

2022

# Final report

## Grade Management System

DBI202 - SE1647

Prepared By Lê Đình Tuấn

HE160710

Lecturer Ngô Tùng Sơn

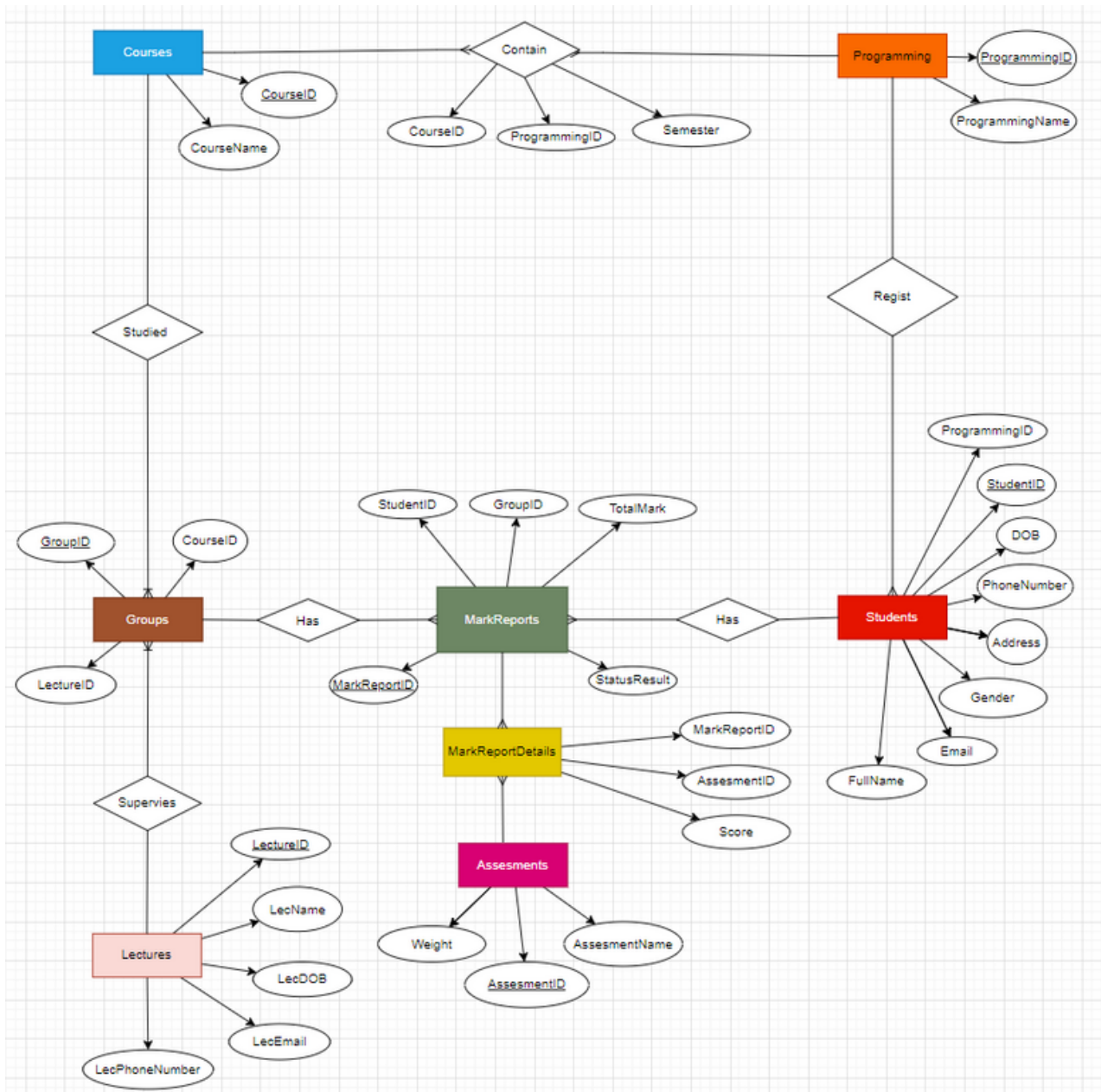


# Brief description

To create a database for grade management, we must clear that database has some rules below

