160136	LAB211	IA1604	1
33	LAB211_SLOT3_SUMMER2022_Nang...	DatHMHE160594	LAB211	IA1604	0
34	LAB211_SLOT3_SUMMER2022_Nang...	DucCNHE161106	LAB211	IA1604	1
35	LAB211_SLOT3_SUMMER2022_Nang...	DucNVHE160307	LAB211	IA1604	1
36	LAB211_SLOT3_SUMMER2022_Nang...	HiepDVHE163693	LAB211	IA1604	0
37	LAB211_SLOT3_SUMMER2022_Nang...	HuyNNHE161198	LAB211	IA1604	0
38	LAB211_SLOT3_SUMMER2022_Nang...	TamTTTHE161665	LAB211	IA1604	0
39	LAB211_SLOT3_SUMMER2022_Nang...	ThinhNDHE161890	LAB211	IA1604	1
40	LAB211_SLOT3_SUMMER2022_Nang...	VuongNVHE163581	LAB211	IA1604	1
41	WED201c_SLOT5_SUMMER2022_Tun...	ABCD	WED20...	SE1637	1
42	WED201c_SLOT5_SUMMER2022_Tun...	DucCNHE161106	WED20...	SE1637	0

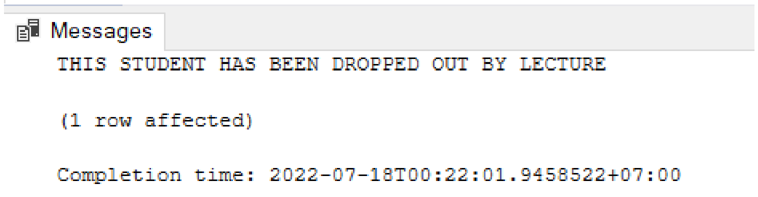
116 %



Messages

```
YOU CANT DELETE THIS STUDENT
Msg 3609, Level 16, State 1, Line 122
The transaction ended in the trigger. The batch has been aborted.

Completion time: 2022-07-18T00:21:39.8716244+07:00
```



Messages

```
THIS STUDENT HAS BEEN DROPPED OUT BY LECTURE

(1 row affected)

Completion time: 2022-07-18T00:22:01.9458522+07:00
```