Enhanced Enlistment System: Course Categorization and Prerequisite Management

A Project

Presented to the Faculty of the

School of Management and Information Technology

Information Systems

De La Salle – College of Saint Benilde

Manila

In Partial Fulfillment

of the Requirements for the subject

INFORMATION MANAGEMENT

Submitted By:

Cudal, Princess Shakira
Estor, Rezzylaine
Jang, Dae In
Sabarre, Christian S.

DR. JOSELITO EDUARD E. GOH
Professor

April 3, 2025

TABLE OF CONTENTS

RUBRICS FOR
DATABASE CONCEPT – FINAL PROJECT

Name of Students	Project Rating (Project)	Presentation Rating (Final Exam)
CUDAL, PRINCESS SHAKIRA, T.		
ESTOR, REZZYLAINE, R.		
JANG, DAE IN A.		
SABARRE, CHRISTIAN		

PROJECT CRITERIA	Highly Implemented	Satisfactory Implemented	Fairly Implemented	Partially Implemented	Not Implemented
Table Design Do the tables have appropriate fields, keys, and other constraints needed?	20	17	14	11	8
Database Normalization Do the tables apply 1st, 2nd and 3rd normal forms with correct relationships?	20	17	14	11	8
Database Functionality Does the database include appropriate tables to contain enough information with queries that support the intended functionalities presented in the wireframes?	20	17	14	11	8
Query Accuracy Do the queries produce results as expected?	20	17	14	11	8
Documentation Are the parts of the required document present?	20	17	14	11	8
TOTAL					
PRESENTATION CRITERIA	**Highly Achieved**	**Satisfactory Achieved**	**Fairly Achieved**	**Partially Achieved**	**Not Achieved**
Voice Quality, Delivery, & Presentation Content Explains the presentation clearly and understandably.	30	26	22	18	14
Question and Answer Answers the questions correctly with confidence.	50	43	36	29	22
Time Management Finishes within the prescribe time with appropriate pacing	20	17	14	11	8
TOTAL					
REMARKS					

A. INTRODUCTION

In De La Salle-College of Saint Benilde, the enlistment process is a crucial step in which students select their subjects for the upcoming term. However, the current system presents all available subjects without filtering based on the student's curriculum flowchart. This means that students must manually check which courses they are eligible for, often cross-referencing their academic records and prerequisites. This setup can lead to confusion, misclicks, and errors in enlistment, as students may accidentally enlist in subjects they are not supposed to take yet. Additionally, even if a student fails a prerequisite subject, the system still allows them to enroll in the next-level course, creating potential academic issues that may only be corrected later.

To address these challenges, our proposed solution introduces a subject filtering system that streamlines the enlistment process. Instead of displaying all available courses, the system will also filter subjects that a student is eligible for based on their academic progress and curriculum flowchart together with the other courses available. This ensures that students can easily identify and enroll in the appropriate subjects without the risk of enrolling in courses for which they are not qualified. By implementing this feature, we aim to enhance the efficiency of the enlistment system, reduce errors, and provide a more user-friendly experience for students navigating their academic journey.

B. PROJECT DESCRIPTION

Proposed Solution

The enhanced enlistment system introduces filtering queries that:

1. **Automate Curriculum-Based Course Selection**
   - Filters available courses according to:
     i. Program-specific curriculum requirements
     ii. Student academic standing (year level)
     iii. Term sequencing recommendations
2. **Enforce Prerequisite Requirements**
   - Validates course eligibility through:
     i. Automated grade checks for prerequisite courses
     ii. Minimum grade requirements
3. **User Interface and User Experience**
   - Enhance the existing user interface by:
     i. Improving readability and ease of navigation
     ii. Categorizing courses based on:

   a. Suggested courses (from the user's flowchart).
   b. Advanced courses (that the user may take early).
   c. Retake courses (courses the student must repeat).

C. SCOPE AND LIMITATIONS OF THE PROJECT

SCOPE:

The Enhanced Enlistment System automates course selection by filtering subjects based on a student's curriculum flowchart and prerequisite requirements. The database structure ensures that students can only view and enlist in courses they are eligible for, reducing manual checking and enrollment errors.

The Prerequisite Table enforces subject dependencies, preventing students from enrolling in courses without meeting the necessary prerequisite grades, which are validated using the Grades Table. The Enlistment Table records course registrations and tracks enrollment status, while the Term Table manages enlistments per academic term.

By leveraging these database components, the system enhances the enlistment process by minimizing errors, enforcing prerequisite rules, and ensuring students enroll in the correct subjects based on their academic progress.

View Courses that could be enlisted for the next term based on the BS - IS Flowchart. Courses available for enlistment would be Courses the student has already taken before and failed, and courses that the student is eligible for the next term based on predefined prerequisites. The student would be able to add these courses to an enlistment table, they could save their courses as long as it is not yet submitted. The student could delete initial and saved courses from their enlistment table and add. However, once the courses are submitted the students should not be able to modify their courses

TABLES AND COLUMNS

Students	Contains student records with StudentID, ProgramID, and CurriculumID, linking students to their academic program and curriculum.
Program	Stores details of academic programs (ProgramID, ProgramName, ProgramCode) that students can enroll in.

Curriculum	Represents a curriculum structure (CurriculumID, ProgramID, YearEffective, TotalUnitsRequired), defining the courses required for a specific program.
CurriculumCourses	Links curricula to courses (CurriculumCourseID, CurriculumID, CourseID, TermRecommended), specifying which courses belong to a curriculum and the recommended term to take them.
Courses	Contains course details (CourseID, CourseCode, CourseName, Units, Department).
Prerequisite	Defines prerequisite relationships between courses (PrereqID, CourseID, RequiredCourseID, PassingGrade), ensuring students meet requirements before taking certain courses.
Term	Represents academic terms (TermID, AcademicYear, StartDate, EndDate).
Enlistment	Tracks student course registration (EnlistmentID, StudentID, CourseID, TermID, EnlistmentStatus).
StudentsCourseGrade	Stores student grades for courses (StudentCourseGradeID, StudentID, CurriculumCourseID, Grade, Status, TermID).

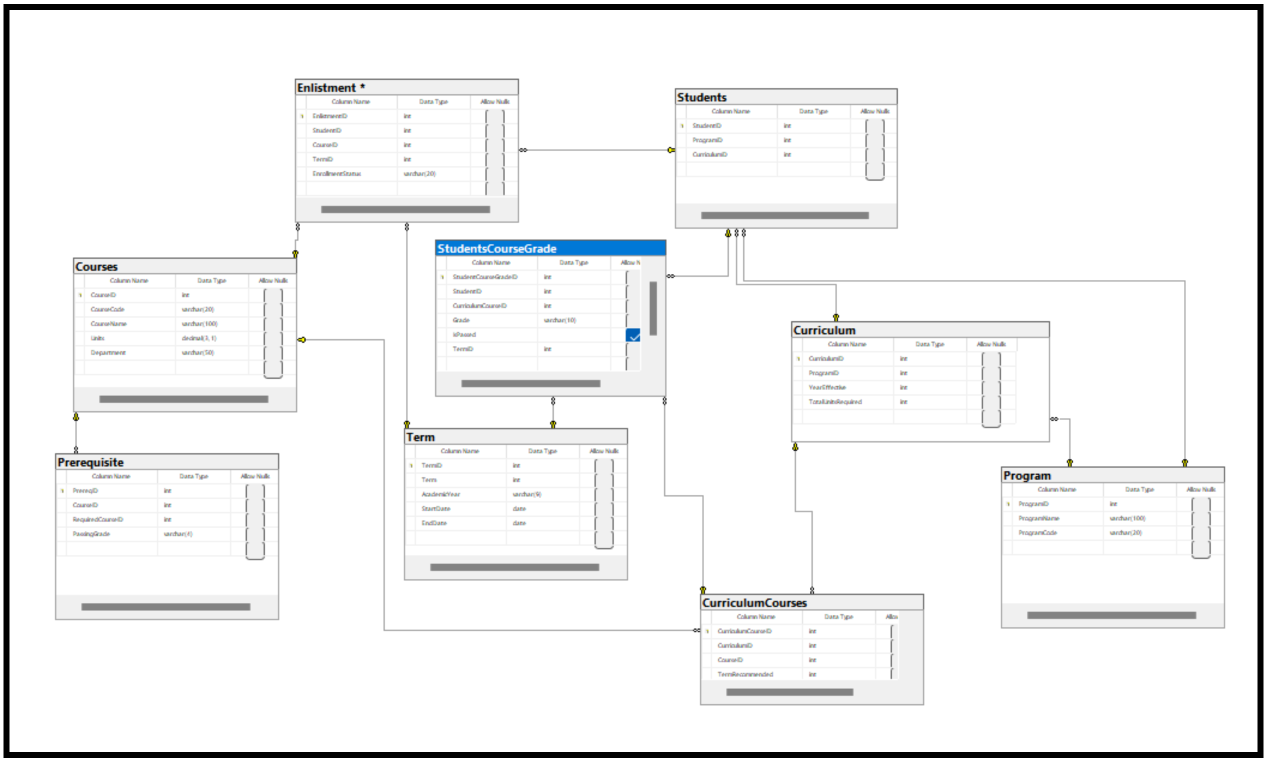
LIMITATIONS:

The current enlistment implementation in De La Salle-College of Saint Benilde exhibits significant limitations:

1. **Uncertain Course Eligibility Due to Delayed Release of Grades**
    - ○ If professors do not release final grades before enlistment, students may unknowingly enroll in courses they have already failed or passed.
    - ○ This can result in forced course drops, leading to potential financial penalties.

2. **System Overload and Technical Issues**
   - High traffic during enlistment may cause system slowdowns or crashes.
   - Students' internet connection issues can affect their enrollment process, but the system cannot control these factors.
3. **Grade Release and Enlistment Period Mismatch**
   - If the institution does not adjust the schedule to align the grade release period with the enlistment period, the Enhanced Enlistment System will not function as intended. Without synchronized timing, the system cannot validate whether students have failed prerequisite courses before they enroll in related subjects, rendering the system ineffective and reducing its value;
   - This limitation emphasizes that **the system's success relies on the institution addressing this scheduling conflict to ensure its functionality.**
4. **Restricted to BS IS Program Only**
   - Each department has its own prerequisite structure and required courses. Since the system is specifically designed for the BS IS curriculum, it is not compatible with other programs.
   - Expanding the system to all departments would require a larger dataset, but it is currently developed with a focus on the BS IS program.
5. **Credit Unit Counting Limitation**
   - The system is unable to track the total enrolled units, which may lead to students exceeding or falling short of the required max unit.

D. DATABASE DIAGRAM



Enlistment *

Column Name	Data Type	Allow Nulls
EnlistmentID	int	□
StudentID	int	□
CourseID	int	□
TermID	int	□
EnrollmentStatus	varchar(20)	□

Students

Column Name	Data Type	Allow Nulls
StudentID	int	□
ProgramID	int	□
CurriculumID	int	□

Courses

Column Name	Data Type	Allow Nulls
CourseID	int	□
CourseCode	varchar(20)	□
CourseName	varchar(100)	□
Units	decimal(3, 1)	□
Department	varchar(50)	□

StudentsCourseGrade

Column Name	Data Type	Allow N
StudentCourseGradeID	int	□
StudentID	int	□
CurriculumCourseID	int	□
Grade	varchar(10)	□
isPassed		☑
TermID	int	□

Curriculum

Column Name	Data Type	Allow Nulls
CurriculumID	int	□
ProgramID	int	□
YearEffective	int	□
TotalUnitsRequired	int	□

Prerequisite

Column Name	Data Type	Allow Nulls
PrereqID	int	□
CourseID	int	□
RequiredCourseID	int	□
PassingGrade	varchar(4)	□

Term

Column Name	Data Type	Allow Nulls
TermID	int	□
Term	int	□
AcademicYear	varchar(9)	□
StartDate	date	□
EndDate	date	□

Program

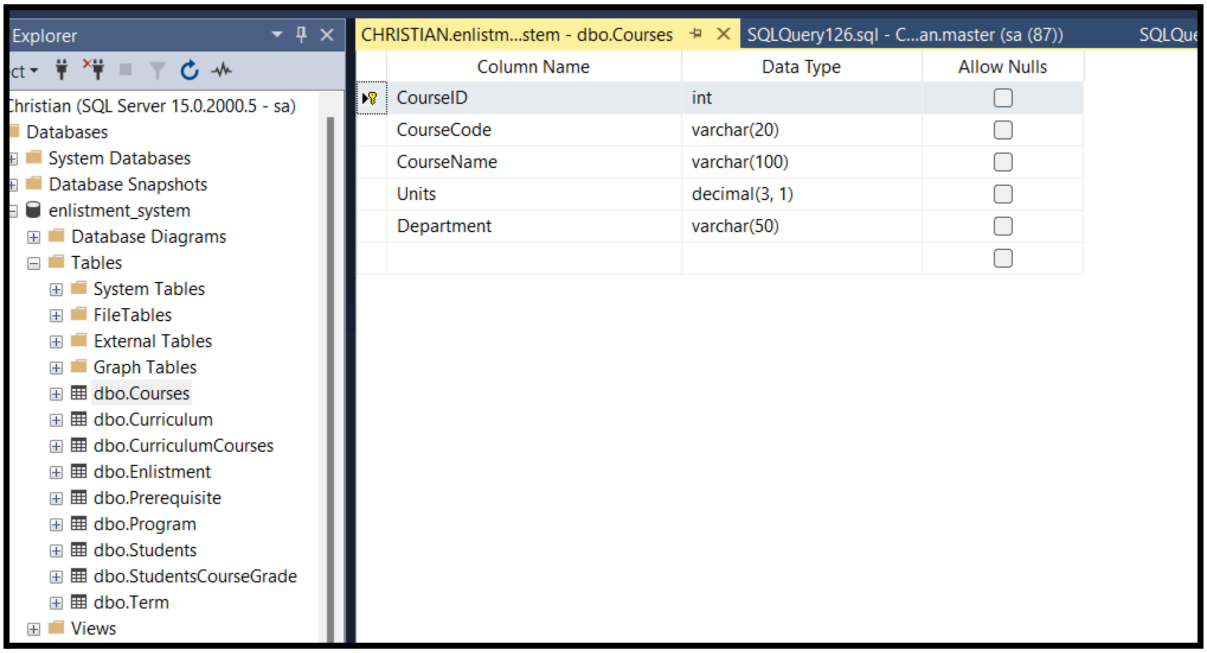
Column Name	Data Type	Allow Nulls
ProgramID	int	□
ProgramName	varchar(100)	□
ProgramCode	varchar(20)	□

CurriculumCourses

Column Name	Data Type	Allow
CurriculumCourseID	int	□
CurriculumID	int	□
CourseID	int	□
TermRecommended	int	□

E. TABLE DESIGN SPECIFICATION

Courses:

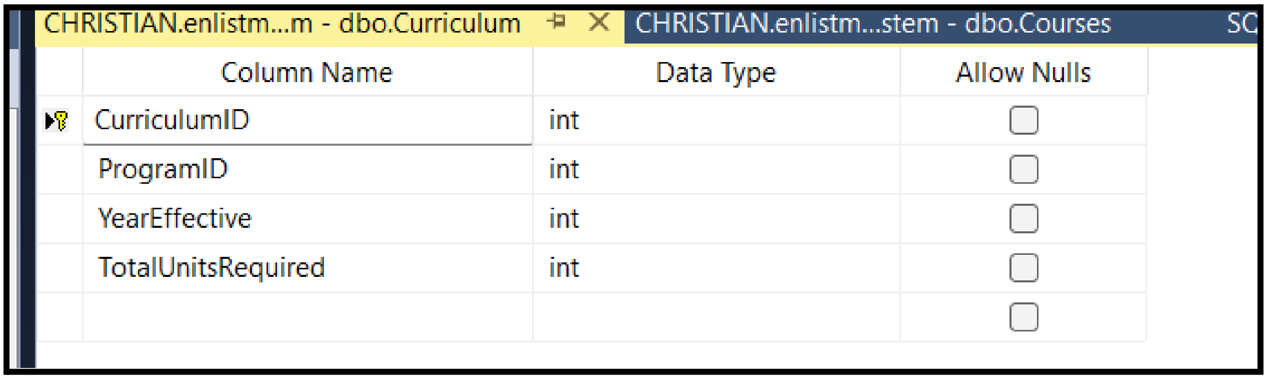


Explorer
ct
CHRISTIAN.enlistm...stem - dbo.Courses
SQLQuery126.sql - C...an.master (sa (87))
SQLQue

hristian (SQL Server 15.0.2000.5 - sa)
- Databases
  - System Databases
  - Database Snapshots
  - enlistment_system
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.Courses
      - dbo.Curriculum
      - dbo.CurriculumCourses
      - dbo.Enlistment
      - dbo.Prerequisite
      - dbo.Program
      - dbo.Students
      - dbo.StudentsCourseGrade
      - dbo.Term
    - Views

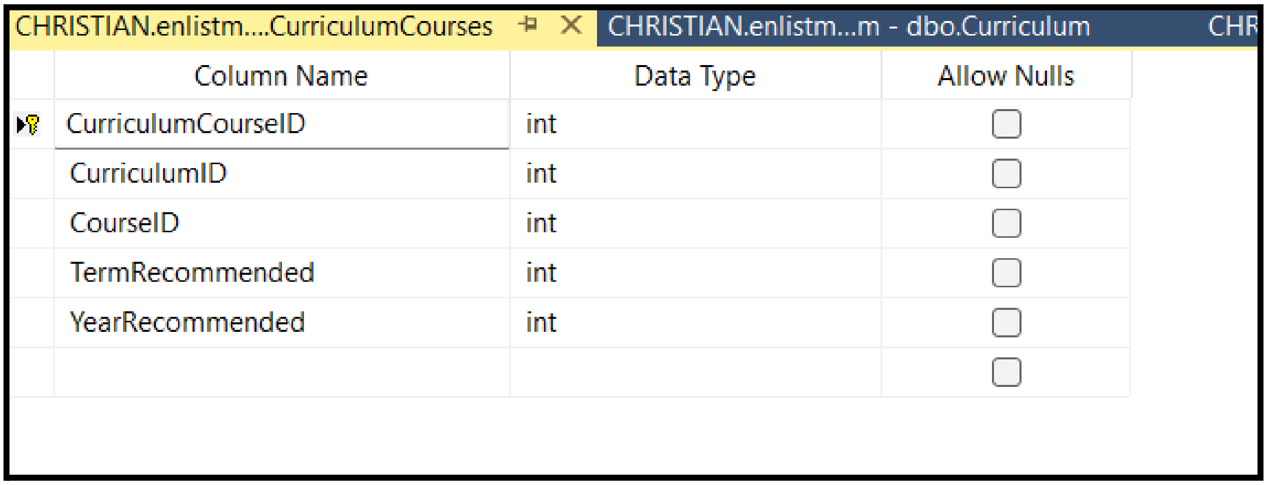
Column Name	Data Type	Allow Nulls
CourseID	int	□
CourseCode	varchar(20)	□
CourseName	varchar(100)	□
Units	decimal(3, 1)	□
Department	varchar(50)	□
		□

Curriculum:


CHRISTIAN.enlistm...m - dbo.Curriculum
CHRISTIAN.enlistm...stem - dbo.Courses
SC

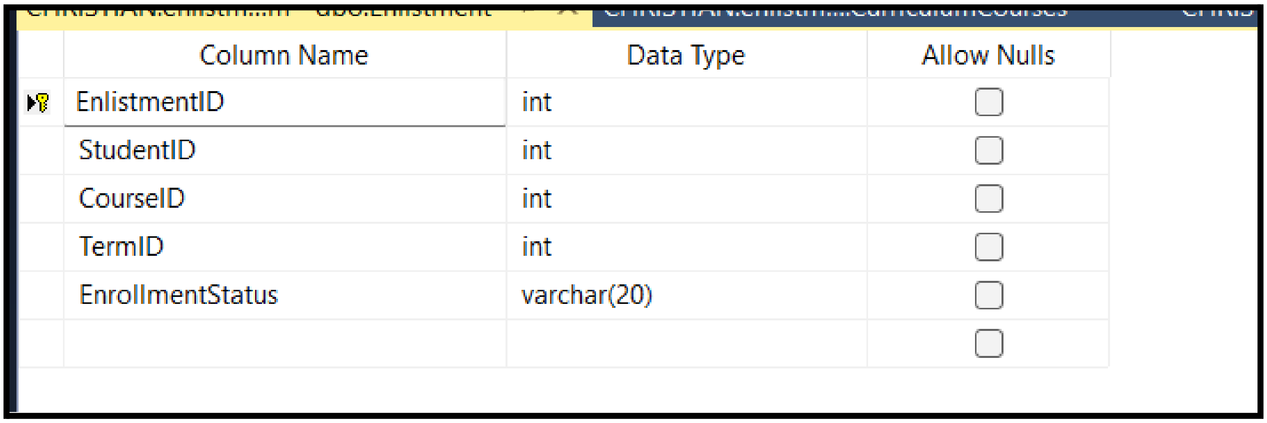
Column Name	Data Type	Allow Nulls
CurriculumID	int	☐
ProgramID	int	☐
YearEffective	int	☐
TotalUnitsRequired	int	☐
		☐

CurriculumCourses:


CHRISTIAN.enlistm....CurriculumCourses
CHRISTIAN.enlistm...m - dbo.Curriculum
CHR

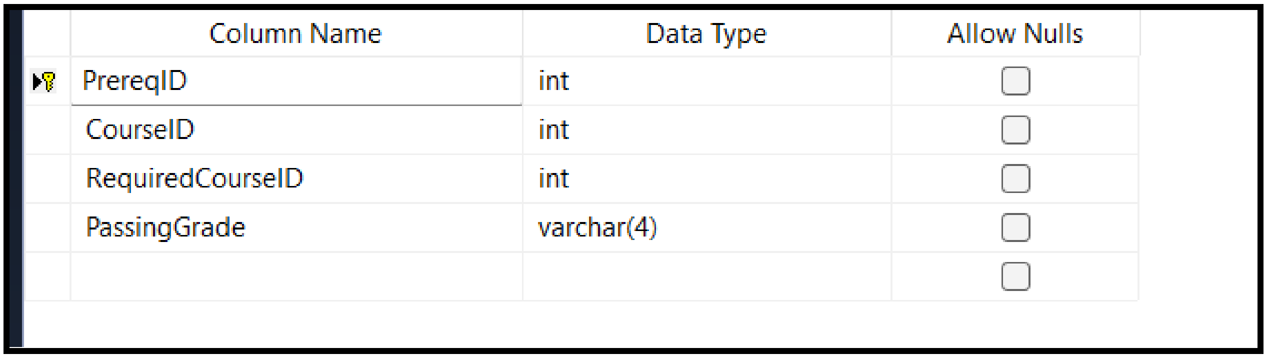
Column Name	Data Type	Allow Nulls
CurriculumCourseID	int	☐
CurriculumID	int	☐
CourseID	int	☐
TermRecommended	int	☐
YearRecommended	int	☐
		☐

Enlistment:


CHRISTIAN.enlistm...m - dbo.Enlistment
CHRISTIAN.enlistm....CurriculumCourses
CHRIS

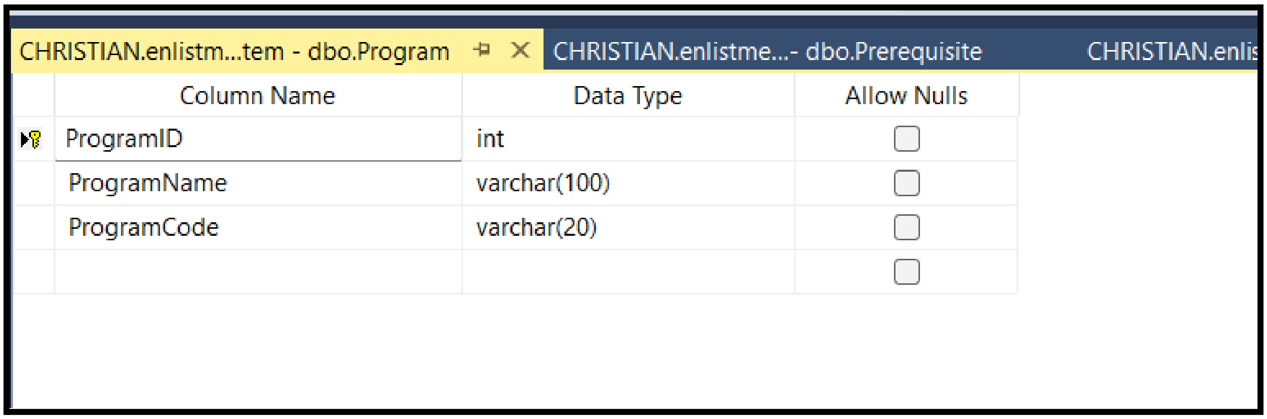
Column Name	Data Type	Allow Nulls
EnlistmentID	int	☐
StudentID	int	☐
CourseID	int	☐
TermID	int	☐
EnrollmentStatus	varchar(20)	☐
		☐

Prerequisite:



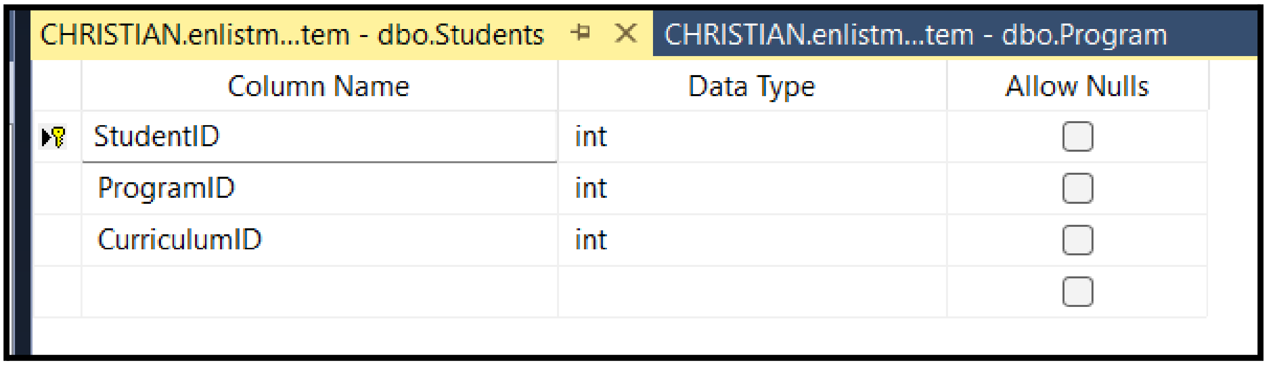
Column Name	Data Type	Allow Nulls
PrereqID	int	☐
CourseID	int	☐
RequiredCourseID	int	☐
PassingGrade	varchar(4)	☐
		☐

Program:


CHRISTIAN.enlistm...tem - dbo.Program
CHRISTIAN.enlistme...- dbo.Prerequisite
CHRISTIAN.enlis

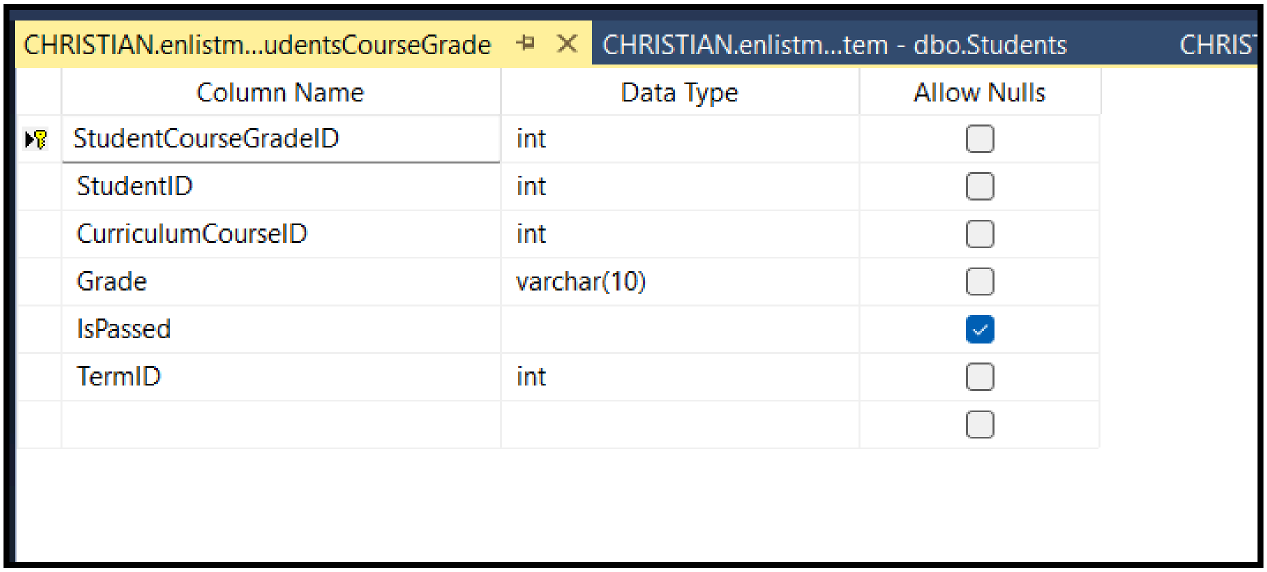
Column Name	Data Type	Allow Nulls
ProgramID	int	☐
ProgramName	varchar(100)	☐
ProgramCode	varchar(20)	☐
		☐

Students:


CHRISTIAN.enlistm...tem - dbo.Students
CHRISTIAN.enlistm...tem - dbo.Program

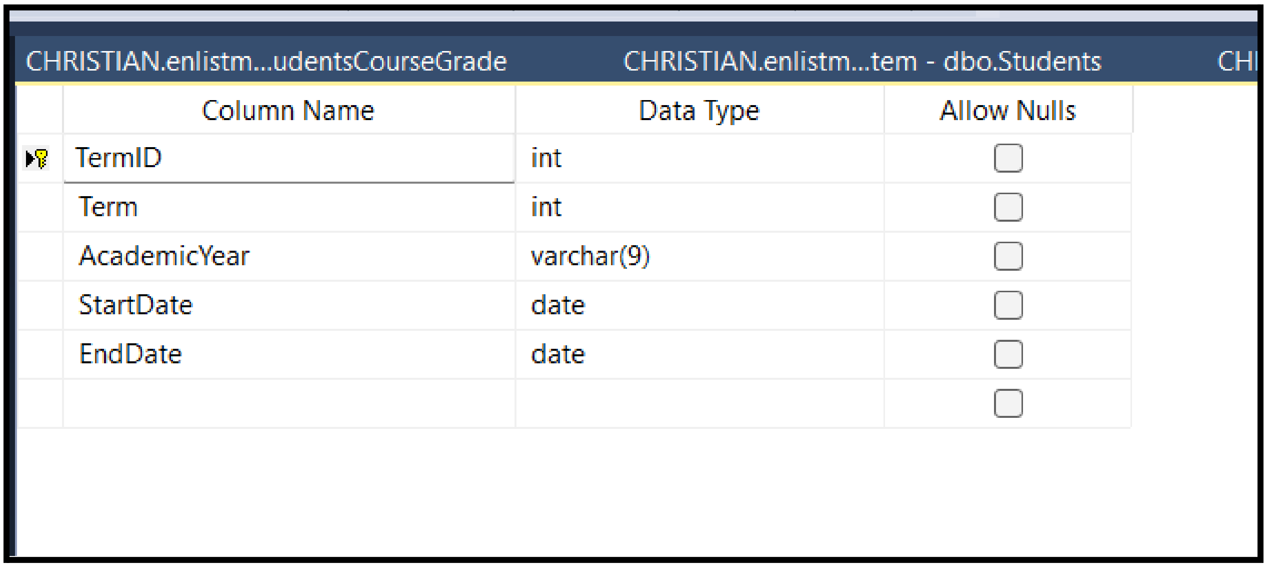
Column Name	Data Type	Allow Nulls
StudentID	int	☐
ProgramID	int	☐
CurriculumID	int	☐
		☐

StudentsCourseGrade:


CHRISTIAN.enlistm...udentsCourseGrade
CHRISTIAN.enlistm...tem - dbo.Students
CHRIS

Column Name	Data Type	Allow Nulls
StudentCourseGradeID	int	☐
StudentID	int	☐
CurriculumCourseID	int	☐
Grade	varchar(10)	☐
IsPassed		☑
TermID	int	☐
		☐

Term:


CHRISTIAN.enlistm...udentsCourseGrade
CHRISTIAN.enlistm...tem - dbo.Students
CHI

Column Name	Data Type	Allow Nulls
TermID	int	☐
Term	int	☐
AcademicYear	varchar(9)	☐
StartDate	date	☐
EndDate	date	☐
		☐

Foreign Key Relationships

Curriculum → Program

Curriculum.ProgramID → Program.ProgramID

CurriculumCourses → Curriculum

CurriculumCourses.CurriculumID → Curriculum.CurriculumID

CurriculumCourses → Courses

CurriculumCourses.CourseID → Courses.CourseID

Enlistment → Students

Enlistment.StudentID → Students.StudentID

Enlistment → Courses

Enlistment.CourseID → Courses.CourseID

Enlistment → Term

Enlistment.TermID → Term.TermID

Prerequisite → Courses (2x)

Prerequisite.CourseID → Courses.CourseID

Prerequisite.RequiredCourseID → Courses.CourseID

Students → Program

Students.ProgramID → Program.ProgramID

Students → Curriculum

Students.CurriculumID → Curriculum.CurriculumID

StudentsCourseGrade → Students

StudentsCourseGrade.StudentID → Students.StudentID

StudentsCourseGrade → CurriculumCourses

StudentsCourseGrade.CurriculumCourseID → CurriculumCourses.CurriculumCourseID

StudentsCourseGrade → Term

StudentsCourseGrade.TermID → Term.TermID

CONSTRAINTS

TABLE	**CONSTRAINTS**	**CONDITION**
CurriculumCourses	TermRecommended	BETWEEN 1 AND 3

	YearRecommended	BETWEEN 1 AND 4
Enlistment	EnlistmentStatus	IN ('APPROVED', 'SUBMITTED', 'SAVED', 'INITIAL')
Prerequisite	CourseID <> RequiredCourseID	No self-prerequisites
StudentsCourseGrade	PassingGrade	Valid grade format
Term	Term	BETWEEN 1 AND 3
StudentsCourseGrade (Computed)	IsPassed	CONVERT([int], CASE WHEN grade is 1.0-4.0 THEN 1 ELSE 0 END)

KEY TABLE RELATIONSHIPS

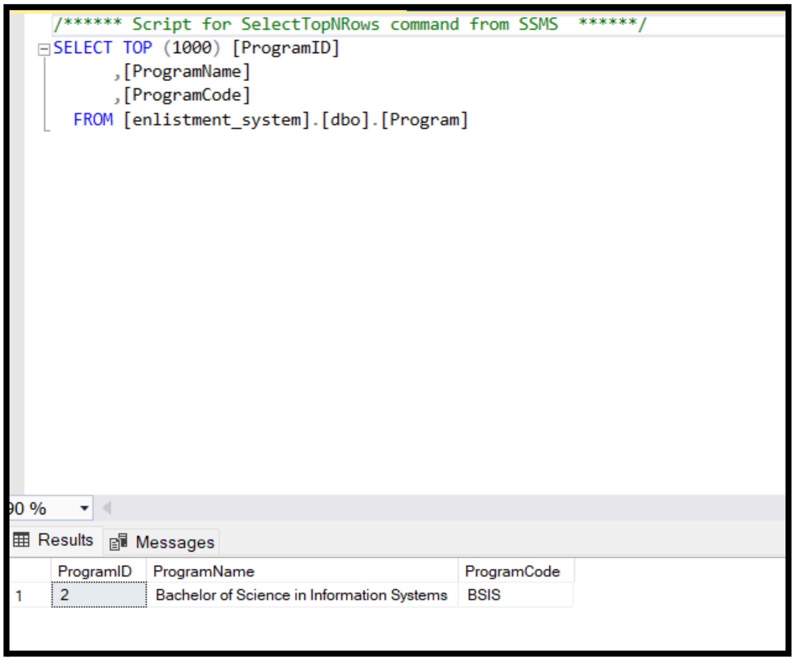
Source Table	**Relationship Type**	**Target Table**	**Relationship Details**
Program	1-to-many	Curriculum	One program can have many curriculums
Program	1-to-many	Students	One program can have many students
Curriculum	1-to-many	CurriculumCourses	One curriculum can have many curriculum courses
Curriculum	1-to-many	Students	One curriculum can have many students
Courses	1-to-many	CurriculumCourses	One course can appear in many curriculum courses
Courses	1-to-many	Enlistment	One course can be enrolled by many students
Courses	Self-referential-many-to-many	Prerequisite	A course can have many prerequisite courses

Students	1-to-many	Enlistment	One student can enroll in many courses
Students	1-to-many	StudentsCourseGrade	One student can have many course grades
Term	1-to-many	Enlistment	One term can have many student enrollments
Term	1-to-many	StduentscourseGrade	One Term can have many course grades for students
CurriculumCourses	1-to-many	StudentsCourseGrade	One curriculum course can have many student course grades

JUNCTION TABLE

Table Name	Type	Entities Connected	Relationship Type
CurriculumCourse	Junction Table	Curriculum ↔ Course	Many-to-Many
Prerequisite	Self-Referential Junction Table	Course ↔ Course (as Prerequisites)	Many-to-Many
Enlistment	Junction Table (Ternary)	Student ↔ Course ↔ Term	3-Way Relationship (Ternary)
StudentsCourseGrade	Junction Table (Ternary)	Student ↔ CurriculumCourse ↔ Term	3-Way Relationship (Ternary)

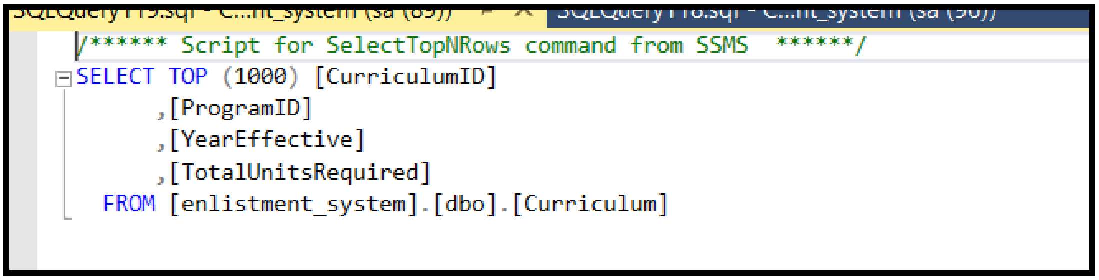
Sample Program



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [ProgramID]
      ,[ProgramName]
      ,[ProgramCode]
  FROM [enlistment_system].[dbo].[Program]
```

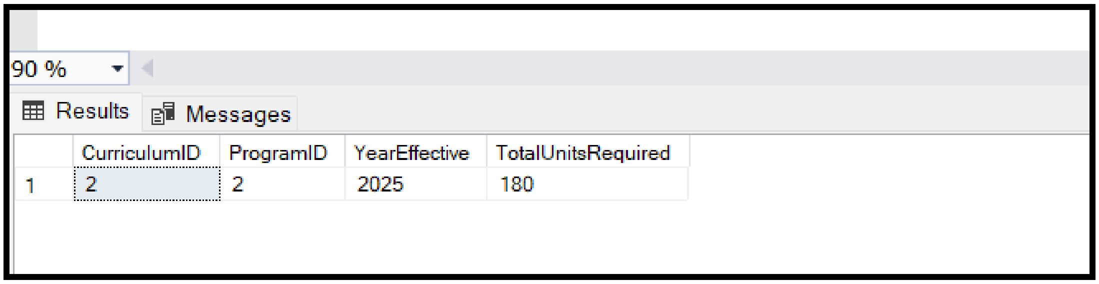
	ProgramID	ProgramName	ProgramCode
1	2	Bachelor of Science in Information Systems	BSIS

Sample Curriculum



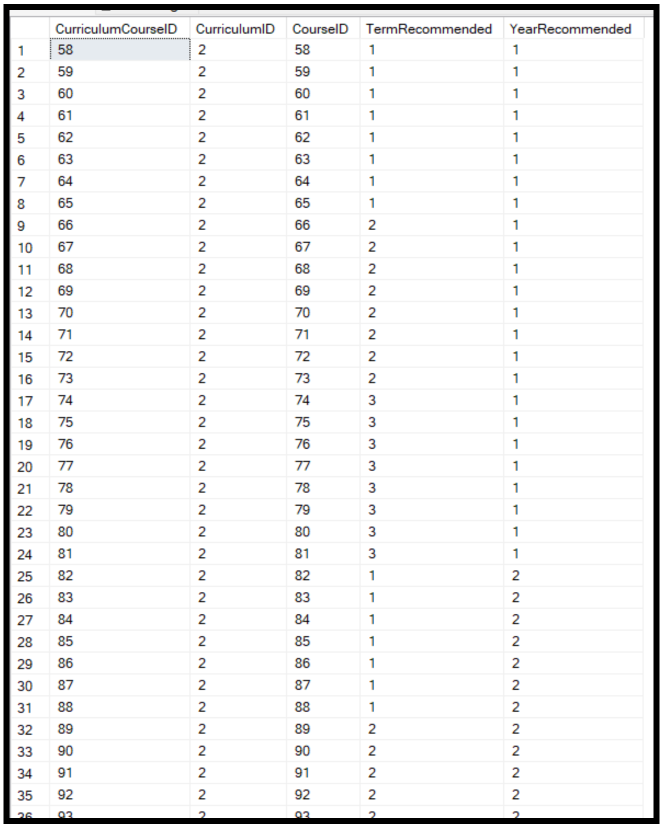
SQLQuery119.sql - C...nt_system (sa (89))
SQLQuery118.sql - C...nt_system (sa (90))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [CurriculumID]
      ,[ProgramID]
      ,[YearEffective]
      ,[TotalUnitsRequired]
  FROM [enlistment_system].[dbo].[Curriculum]
```

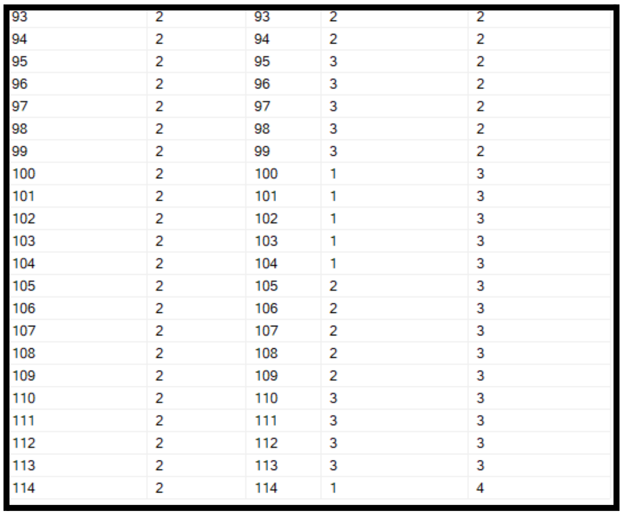


	CurriculumID	ProgramID	YearEffective	TotalUnitsRequired
1	2	2	2025	180

CurriculumCourses Records (To make sure each course is connected to one curriculum)

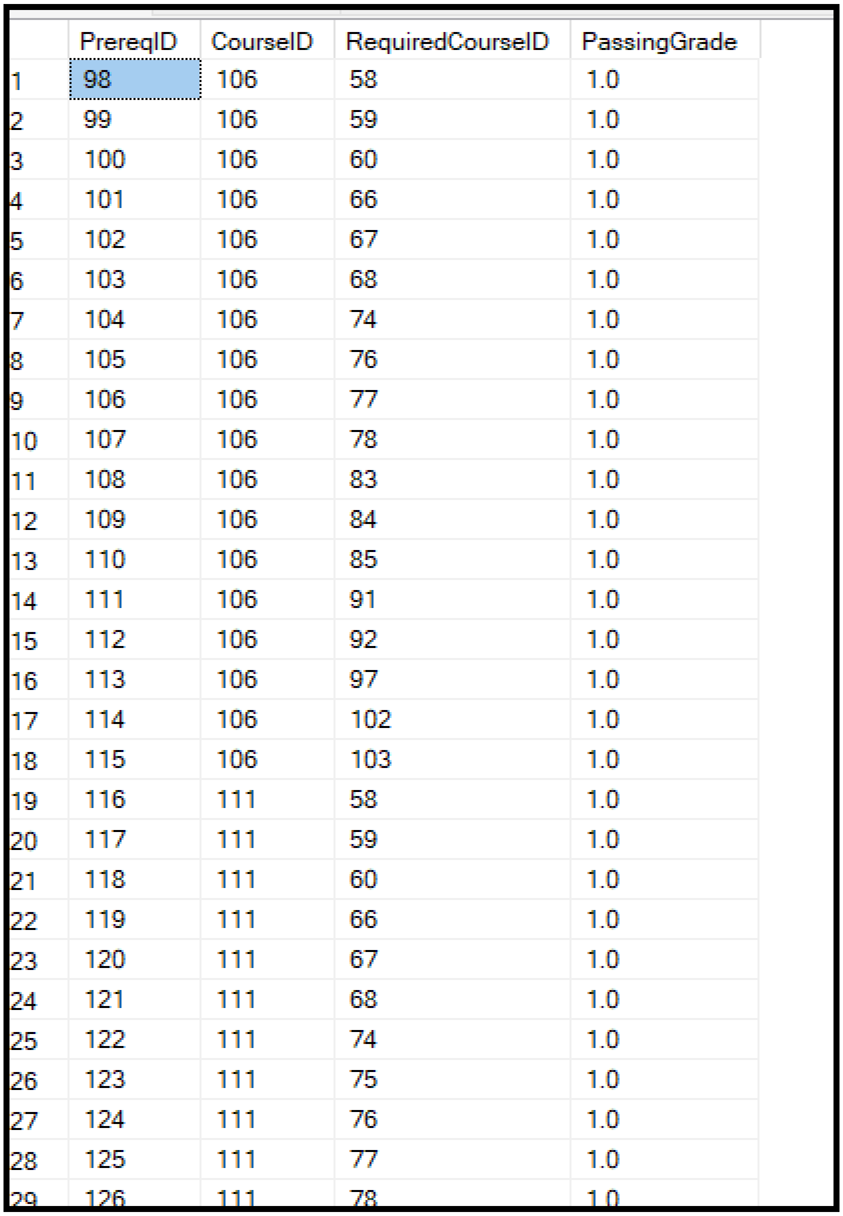


	CurriculumCourseID	CurriculumID	CourseID	TermRecommended	YearRecommended
1	58	2	58	1	1
2	59	2	59	1	1
3	60	2	60	1	1
4	61	2	61	1	1
5	62	2	62	1	1
6	63	2	63	1	1
7	64	2	64	1	1
8	65	2	65	1	1
9	66	2	66	2	1
10	67	2	67	2	1
11	68	2	68	2	1
12	69	2	69	2	1
13	70	2	70	2	1
14	71	2	71	2	1
15	72	2	72	2	1
16	73	2	73	2	1
17	74	2	74	3	1
18	75	2	75	3	1
19	76	2	76	3	1
20	77	2	77	3	1
21	78	2	78	3	1
22	79	2	79	3	1
23	80	2	80	3	1
24	81	2	81	3	1
25	82	2	82	1	2
26	83	2	83	1	2
27	84	2	84	1	2
28	85	2	85	1	2
29	86	2	86	1	2
30	87	2	87	1	2
31	88	2	88	1	2
32	89	2	89	2	2
33	90	2	90	2	2
34	91	2	91	2	2
35	92	2	92	2	2
36	93	2	93	2	2