- Every student can regist a programing and each programing include some courses.
- One programming has many course and one course can be in many programming.
- One group only teach one course but one course can be touch in many group.
- One lecturer can supervise many group but one group only is supervised by one lecturer.
- Each group manage many student, so one student can join many group, therefore each group has many mark report and each student too.
- Each mark report has detail of them, each detail include each assesment and score of each assesment
- Each mark report detail has many assesment, and one assesment must be in one mark report detail
- Mark report detail does not include total mark of each course of each student, this is in mark report.
- Programming does not record semester and year, but they were recorded in a relationship between programming and course called "Contain".
- Some students has the different programming but they can study in the same class

# 2 Entity Relationship Diagram



Fully entity relationship diagram for  
grade management system

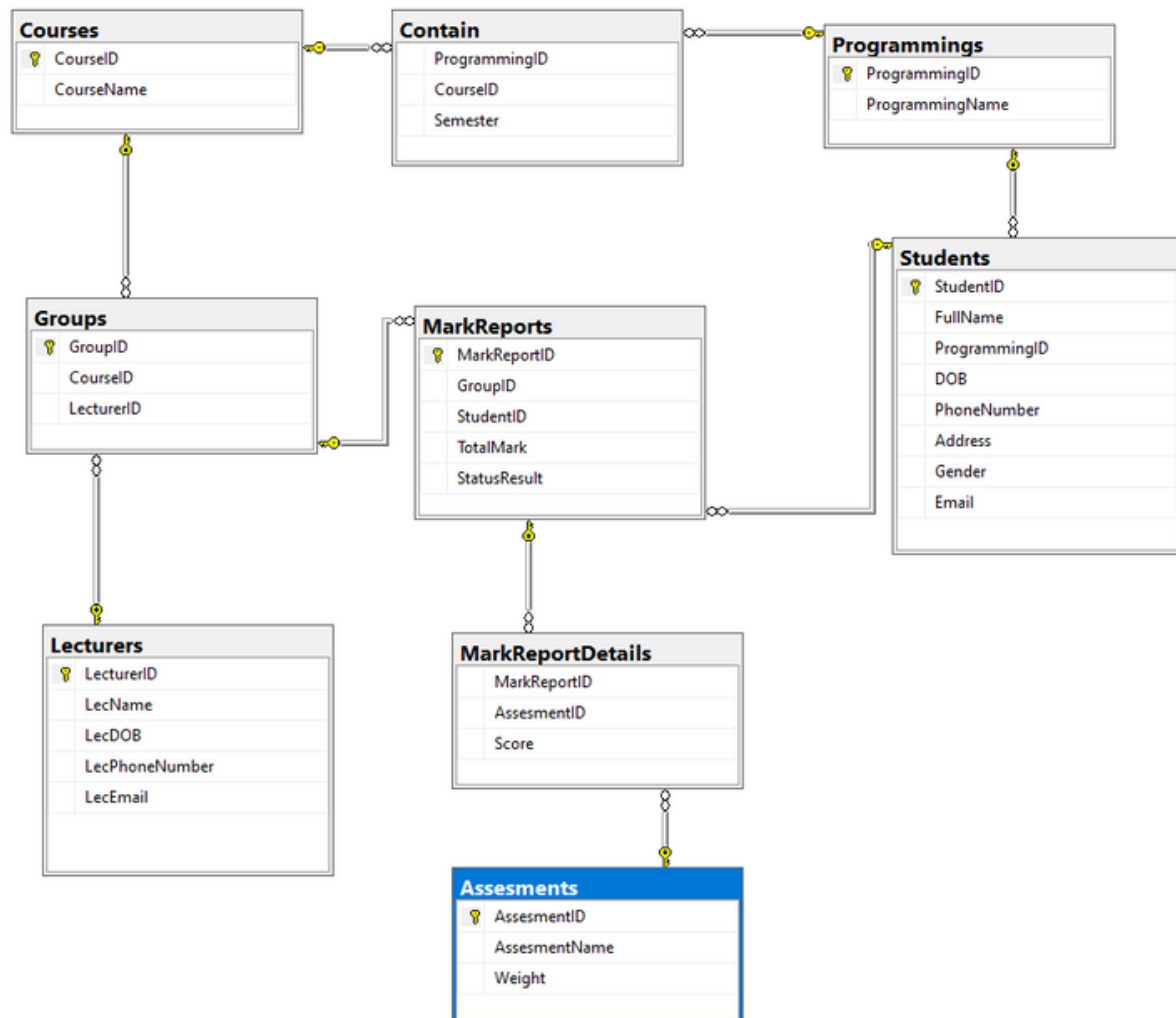
# 2 Entity Relationship Diagram

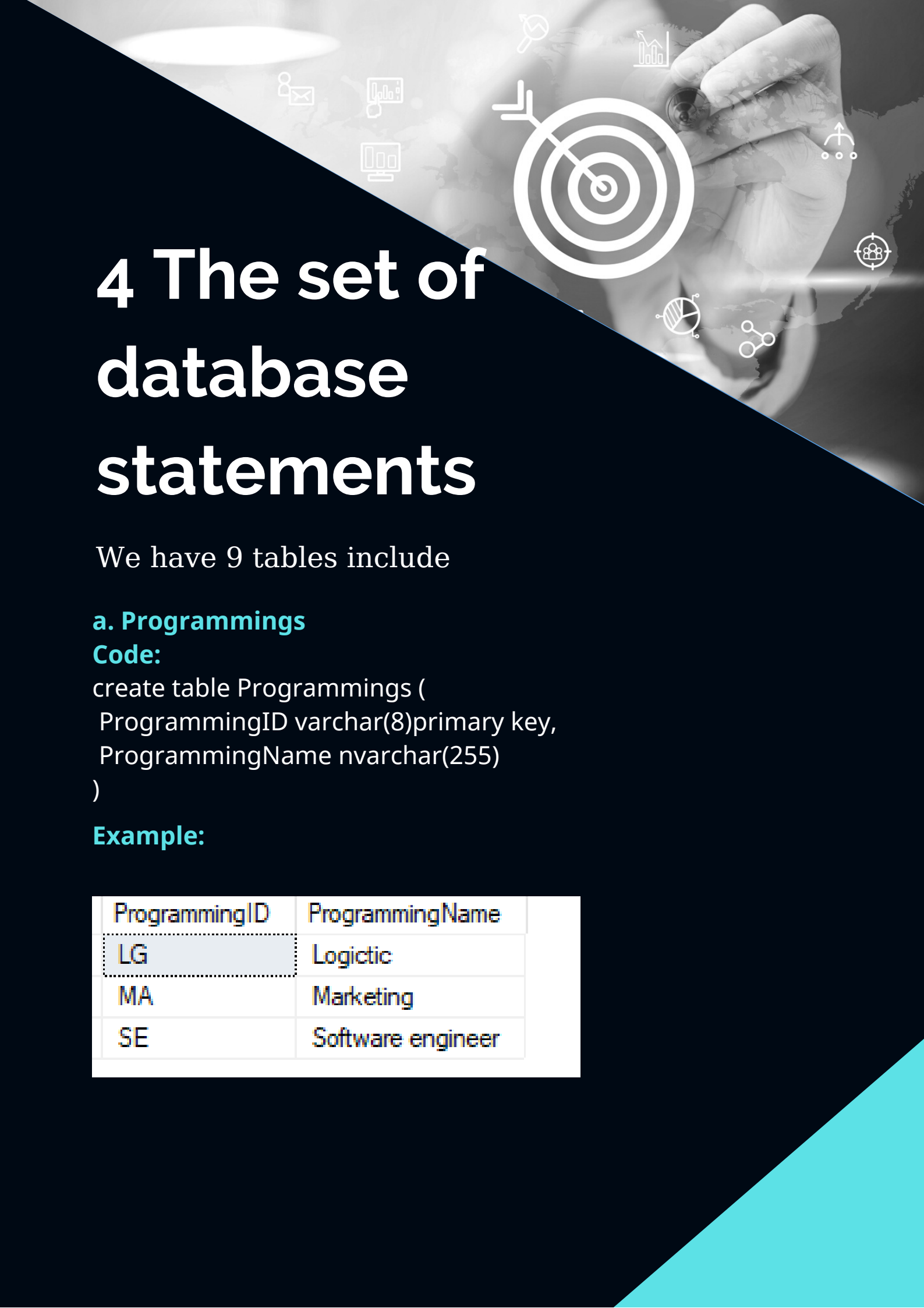
---

Base on description, we can present serveral entities and attributes of them

- The database has the following tables:
  - **Students:** StudentID(primary key), FullName, Address, ProgrammingID, DOB, Email, PhoneNumber, Gender
  - **Programmings:** ProgrammingID(primary key), ProgrammingName
  - Relationship **Contain:** CourseID, ProgrammingID, Semester
  - **Courses:** CourseID(primary key), CourseName
  - **Lecturers:** LecturerID(primary key), LecName, LecEmail, LecDOB, LecPhoneNumber
  - **Groups:** GroupID(primary key), LecID, CourseID
  - **MarkReports:** MarkReportID(primary key), GroupID, StudentID, TotalMark, StatusResult
  - **MarkReportDetails:** AssesmenrID, MarkReportID, score
  - **Assesments:** AssesmentID(primary key), AssesmentName, Weight

# 3 The relational schema





# 4 The set of database statements

We have 9 tables include

## a. Programmings

### Code:

```
create table Programmings (  
  ProgrammingID varchar(8)primary key,  
  ProgrammingName nvarchar(255)  
)
```

### Example:

ProgrammingID	ProgrammingName
LG	Logistic
MA	Marketing
SE	Software engineer

# 4 The set of database statements

## b. Students

### Code:

```
create table Students (  
  StudentID varchar(8) primary key,  
  FullName nvarchar(255),  
  ProgrammingID varchar(8),  
  DOB date,  
  PhoneNumber varchar(10),  
  [Address] nvarchar(300),  
  Gender bit default 1,  
  Email nvarchar(255)  
  foreign key (ProgrammingID) references  
  Programmings(ProgrammingID)  
)
```

### Example:

StudentID	FullName	ProgrammingID	DOB	PhoneNumber	Address	Gender	Email
GS187273	AnhND	MA	2003-12-02	0784231565	Bac Ninh	1	anhnd@gmail.com
HE127365	HieuNH	LG	1998-03-26	0423159876	Hai Duong	1	hieunh@gmail.com
HE160237	NghiaTD	MA	2001-06-11	0457269452	Quy Nhon	1	nghiatd@gmail.com
HE160456	AnhTN	MA	2002-02-17	0342756892	Da Nang	1	anhntn@gmail.com
HE160710	TuanLD	SE	2002-05-11	0348737721	Ha Noi	1	tuanld@gmail.com
HE160891	DuongNH	SE	2002-06-12	0986806300	Hai Phong	1	duongnh@gmail.com
HE169281	VinhNH	SE	2002-06-12	0523154562	Ha Noi	1	vinhnh@gmail.com
HE178273	HaiVD	SE	1999-09-02	0123452158	Ninh Binh	1	haivd@gmail.com
HS152347	QuynhNT	LG	2001-06-11	0951234567	Hai Duong	0	quynhnt@gmail.com
HS162936	HieuNT	LG	1999-09-02	0985423621	Bac Ninh	1	hieunt@gmail.com
HS165234	TrangDT	MA	2001-06-11	0876451234	Bac Ninh	0	trangdt@gmail.com
HS169256	ThaoVTT	LG	2002-12-29	0245866523	Ha Noi	0	thaovtt@gmail.com

## 4 The set of database statements

### c. Courses

#### Code:

```
create table Courses(  
  CourseID varchar(8) primary key,  
  CourseName nvarchar(255),  
)
```

#### Example:

CourseID	CourseName
CSD	Datastructure and algorithm
DBI	Database
JDP	Japanese
LGA	Logictics Advance
LGB	Logictics Begin
LGM	Logictics Master
MAD	MATH2
MAE	MATH1
MAS	MATH3
MKA	Maketing Advance
MKB	Maketing Begin
MKM	Maketing Master
PRF	C Language
PRJ	JAVA WED
PRO	OOP
SSG	Team work
SSL	Soft skill



## 4 The set of database statements

### d. Contain

#### Code:

```
create table Contain (  
  ProgrammingID varchar(8),  
  CourseID varchar(8),  
  Semester int,  
  foreign key (CourseID) references Courses(CourseID),  
  foreign key (ProgrammingID) references  
  Programmings(ProgrammingID)  
)
```

#### Example:

ProgrammingID	CourseID	Semester
SE	JDP	1
SE	CSD	2
SE	DBI	3
SE	MAE	1
SE	MAD	2
SE	MAS	3
MA	MKB	1
MA	MKA	2
MA	MKM	3
MA	SSL	1
MA	SSG	2
MA	MAS	3
LG	LGB	1
LG	LGA	2
LG	LGM	3

## 4 The set of database statements

### e. Lecturers

#### Code:

```
create table Lecturers (  
  LecturerID varchar(8) primary key,  
  LecName nvarchar(255),  
  LecDOB date,  
  LecPhoneNumber varchar(10),  
  LecEmail nvarchar(255)  
)
```