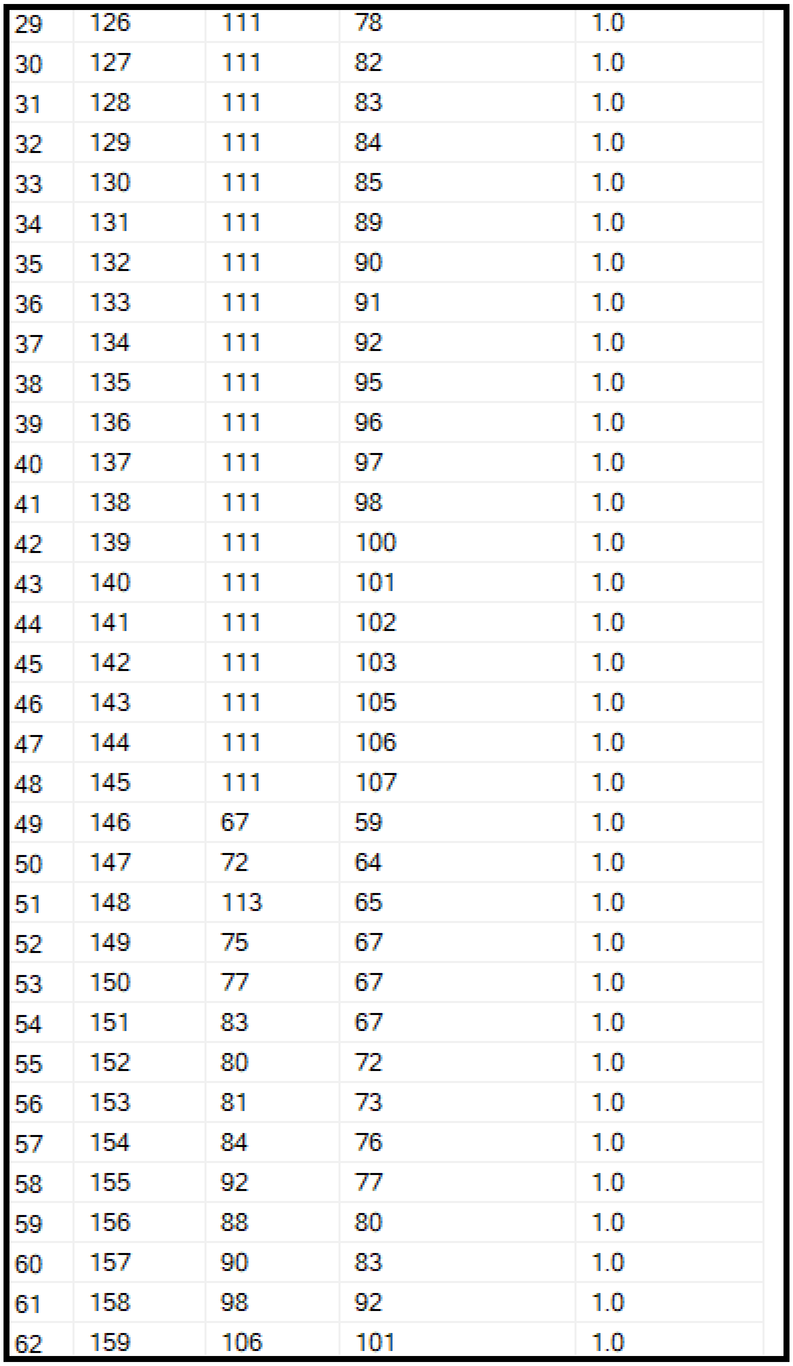


93	2	93	2	2
94	2	94	2	2
95	2	95	3	2
96	2	96	3	2
97	2	97	3	2
98	2	98	3	2
99	2	99	3	2
100	2	100	1	3
101	2	101	1	3
102	2	102	1	3
103	2	103	1	3
104	2	104	1	3
105	2	105	2	3
106	2	106	2	3
107	2	107	2	3
108	2	108	2	3
109	2	109	2	3
110	2	110	3	3
111	2	111	3	3
112	2	112	3	3
113	2	113	3	3
114	2	114	1	4

Prerequisites Table contains a CourseID and a RequiredCourseID that needs to be passed

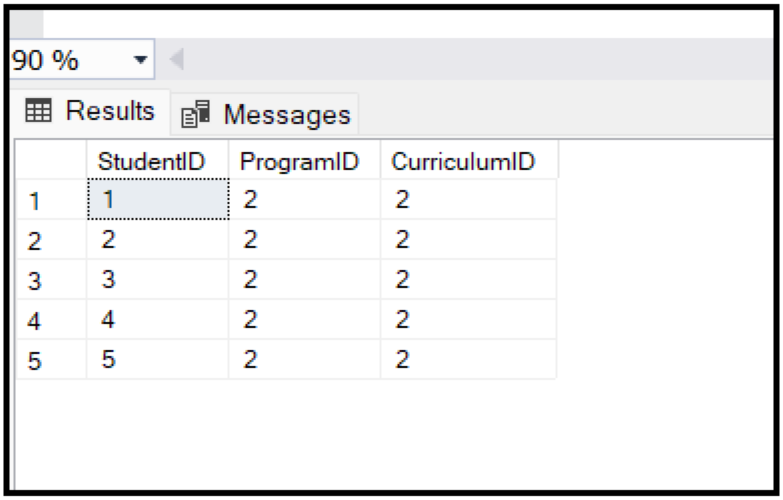


	PrereqID	CourseID	RequiredCourseID	PassingGrade
1	98	106	58	1.0
2	99	106	59	1.0
3	100	106	60	1.0
4	101	106	66	1.0
5	102	106	67	1.0
6	103	106	68	1.0
7	104	106	74	1.0
8	105	106	76	1.0
9	106	106	77	1.0
10	107	106	78	1.0
11	108	106	83	1.0
12	109	106	84	1.0
13	110	106	85	1.0
14	111	106	91	1.0
15	112	106	92	1.0
16	113	106	97	1.0
17	114	106	102	1.0
18	115	106	103	1.0
19	116	111	58	1.0
20	117	111	59	1.0
21	118	111	60	1.0
22	119	111	66	1.0
23	120	111	67	1.0
24	121	111	68	1.0
25	122	111	74	1.0
26	123	111	75	1.0
27	124	111	76	1.0
28	125	111	77	1.0
29	126	111	78	1.0



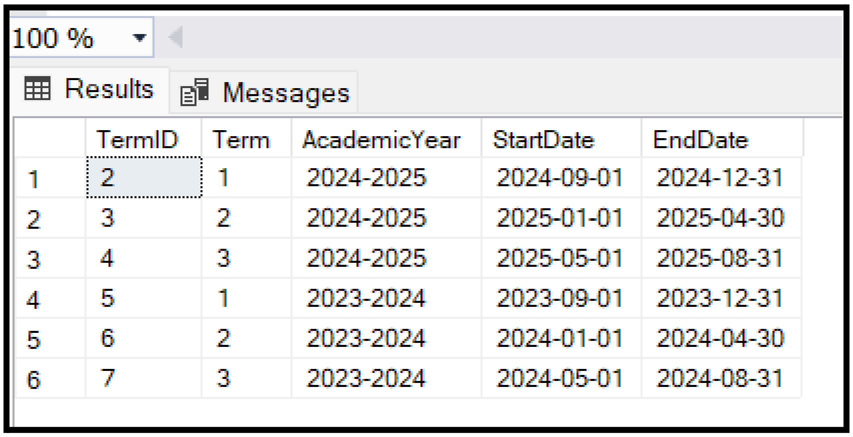
29	126	111	78	1.0
30	127	111	82	1.0
31	128	111	83	1.0
32	129	111	84	1.0
33	130	111	85	1.0
34	131	111	89	1.0
35	132	111	90	1.0
36	133	111	91	1.0
37	134	111	92	1.0
38	135	111	95	1.0
39	136	111	96	1.0
40	137	111	97	1.0
41	138	111	98	1.0
42	139	111	100	1.0
43	140	111	101	1.0
44	141	111	102	1.0
45	142	111	103	1.0
46	143	111	105	1.0
47	144	111	106	1.0
48	145	111	107	1.0
49	146	67	59	1.0
50	147	72	64	1.0
51	148	113	65	1.0
52	149	75	67	1.0
53	150	77	67	1.0
54	151	83	67	1.0
55	152	80	72	1.0
56	153	81	73	1.0
57	154	84	76	1.0
58	155	92	77	1.0
59	156	88	80	1.0
60	157	90	83	1.0
61	158	98	92	1.0
62	159	106	101	1.0

Students



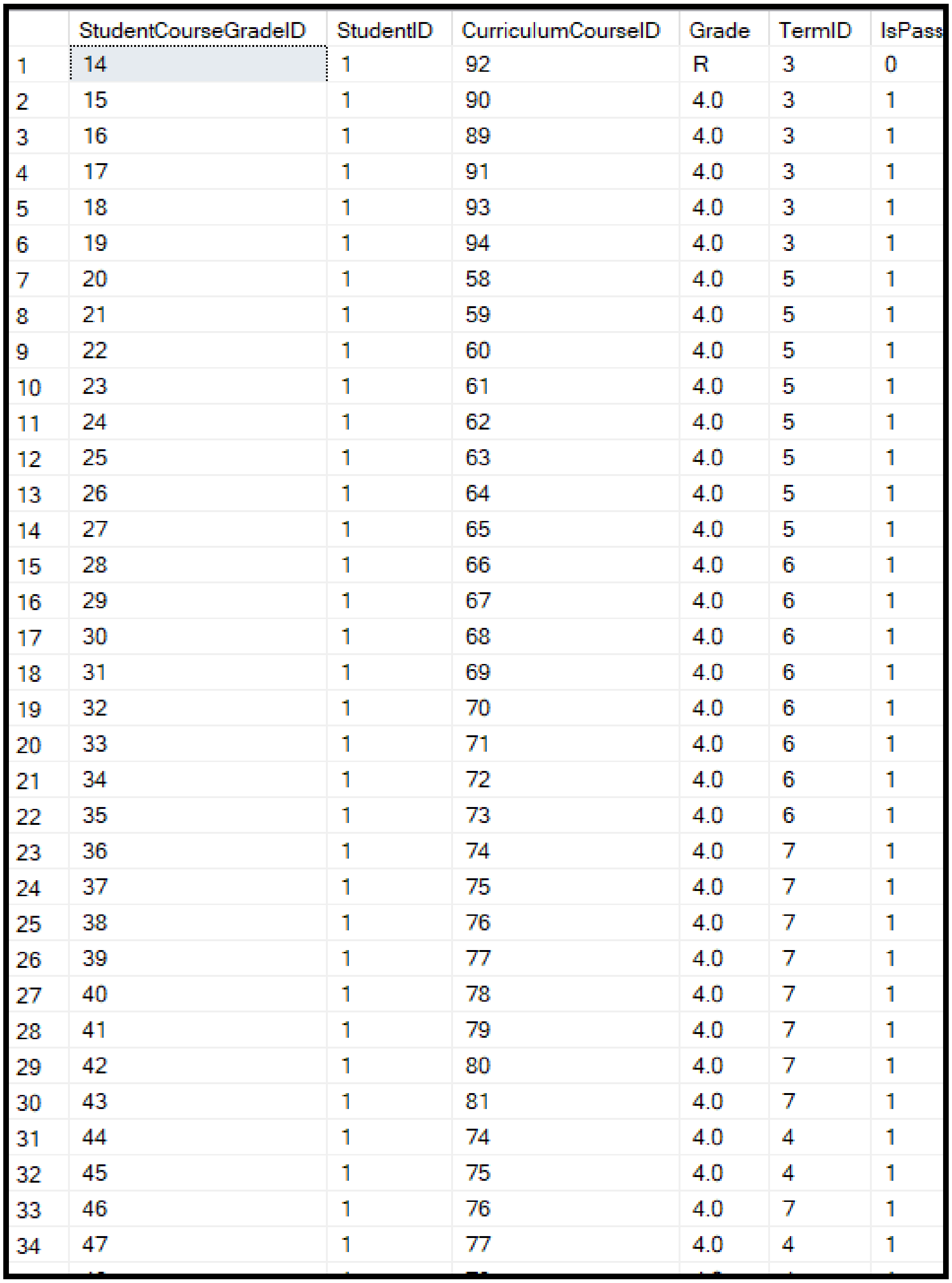
	StudentID	ProgramID	CurriculumID
1	1	2	2
2	2	2	2
3	3	2	2
4	4	2	2
5	5	2	2

Term

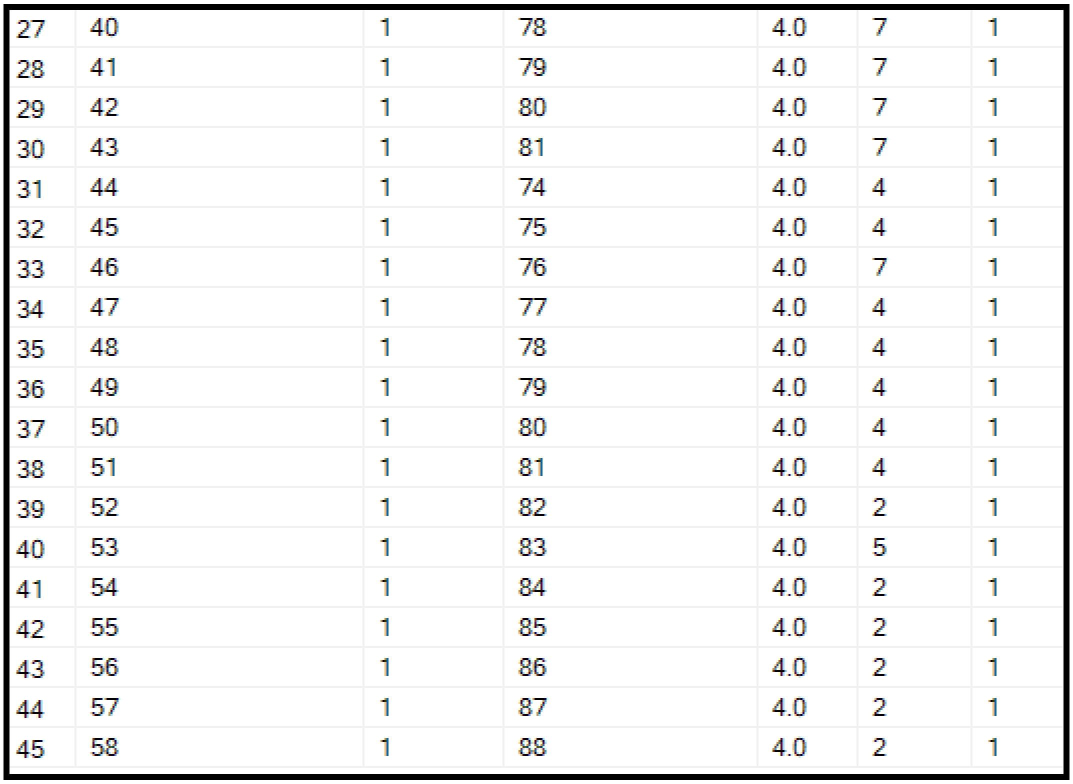


	TermID	Term	AcademicYear	StartDate	EndDate
1	2	1	2024-2025	2024-09-01	2024-12-31
2	3	2	2024-2025	2025-01-01	2025-04-30
3	4	3	2024-2025	2025-05-01	2025-08-31
4	5	1	2023-2024	2023-09-01	2023-12-31
5	6	2	2023-2024	2024-01-01	2024-04-30
6	7	3	2023-2024	2024-05-01	2024-08-31

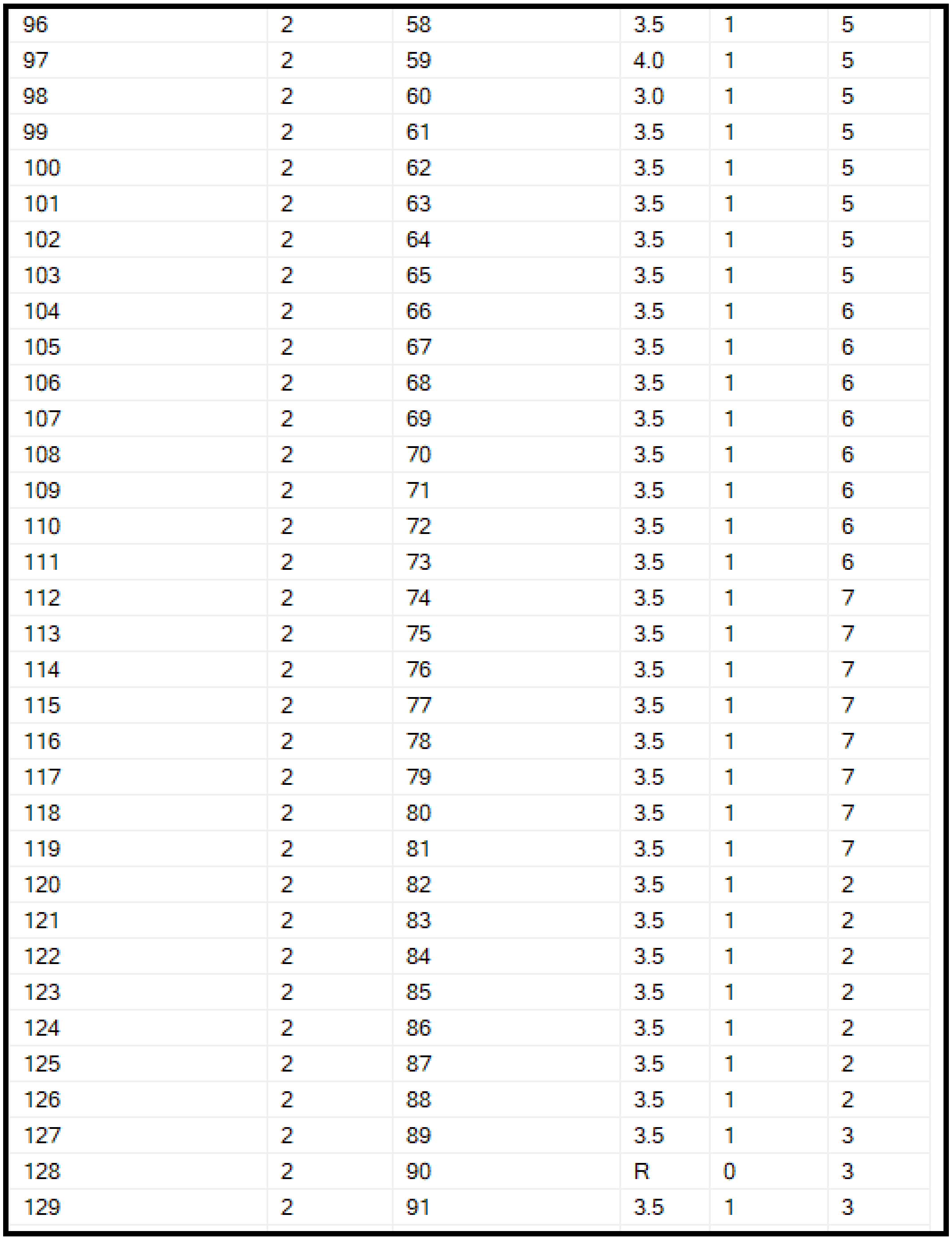
Sample Records for StudentsCourseGrades (Grades from 2023-2045 Term 1 - 3 and 2024-2025 Term 1 - 2)



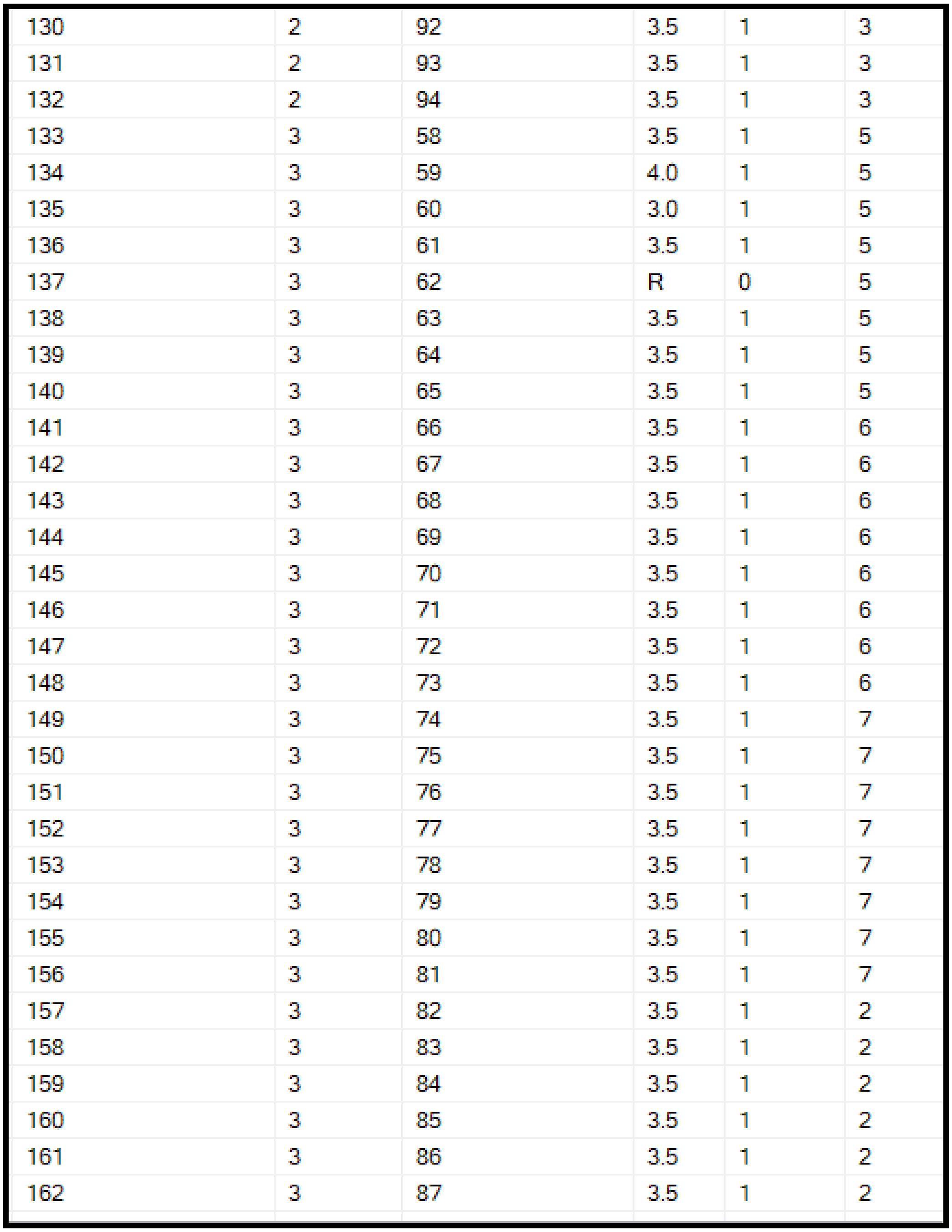
	StudentCourseGradeID	StudentID	CurriculumCourseID	Grade	TermID	IsPass
1	14	1	92	R	3	0
2	15	1	90	4.0	3	1
3	16	1	89	4.0	3	1
4	17	1	91	4.0	3	1
5	18	1	93	4.0	3	1
6	19	1	94	4.0	3	1
7	20	1	58	4.0	5	1
8	21	1	59	4.0	5	1
9	22	1	60	4.0	5	1
10	23	1	61	4.0	5	1
11	24	1	62	4.0	5	1
12	25	1	63	4.0	5	1
13	26	1	64	4.0	5	1
14	27	1	65	4.0	5	1
15	28	1	66	4.0	6	1
16	29	1	67	4.0	6	1
17	30	1	68	4.0	6	1
18	31	1	69	4.0	6	1
19	32	1	70	4.0	6	1
20	33	1	71	4.0	6	1
21	34	1	72	4.0	6	1
22	35	1	73	4.0	6	1
23	36	1	74	4.0	7	1
24	37	1	75	4.0	7	1
25	38	1	76	4.0	7	1
26	39	1	77	4.0	7	1
27	40	1	78	4.0	7	1
28	41	1	79	4.0	7	1
29	42	1	80	4.0	7	1
30	43	1	81	4.0	7	1
31	44	1	74	4.0	4	1
32	45	1	75	4.0	4	1
33	46	1	76	4.0	7	1
34	47	1	77	4.0	4	1



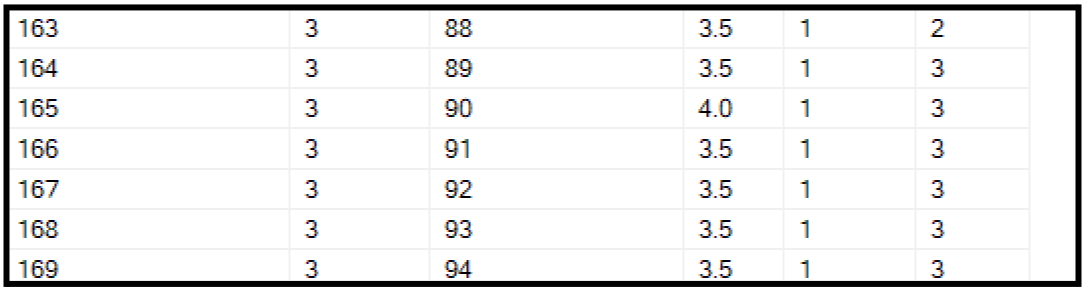
27	40	1	78	4.0	7	1
28	41	1	79	4.0	7	1
29	42	1	80	4.0	7	1
30	43	1	81	4.0	7	1
31	44	1	74	4.0	4	1
32	45	1	75	4.0	4	1
33	46	1	76	4.0	7	1
34	47	1	77	4.0	4	1
35	48	1	78	4.0	4	1
36	49	1	79	4.0	4	1
37	50	1	80	4.0	4	1
38	51	1	81	4.0	4	1
39	52	1	82	4.0	2	1
40	53	1	83	4.0	5	1
41	54	1	84	4.0	2	1
42	55	1	85	4.0	2	1
43	56	1	86	4.0	2	1
44	57	1	87	4.0	2	1
45	58	1	88	4.0	2	1



96	2	58	3.5	1	5
97	2	59	4.0	1	5
98	2	60	3.0	1	5
99	2	61	3.5	1	5
100	2	62	3.5	1	5
101	2	63	3.5	1	5
102	2	64	3.5	1	5
103	2	65	3.5	1	5
104	2	66	3.5	1	6
105	2	67	3.5	1	6
106	2	68	3.5	1	6
107	2	69	3.5	1	6
108	2	70	3.5	1	6
109	2	71	3.5	1	6
110	2	72	3.5	1	6
111	2	73	3.5	1	6
112	2	74	3.5	1	7
113	2	75	3.5	1	7
114	2	76	3.5	1	7
115	2	77	3.5	1	7
116	2	78	3.5	1	7
117	2	79	3.5	1	7
118	2	80	3.5	1	7
119	2	81	3.5	1	7
120	2	82	3.5	1	2
121	2	83	3.5	1	2
122	2	84	3.5	1	2
123	2	85	3.5	1	2
124	2	86	3.5	1	2
125	2	87	3.5	1	2
126	2	88	3.5	1	2
127	2	89	3.5	1	3
128	2	90	R	0	3
129	2	91	3.5	1	3