#### Example:

LecturerID	LecName	LecDOB	LecPhoneNumber	LecEmail
LLG1	ChiLP	1989-04-15	0452136789	chilp@gmail.com
LLG2	AnhBN	1979-04-23	0421563789	anhbn@gmail.com
LLG3	PhuongVT	1988-02-20	0452315648	phuongvt@gmail.com
LMK1	TuanVM	1990-04-05	0125487964	tuanvm@gmail.com
LMK2	LinhCD	1991-05-07	0127546125	linhcd@gmail.com
LMK3	HaPTH	1988-02-20	0963215421	hapth@gmail.com
LSE1	SonNT	1988-02-20	0123456789	sonnt@gmail.com
LSE2	HaiLT	1986-03-20	0123654987	hailt@gmail.com
LSE3	ThoPN	1990-04-05	0487623154	thopn@gmail.com

## 4 The set of database statements

### f. Groups

#### Code:

```
create table Groups(  
  GroupID int primary key,  
  CourseID varchar(8),  
  LecturerID varchar(8),  
  foreign key (CourseID) references Courses(CourseID),  
  foreign key (LecturerID) references Lecturers(LecturerID)  
)
```

#### Example:

GroupID	CourseID	LecturerID
1	CSD	LSE3
2	JDP	LSE2
3	PRO	LSE2

### g. MarkReports

#### Code:

```
create table MarkReports(  
  MarkReportID int primary key,  
  GroupID int,  
  StudentID varchar(8),  
  TotalMark float,  
  StatusResult bit,  
  foreign key (GroupID) references Groups(GroupID),  
  foreign key (StudentID) references Students(StudentID)  
)
```

#### Example:

MarkReportID	GroupID	StudentID	TotalMark	StatusResult
1	1	HE160710	NULL	NULL
2	1	HE160891	NULL	NULL
3	8	HE160710	NULL	NULL
4	8	HE160891	NULL	NULL
5	3	HE160710	NULL	NULL
6	3	HE160891	NULL	NULL
7	2	HE178273	NULL	NULL

## 4 The set of database statements

### h. Assessments

#### Code:

```
create table Assessments(  
  AssessmentID varchar(8) primary key,  
  AssessmentName varchar(50),  
  [Weight] float  
)
```

#### Example:

AssessmentID	AssessmentName	Weight
ACSD	ASSIGNMENT CSD	0.3
ADBI	ASSIGNMENT DBI	0.3
AJDP	ASSIGNMENT JDP	0.3

### i. MarkReportDetails

#### Code:

```
create table MarkReportDetails(  
  MarkReportID int,  
  AssessmentID varchar(8),  
  Score float,  
  foreign key (MarkReportID) references  
  MarkReports(MarkReportID),  
  foreign key (AssessmentID) references  
  Assessments(AssessmentID)  
)
```

#### Example:

MarkReportID	AssessmentID	Score
1	ACSD	8
1	PCSD	8.2
1	FCSD	7.8

## 5. Ten queries that demonstrate the usefulness of the database

a. select all students and display StudentID, Fullname, DOB, Address and ProgrammingName regist programming has ProgrammingID is 'SE' (Innerjoin).

### code

```
select s.StudentID ,s.FullName,s.DOB, s.[Address], p.ProgrammingName from
Students s inner join Programmings p on s.ProgrammingID = p.ProgrammingID
where p.ProgrammingID = 'SE'
```

### Result

StudentID	FullName	DOB	Address	ProgrammingName
HE160710	TuanLD	2002-05-11	Ha Noi	Software engineer
HE160891	DuongNH	2002-06-12	Hai Phong	Software engineer
HE169281	VinhNH	2002-06-12	Ha Noi	Software engineer
HE178273	HaiVD	1999-09-02	Ninh Binh	Software engineer

b. Display each programming has information about ProgrammingID, ProgrammingName, and NumberOfStudent by programming (aggregate functions).

### code

```
select p.ProgrammingID,p.ProgrammingName,COUNT(s.StudentID) as
NumberOfStudent from
Students s inner join Programmings p on s.ProgrammingID = p.ProgrammingID
group by p.ProgrammingID,p.ProgrammingName
```

### Result

ProgrammingID	ProgrammingName	NumberOfStudent
LG	Logistic	4
MA	Marketing	4
SE	Software engineer	4

## 5. Ten queries that demonstrate the usefulness of the database

c. Display all information of each student and sort them by FullName by asc, if two people have same name order by Address desc (ORDER BY).