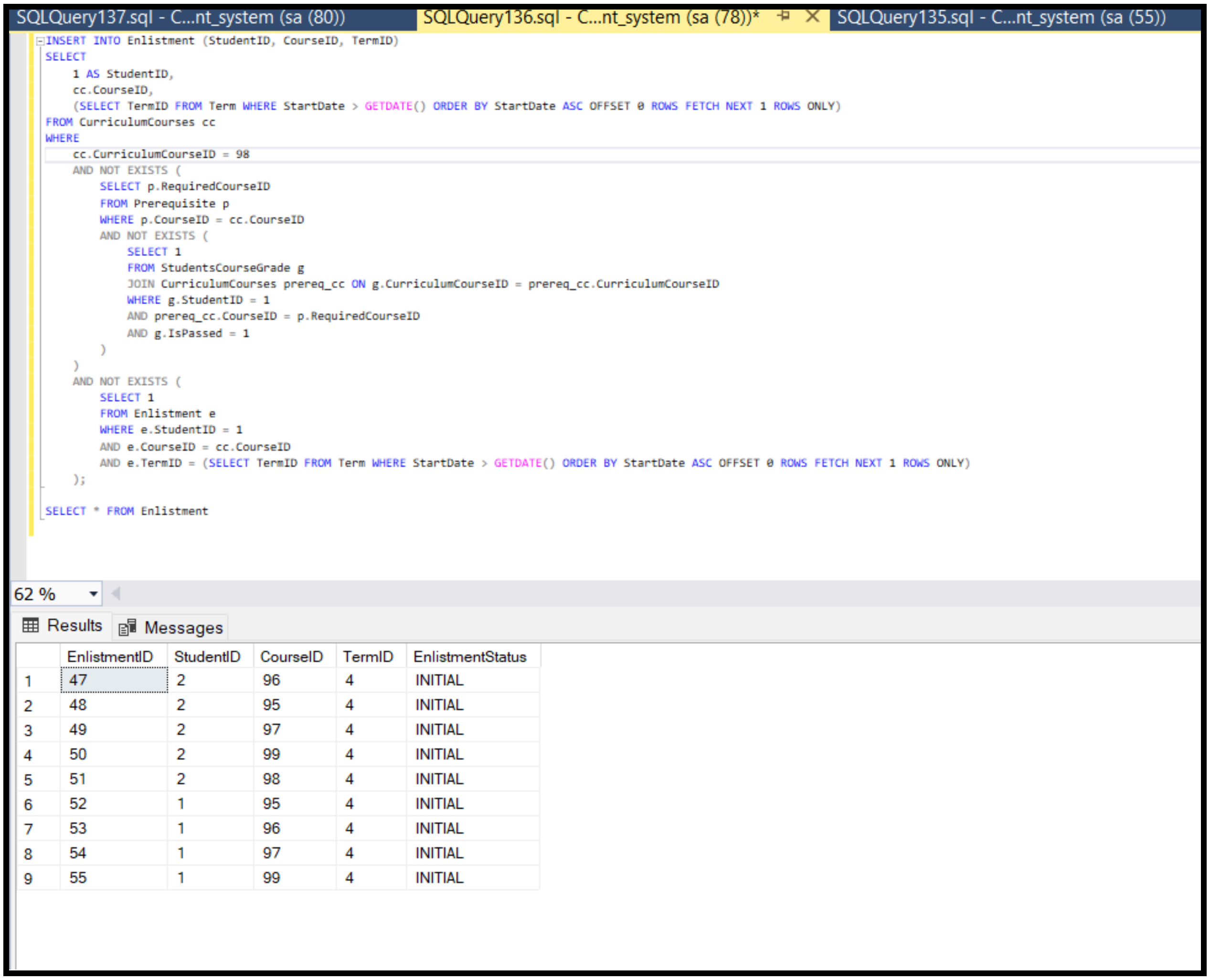
130	2	92	3.5	1	3
131	2	93	3.5	1	3
132	2	94	3.5	1	3
133	3	58	3.5	1	5
134	3	59	4.0	1	5
135	3	60	3.0	1	5
136	3	61	3.5	1	5
137	3	62	R	0	5
138	3	63	3.5	1	5
139	3	64	3.5	1	5
140	3	65	3.5	1	5
141	3	66	3.5	1	6
142	3	67	3.5	1	6
143	3	68	3.5	1	6
144	3	69	3.5	1	6
145	3	70	3.5	1	6
146	3	71	3.5	1	6
147	3	72	3.5	1	6
148	3	73	3.5	1	6
149	3	74	3.5	1	7
150	3	75	3.5	1	7
151	3	76	3.5	1	7
152	3	77	3.5	1	7
153	3	78	3.5	1	7
154	3	79	3.5	1	7
155	3	80	3.5	1	7
156	3	81	3.5	1	7
157	3	82	3.5	1	2
158	3	83	3.5	1	2
159	3	84	3.5	1	2
160	3	85	3.5	1	2
161	3	86	3.5	1	2
162	3	87	3.5	1	2



163	3	88	3.5	1	2
164	3	89	3.5	1	3
165	3	90	4.0	1	3
166	3	91	3.5	1	3
167	3	92	3.5	1	3
168	3	93	3.5	1	3
169	3	94	3.5	1	3

F. QUERIES & RESULT SET

INSERT Query that enlists a student to INFODBM



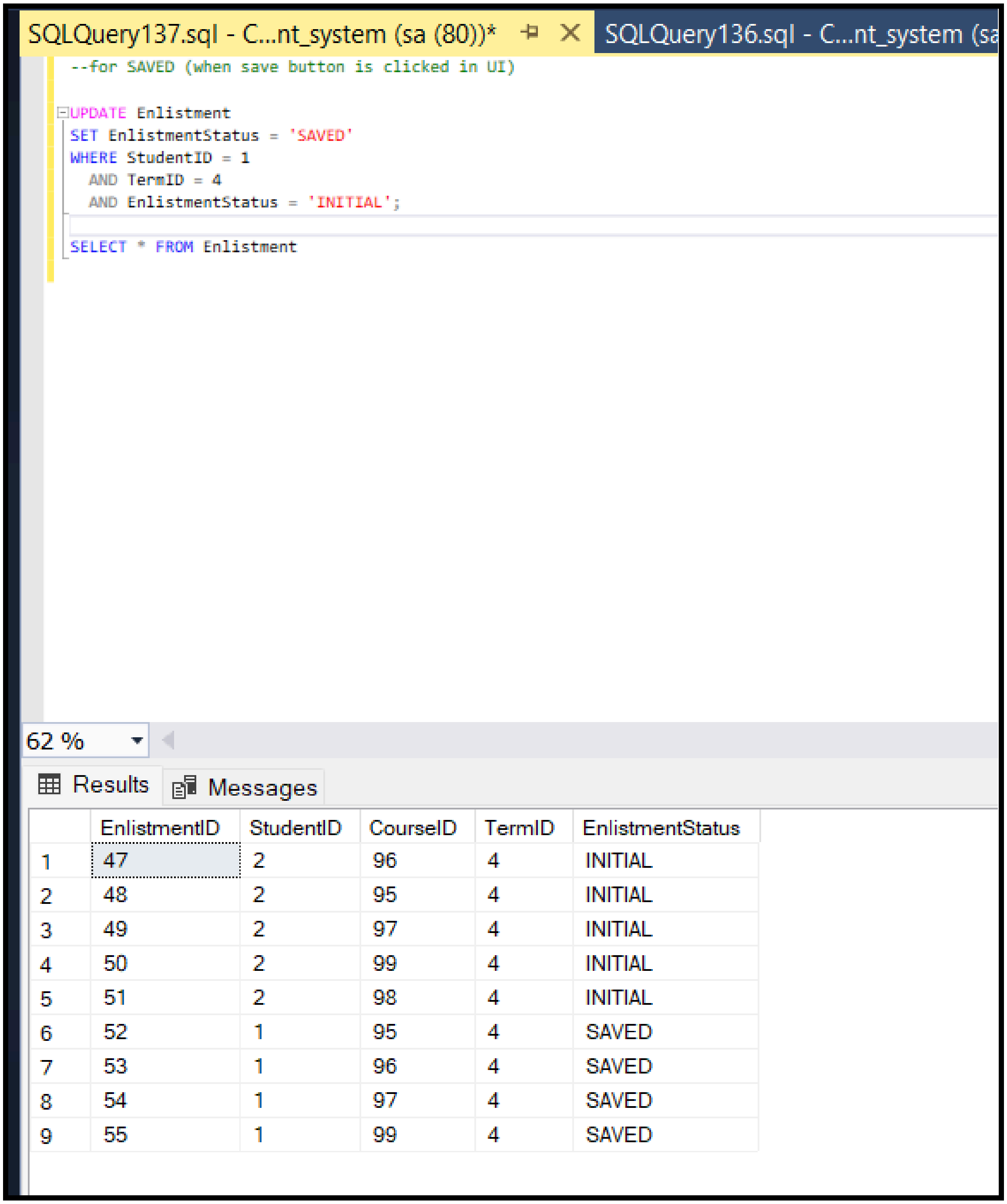
SQLQuery137.sql - C...nt_system (sa (80))
SQLQuery136.sql - C...nt_system (sa (78))*
SQLQuery135.sql - C...nt_system (sa (55))
62 %
Results
Messages

```sql
INSERT INTO Enlistment (StudentID, CourseID, TermID)
SELECT
    1 AS StudentID,
    cc.CourseID,
    (SELECT TermID FROM Term WHERE StartDate > GETDATE() ORDER BY StartDate ASC OFFSET 0 ROWS FETCH NEXT 1 ROWS ONLY)
FROM CurriculumCourses cc
WHERE
    cc.CurriculumCourseID = 98
    AND NOT EXISTS (
        SELECT p.RequiredCourseID
        FROM Prerequisite p
        WHERE p.CourseID = cc.CourseID
        AND NOT EXISTS (
            SELECT 1
            FROM StudentsCourseGrade g
            JOIN CurriculumCourses prereq_cc ON g.CurriculumCourseID = prereq_cc.CurriculumCourseID
            WHERE g.StudentID = 1
            AND prereq_cc.CourseID = p.RequiredCourseID
            AND g.IsPassed = 1
        )
    )
    AND NOT EXISTS (
        SELECT 1
        FROM Enlistment e
        WHERE e.StudentID = 1
        AND e.CourseID = cc.CourseID
        AND e.TermID = (SELECT TermID FROM Term WHERE StartDate > GETDATE() ORDER BY StartDate ASC OFFSET 0 ROWS FETCH NEXT 1 ROWS ONLY)
    );

SELECT * FROM Enlistment
```

	EnlistmentID	StudentID	CourseID	TermID	EnlistmentStatus
1	47	2	96	4	INITIAL
2	48	2	95	4	INITIAL
3	49	2	97	4	INITIAL
4	50	2	99	4	INITIAL
5	51	2	98	4	INITIAL
6	52	1	95	4	INITIAL
7	53	1	96	4	INITIAL
8	54	1	97	4	INITIAL
9	55	1	99	4	INITIAL

Now using the same **INSERT** Query for the other CourseID's but if you notice there is no 98 for StudentID 1

UPDATE Command for Enlistment, this saves the student's enlistment

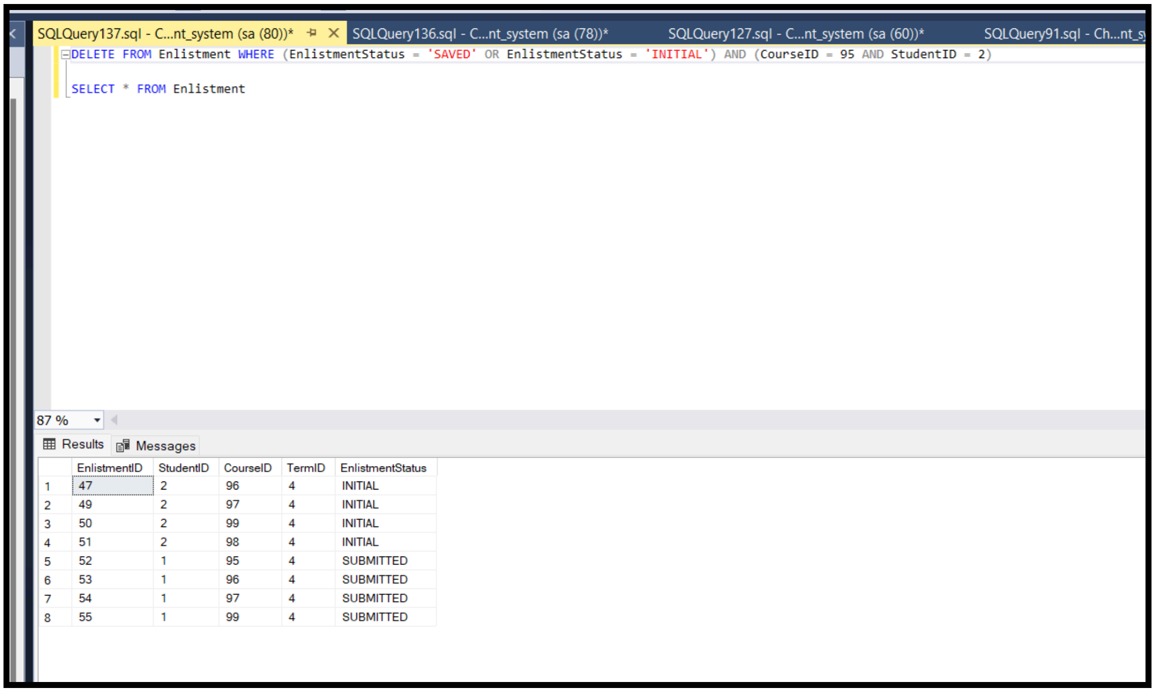


SQLQuery137.sql - C...nt_system (sa (80))*
SQLQuery136.sql - C...nt_system (sa
62 %
Results
Messages

```sql
--for SAVED (when save button is clicked in UI)

UPDATE Enlistment
SET EnlistmentStatus = 'SAVED'
WHERE StudentID = 1
  AND TermID = 4
  AND EnlistmentStatus = 'INITIAL';

SELECT * FROM Enlistment
```

	EnlistmentID	StudentID	CourseID	TermID	EnlistmentStatus
1	47	2	96	4	INITIAL
2	48	2	95	4	INITIAL
3	49	2	97	4	INITIAL
4	50	2	99	4	INITIAL
5	51	2	98	4	INITIAL
6	52	1	95	4	SAVED
7	53	1	96	4	SAVED
8	54	1	97	4	SAVED
9	55	1	99	4	SAVED

DELETE Command for Enlistment, this deletes the students' currently enlisted course if its status is SAVED or INITIAL. If it is submitted, it can't be deleted anymore.

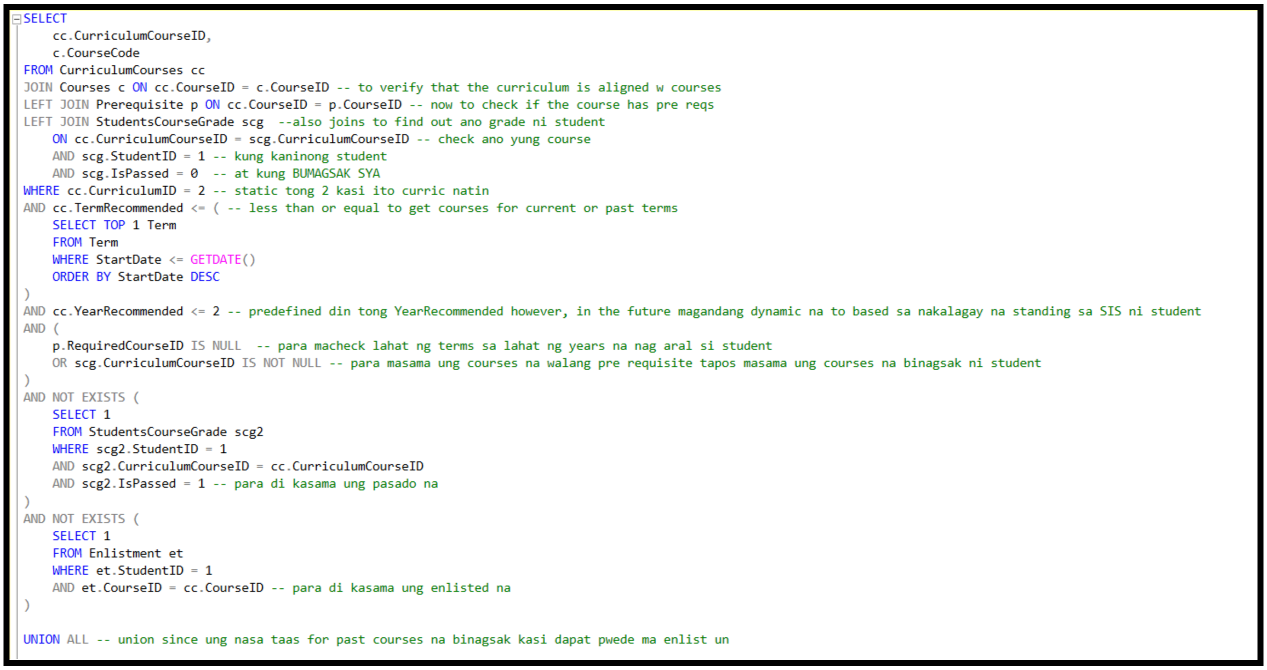


SQLQuery137.sql - C...nt_system (sa (80))*
SQLQuery136.sql - C...nt_system (sa (78))*
SQLQuery127.sql - C...nt_system (sa (60))*
SQLQuery91.sql - Ch...nt_s
87 %
Results
Messages

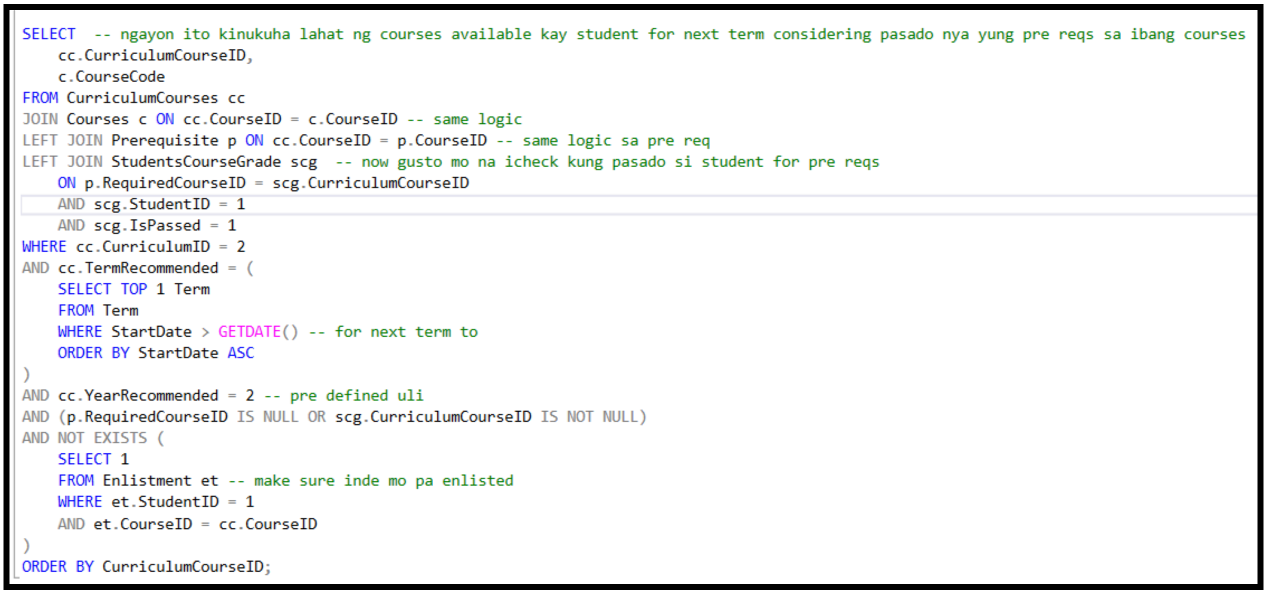
```
DELETE FROM Enlistment WHERE (EnlistmentStatus = 'SAVED' OR EnlistmentStatus = 'INITIAL') AND (CourseID = 95 AND StudentID = 2)

SELECT * FROM Enlistment
```

	EnlistmentID	StudentID	CourseID	TermID	EnlistmentStatus
1	47	2	96	4	INITIAL
2	49	2	97	4	INITIAL
3	50	2	99	4	INITIAL
4	51	2	98	4	INITIAL
5	52	1	95	4	SUBMITTED
6	53	1	96	4	SUBMITTED
7	54	1	97	4	SUBMITTED
8	55	1	99	4	SUBMITTED

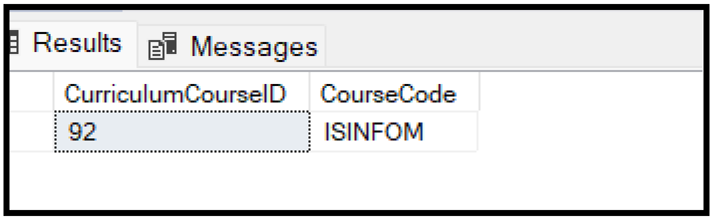
SELECT Query to View available courses that the student could enlist. The student would be able to view courses he/she failed in the previous terms and the courses he/she could enlist for the next term. This strictly follows courses for a specific year in a specific term.



```sql
SELECT
    cc.CurriculumCourseID,
    c.CourseCode
FROM CurriculumCourses cc
JOIN Courses c ON cc.CourseID = c.CourseID -- to verify that the curriculum is aligned w courses
LEFT JOIN Prerequisite p ON cc.CourseID = p.CourseID -- now to check if the course has pre reqs
LEFT JOIN StudentsCourseGrade scg  --also joins to find out ano grade ni student
    ON cc.CurriculumCourseID = scg.CurriculumCourseID -- check ano yung course
    AND scg.StudentID = 1 -- kung kaninong student
    AND scg.IsPassed = 0  -- at kung BUMAGSAK SYA
WHERE cc.CurriculumID = 2 -- static tong 2 kasi ito curric natin
AND cc.TermRecommended <= ( -- less than or equal to get courses for current or past terms
    SELECT TOP 1 Term
    FROM Term
    WHERE StartDate <= GETDATE()
    ORDER BY StartDate DESC
)
AND cc.YearRecommended <= 2 -- predefined din tong YearRecommended however, in the future magandang dynamic na to based sa nakalagay na standing sa SIS ni student
AND (
    p.RequiredCourseID IS NULL  -- para macheck lahat ng terms sa lahat ng years na nag aral si student
    OR scg.CurriculumCourseID IS NOT NULL -- para masama ung courses na walang pre requisite tapos masama ung courses na binagsak ni student
)
AND NOT EXISTS (
    SELECT 1
    FROM StudentsCourseGrade scg2
    WHERE scg2.StudentID = 1
    AND scg2.CurriculumCourseID = cc.CurriculumCourseID
    AND scg2.IsPassed = 1 -- para di kasama ung pasado na
)
AND NOT EXISTS (
    SELECT 1
    FROM Enlistment et
    WHERE et.StudentID = 1
    AND et.CourseID = cc.CourseID -- para di kasama ung enlisted na
)

UNION ALL -- union since ung nasa taas for past courses na binagsak kasi dapat pwede ma enlist un
```

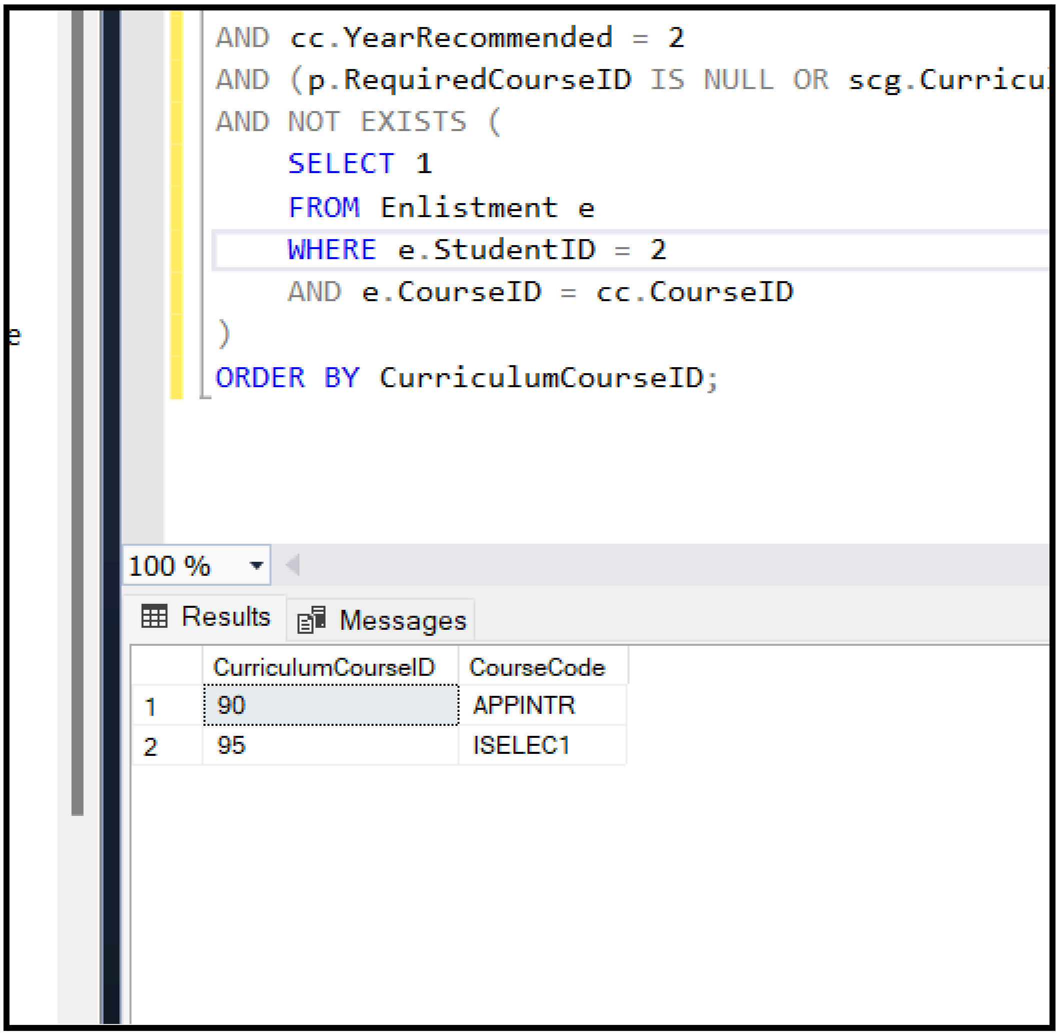


```sql
SELECT  -- ngayon ito kinukuha lahat ng courses available kay student for next term considering pasado nya yung pre reqs sa ibang courses
    cc.CurriculumCourseID,
    c.CourseCode
FROM CurriculumCourses cc
JOIN Courses c ON cc.CourseID = c.CourseID -- same logic
LEFT JOIN Prerequisite p ON cc.CourseID = p.CourseID -- same logic sa pre req
LEFT JOIN StudentsCourseGrade scg  -- now gusto mo na icheck kung pasado si student for pre reqs
    ON p.RequiredCourseID = scg.CurriculumCourseID
    AND scg.StudentID = 1
    AND scg.IsPassed = 1
WHERE cc.CurriculumID = 2
AND cc.TermRecommended = (
    SELECT TOP 1 Term
    FROM Term
    WHERE StartDate > GETDATE() -- for next term to
    ORDER BY StartDate ASC
)
AND cc.YearRecommended = 2 -- pre defined uli
AND (p.RequiredCourseID IS NULL OR scg.CurriculumCourseID IS NOT NULL)
AND NOT EXISTS (
    SELECT 1
    FROM Enlistment et -- make sure inde mo pa enlisted
    WHERE et.StudentID = 1
    AND et.CourseID = cc.CourseID
)
ORDER BY CurriculumCourseID;
```



	CurriculumCourseID	CourseCode
	92	ISINFOM

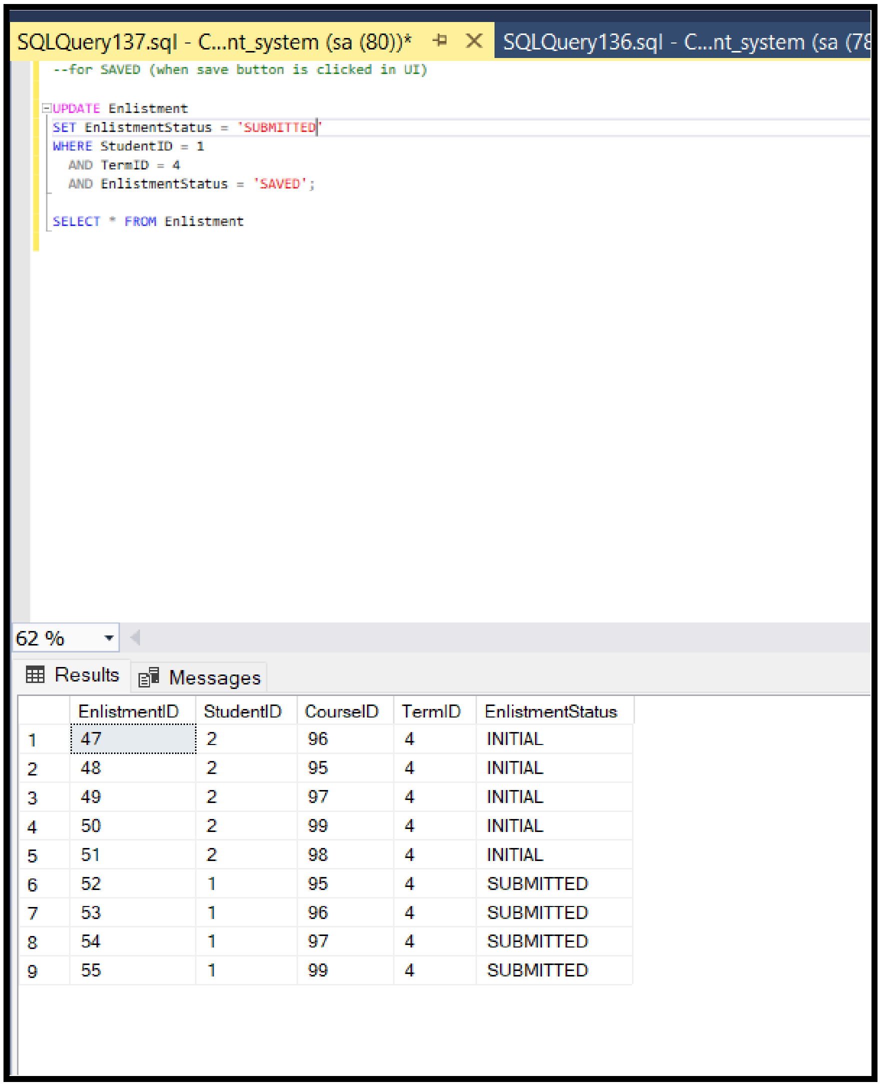
Though to show functionality I used the same query but changed it to ‘2’



```
AND cc.YearRecommended = 2
AND (p.RequiredCourseID IS NULL OR scg.Curricu
AND NOT EXISTS (
    SELECT 1
    FROM Enlistment e
    WHERE e.StudentID = 2
    AND e.CourseID = cc.CourseID
)
ORDER BY CurriculumCourseID;
```