### code

```
select * from Students order by FullName asc, [Address] desc
```

### Result

StudentID	FullName	ProgrammingID	DOB	PhoneNumber	Address	Gender	Email
GS187273	AnhND	MA	2003-12-02	0784231565	Bac Ninh	1	anhnd@gmail.com
HE160456	AnhTN	MA	2002-02-17	0342756892	Da Nang	1	anhntn@gmail.com
HE160891	DuongNH	SE	2002-06-12	0986806300	Hai Phong	1	duongnh@gmail.com
HE178273	HaiVD	SE	1999-09-02	0123452158	Ninh Binh	1	haivd@gmail.com
HE127365	HieuNH	LG	1998-03-26	0423159876	Hai Duong	1	hieunh@gmail.com
HS162936	HieuNT	LG	1999-09-02	0985423621	Bac Ninh	1	hieunt@gmail.com
HE160237	NghiaTD	MA	2001-06-11	0457269452	Quy Nhon	1	nghiatd@gmail.com
HS152347	QuynhNT	LG	2001-06-11	0951234567	Hai Duong	0	quynhnt@gmail.com
HS169256	ThaoVTT	LG	2002-12-29	0245866523	Ha Noi	0	thaovtt@gmail.com
HS165234	TrangDT	MA	2001-06-11	0876451234	Bac Ninh	0	trangdt@gmail.com
HE160710	TuanLD	SE	2002-05-11	0348737721	Ha Noi	1	tuanld@gmail.com
HE169281	VinhNH	SE	2002-06-12	0523154562	Ha Noi	1	vinhnh@gmail.com

d. Display LectureID and LectureName who supervise more than two group (GROUP BY and HAVING clauses).

### code

```
select l.LecturerID,l.LecName,COUNT(g.CourseID) as NumberOfCourse from  
Groups g inner join Lecturers l on l.LecturerID = g.LecturerID  
group by l.LecturerID,l.LecName  
having COUNT(g.CourseID) > 2
```

### Result

LecturerID	LecName	NumberOfCourse
LSE1	SonNT	3
LSE2	HaiLT	3
LSE3	ThoPN	3

## 5. Ten queries that demonstrate the usefulness of the database

e. Display all information of each student and sort them by FullName by asc, if two people have same name order by Address desc

### code

```
select * from Students order by FullName asc, [Address] desc
```

### Result

StudentID	FullName	ProgrammingID	DOB	PhoneNumber	Address	Gender	Email
GS187273	AnhND	MA	2003-12-02	0784231565	Bac Ninh	1	anhnd@gmail.com
HE160456	AnhTN	MA	2002-02-17	0342756892	Da Nang	1	anhntn@gmail.com
HE160891	DuongNH	SE	2002-06-12	0986806300	Hai Phong	1	duongnh@gmail.com
HE178273	HaiVD	SE	1999-09-02	0123452158	Ninh Binh	1	haivd@gmail.com
HE127365	HieuNH	LG	1998-03-26	0423159876	Hai Duong	1	hieunh@gmail.com
HS162936	HieuNT	LG	1999-09-02	0985423621	Bac Ninh	1	hieunt@gmail.com
HE160237	NghiaTD	MA	2001-06-11	0457269452	Quy Nhon	1	nghiatd@gmail.com
HS152347	QuynhNT	LG	2001-06-11	0951234567	Hai Duong	0	quynhnt@gmail.com
HS169256	ThaoVTT	LG	2002-12-29	0245866523	Ha Noi	0	thaoqtt@gmail.com
HS165234	TrangDT	MA	2001-06-11	0876451234	Bac Ninh	0	trangdt@gmail.com
HE160710	TuanLD	SE	2002-05-11	0348737721	Ha Noi	1	tuanld@gmail.com
HE169281	VinhNH	SE	2002-06-12	0523154562	Ha Noi	1	vinhnh@gmail.com

f. Display LectureID and LectureName who supervise a group that has more than two people

### code

```
select l.LecturerID,l.LecName from
(select COUNT(m.StudentID) as NumberOfStudent, g.LecturerID from Groups g
inner join MarkReports m on g.GroupID = m.GroupID
group by m.GroupID,g.LecturerID) as NumberOfStudent, Lecturers as l
where NumberOfStudent.NumberOfStudent > 2 and l.LecturerID =
NumberOfStudent.LecturerID;
```

### Result

LecturerID	LecName
LSE3	ThoPN
LMK3	HaPTH
LLG2	AnhBN

## 5. Ten queries that demonstrate the usefulness of the database

g. Display all couple of students has the same address(Self join).

### code

```
select s1.[Address],s.FullName,s1.FullName from Students s
inner join Students s1 on s.StudentID > s1.StudentID and s.[Address] = s1.[Address]
```

### Result

Address	FullName	FullName
Ha Noi	VinhNH	TuanLD
Hai Duong	QuynhNT	HieuNH
Bac Ninh	HieuNT	AnhND
Bac Ninh	TrangDT	AnhND
Bac Ninh	TrangDT	HieuNT
Ha Noi	ThaoVTT	TuanLD
Ha Noi	ThaoVTT	VinhNH

h. Display all students who have studentid start with 'HE' (partial matching in the WHERE clause).

### code

```
select * from Students
where StudentID like '%HE[0-9][0-9][0-9][0-9][0-9][0-9]%'
```

### Result

StudentID	FullName	ProgrammingID	DOB	PhoneNumber	Address	Gender	Email
HE127365	HieuNH	LG	1998-03-26	0423159876	Hai Duong	1	hieunh@gmail.com
HE160237	NghiaTD	MA	2001-06-11	0457269452	Quy Nhon	1	nghiatd@gmail.com
HE160456	AnhTN	MA	2002-02-17	0342756892	Da Nang	1	anhtn@gmail.com
HE160710	TuanLD	SE	2002-05-11	0348737721	Ha Noi	1	tuanld@gmail.com
HE160891	DuongNH	SE	2002-06-12	0986806300	Hai Phong	1	duongnh@gmail.com
HE169281	VinhNH	SE	2002-06-12	0523154562	Ha Noi	1	vinhnh@gmail.com
HE178273	HaiVD	SE	1999-09-02	0123452158	Ninh Binh	1	haivd@gmail.com



## 5. Ten queries that demonstrate the usefulness of the database

### i. Display all TotalMark of all students

#### code

```
select m.StudentID,s.FullName,m.GroupID,SUM(md.Score * a.[Weight]) as TotalMark from
MarkReports m inner join MarkReportDetails md on m.MarkReportID = md.MarkReportID
inner join Assesments a on a.AssesmentID = md.AssesmentID
inner join Students s on s.StudentID = m.StudentID
Group by m.MarkReportID,m.StudentID,s.FullName,m.GroupID
```

#### Result

StudentID	FullName	GroupID	TotalMark
HE160710	TuanLD	1	7.98
HE160891	DuongNH	1	4.8
HE160710	TuanLD	8	7.4
HE160891	DuongNH	8	9.36
HE160710	TuanLD	3	4.2
HE160891	DuongNH	3	5.8
HE178273	HaiVD	2	4.35
HE178273	HaiVD	7	5.6
HE178273	HaiVD	4	5.66
HE169281	VinhNH	6	7.48
HE169281	VinhNH	9	4.76
HE169281	VinhNH	5	5.07
GS187273	AnhND	10	7
GS187273	AnhND	13	6.47
HE160237	NghiaTD	10	4.9
HE160237	NghiaTD	13	5.8
HE160456	AnhTN	11	8.24
HE160456	AnhTN	14	6.3
HS165234	TrangDT	12	5.6

HS165234	TrangDT	9	7
HE127365	HieuNH	15	6.86
HE127365	HieuNH	13	5.1
HS169256	ThaoVTT	16	7.81
HS169256	ThaoVTT	14	8.49
HS162936	HieuNT	16	7.52
HS162936	HieuNT	14	5.55
HS152347	QuynhNT	17	5.64
HS152347	QuynhNT	9	5.62

## 5. Ten queries that demonstrate the usefulness of the database

k. Update all total mark of all student by cursor

### code

```
declare @id int
declare @total float

declare point cursor for
(select md.MarkReportID, SUM(a.
[Weight]*md.Score) from
Assesments a inner join MarkReportDetails
md on a.AssesmentID = md.AssesmentID
Group by md.MarkReportID)
open point
FETCH NEXT FROM point
into @id, @total
while @@FETCH_STATUS = 0
begin
    update MarkReports
    set TotalMark = @total
    where @id = MarkReportID
    FETCH NEXT FROM point
    into @id, @total
end
CLOSE point
DEALLOCATE point

select * from MarkReports
```

### Result

MarkReportID	GroupID	StudentID	TotalMark	StatusResult
1	1	HE160710	7.98	NULL
2	1	HE160891	4.8	NULL
3	8	HE160710	7.4	NULL
4	8	HE160891	9.36	NULL
5	3	HE160710	4.2	NULL
6	3	HE160891	5.8	NULL
7	2	HE178273	4.35	NULL
8	7	HE178273	5.6	NULL
9	4	HE178273	5.66	NULL
10	6	HE169281	7.48	NULL
11	9	HE169281	4.76	NULL
12	5	HE169281	5.07	NULL
13	10	GS187273	7	NULL
14	13	GS187273	6.47	NULL
15	10	HE160237	4.9	NULL
16	13	HE160237	5.8	NULL
17	11	HE160456	8.24	NULL
18	14	HE160456	6.3	NULL
19	12	HS165234	5.6	NULL
20	9	HS165234	7	NULL
21	15	HE127365	6.86	NULL
22	13	HE127365	5.1	NULL
23	16	HS169256	7.81	NULL
24	14	HS169256	8.49	NULL
25	16	HS162936	7.52	NULL
26	14	HS162936	5.55	NULL
27	17	HS152347	5.64	NULL
28	9	HS152347	5.62	NULL

## 6. The trigger, store procedure and view

### a. Store Poseudo

We create a store pseudo to support we to search total mark of a student by course id and student id

#### Code

```
create proc TotalOfStudent
@idStudent varchar(8),
@idCourse varchar(8)
as
begin
declare @total float
select @total = m.TotalMark from MarkReports m inner join Groups g on
m.GroupID = g.GroupID
where m.StudentID = @idStudent and g.CourseID = @idCourse
print 'Total mark is:' + CAST(@total as varchar(8))
end
```

```
exec TotalOfStudent @idStudent = 'HE160710', @idCourse = 'CSD'
```

```
Total mark is:7.98
```

```
Completion time: 2022-07-18T02:51:50.5132062+07:00
```

## 6. The trigger, store procedure and view

### b. Trigger

We create a trigger to support we, whenever we want to update, delete or insert a new record in MarkReportDetail, total mark in MarkReport is automatically updated

#### Code

```
create trigger insert_update_delete_score
on MarkReportDetails
after insert, update, delete
as
begin
    update MarkReports
    set TotalMark = TotalMark + tbs.UpGrade
    from MarkReports m
    inner join
    (select i.MarkReportID,SUM(i.Score*a.Weight) as UpGrade from inserted i
    inner join Assesments a on i.AssesmentID = a.AssesmentID
    group by i.MarkReportID) as tbs
    on m.MarkReportID = tbs.MarkReportID

    update MarkReports
    set TotalMark = TotalMark - tbs2.DownGrade
    from MarkReports m
    inner join
    (select d.MarkReportID,SUM(d.Score*a.Weight) as DownGrade from deleted d
    inner join Assesments a on d.AssesmentID = a.AssesmentID
    group by d.MarkReportID) as tbs2
    on m.MarkReportID = tbs2.MarkReportID
end
```

```
(1 row affected)

(1 row affected)

(1 row affected)
Total mark is:6.78
```

## 6. The trigger, store procedure and view

### c. Trigger

We create a view to limit the students only see the view we created or we can use the view as a completable table

#### Code

**create view student\_mark\_status**

**as**

**select m.GroupID,m.StudentID,m.TotalMark, IIF(m.TotalMark>=5,'Pass','Not Pass') as [Status] from MarkReports m**

GroupID	StudentID	TotalMark	Status
1	HE160710	6.78	Pass
1	HE160891	4.2	Not Pass
8	HE160710	7.4	Pass
8	HE160891	9.36	Pass
3	HE160710	4.2	Not Pass
3	HE160891	5.8	Pass
2	HE178273	4.35	Not Pass
7	HE178273	5.6	Pass
4	HE178273	5.66	Pass
6	HE169281	7.48	Pass
9	HE169281	4.76	Not Pass
5	HE169281	5.07	Pass
10	GS187273	7	Pass
13	GS187273	6.47	Pass
10	HE160237	4.9	Not Pass
13	HE160237	5.8	Pass
11	HE160456	8.24	Pass
14	HE160456	6.3	Pass
12	HS165234	5.6	Pass

9	HS165234	7	Pass
15	HE127365	6.86	Pass
13	HE127365	5.1	Pass
16	HS169256	7.81	Pass
14	HS169256	8.49	Pass
16	HS162936	7.52	Pass
14	HS162936	5.55	Pass
17	HS152347	5.64	Pass
9	HS152347	5.62	Pass

## 6. The trigger, store procedure and view

### d. Index

We create a index to bost the speed of finding, updating , inserting or deleting. So we can see in this database we spend much time to finding many thing in Student. Therefore we create an index in Colum StudentID

#### Code

```
create nonclustered index index_ma on Students(StudentID)
```



## ● Thank you!

Lê Đình Tuân - HE160710

DBI202 - SE1647

Lecturer: Ngô Tùng Sơn