	CurriculumCourseID	CourseCode
1	90	APPINTR
2	95	ISELEC1

UPDATE Command for Enlistment, this submits the students’ enlistment

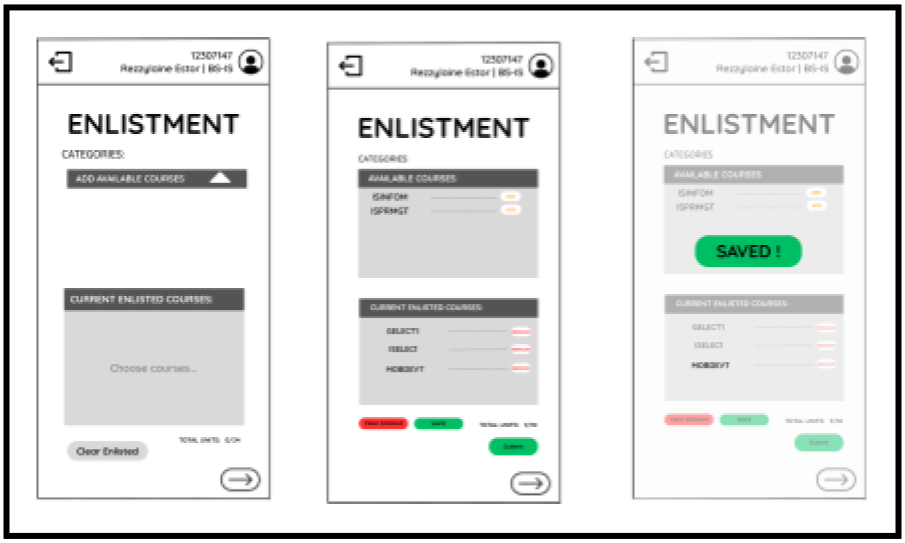

SQLQuery137.sql - C...nt_system (sa (80))*
SQLQuery136.sql - C...nt_system (sa (78
62 %
Results
Messages

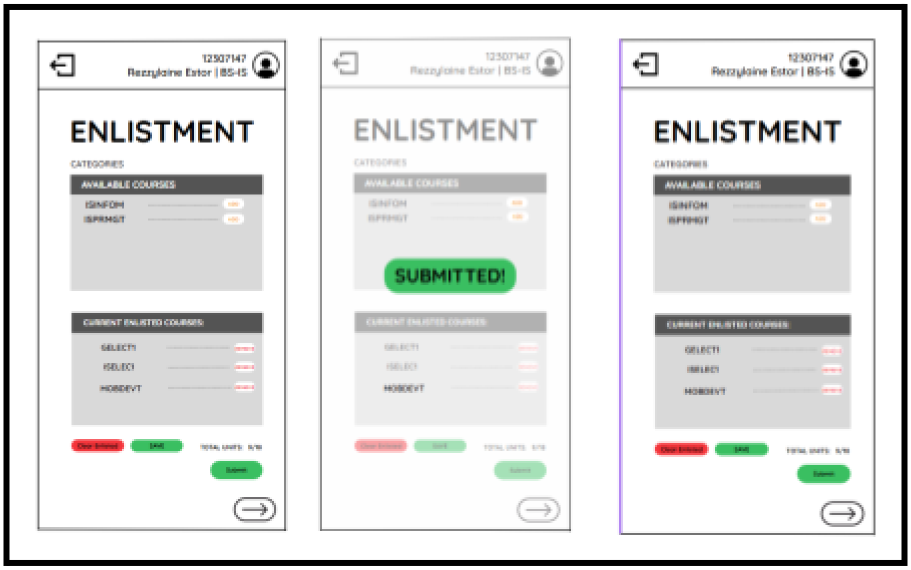
```sql
--for SAVED (when save button is clicked in UI)

UPDATE Enlistment
SET EnlistmentStatus = 'SUBMITTED'
WHERE StudentID = 1
  AND TermID = 4
  AND EnlistmentStatus = 'SAVED';

SELECT * FROM Enlistment
```

	EnlistmentID	StudentID	CourseID	TermID	EnlistmentStatus
1	47	2	96	4	INITIAL
2	48	2	95	4	INITIAL
3	49	2	97	4	INITIAL
4	50	2	99	4	INITIAL
5	51	2	98	4	INITIAL
6	52	1	95	4	SUBMITTED
7	53	1	96	4	SUBMITTED
8	54	1	97	4	SUBMITTED
9	55	1	99	4	SUBMITTED

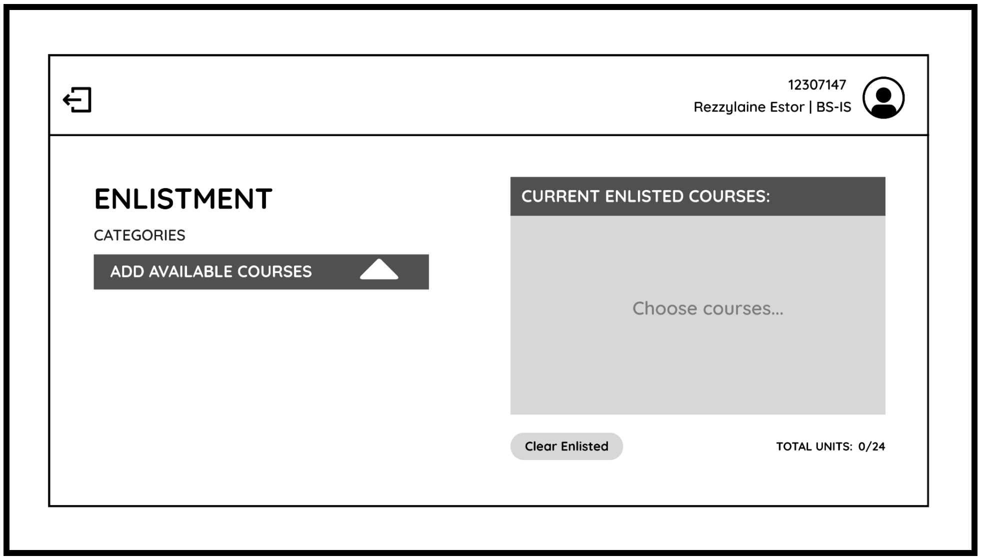
G. SAMPLE APPLICATION UI DESIGN

MOBILE UI:





WEB UI:



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES

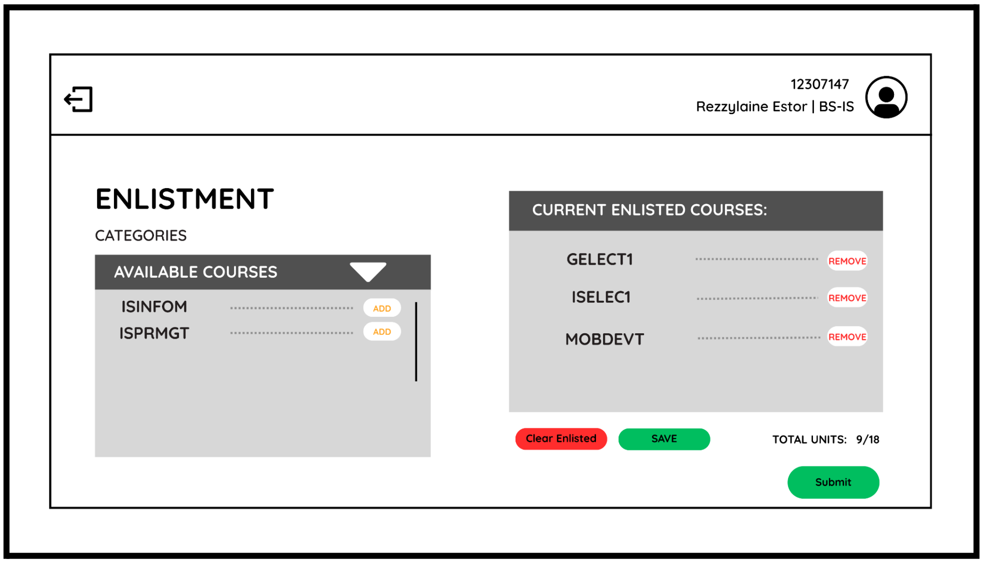
ADD AVAILABLE COURSES

CURRENT ENLISTED COURSES:

Choose courses...

Clear Enlisted

TOTAL UNITS: 0/24



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES

AVAILABLE COURSES

ISINFOM ADD
ISPRMGT ADD

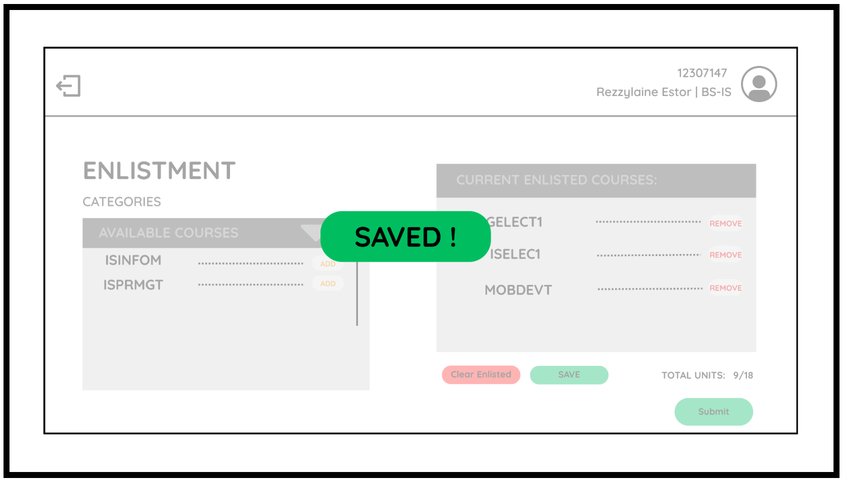
CURRENT ENLISTED COURSES:

GELECT1 REMOVE
ISELEC1 REMOVE
MOBDEVT REMOVE

Clear Enlisted SAVE

TOTAL UNITS: 9/18

Submit

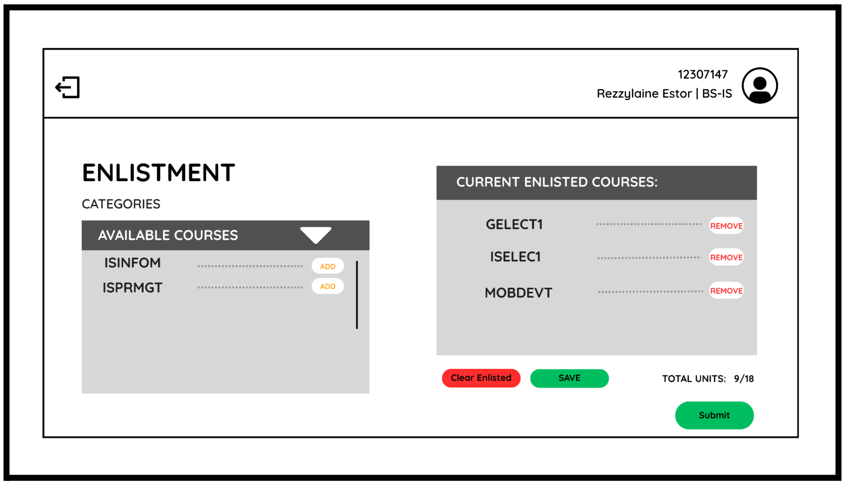


12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES

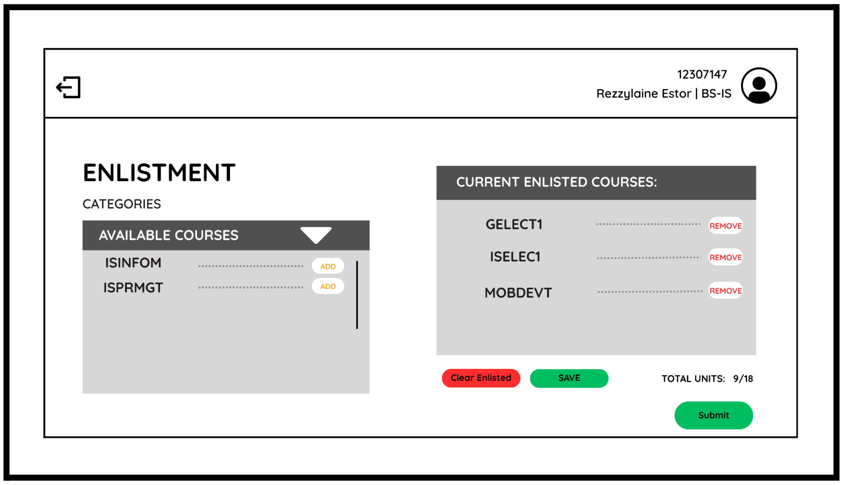
SAVED !



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

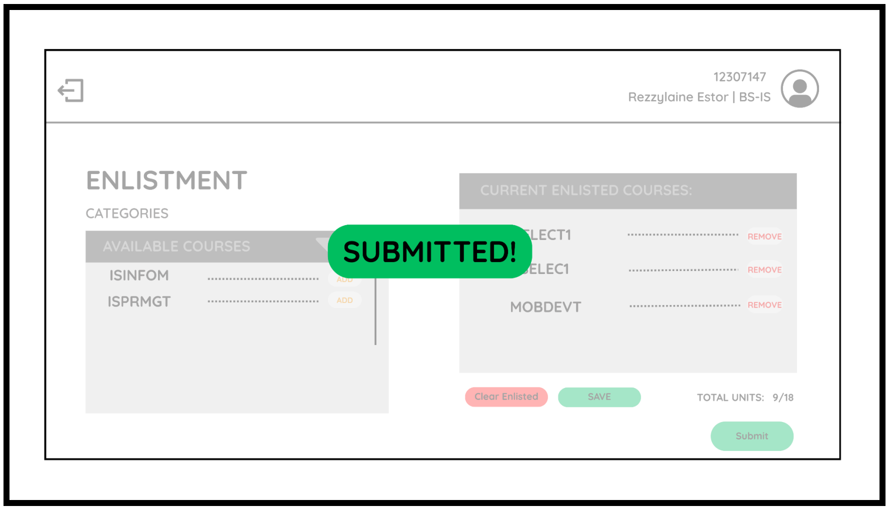
CATEGORIES



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES

AVAILABLE COURSES

ISINFOM .... ADD
ISPRMGT .... ADD

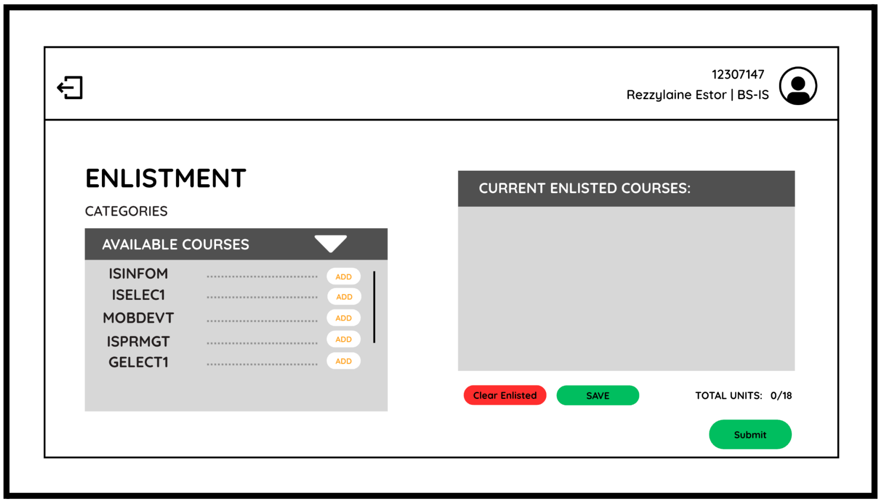
SUBMITTED!

CURRENT ENLISTED COURSES:

LECT1 .... REMOVE
ELEC1 .... REMOVE
MOBDEVT .... REMOVE

Clear Enlisted SAVE TOTAL UNITS: 9/18

Submit



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES

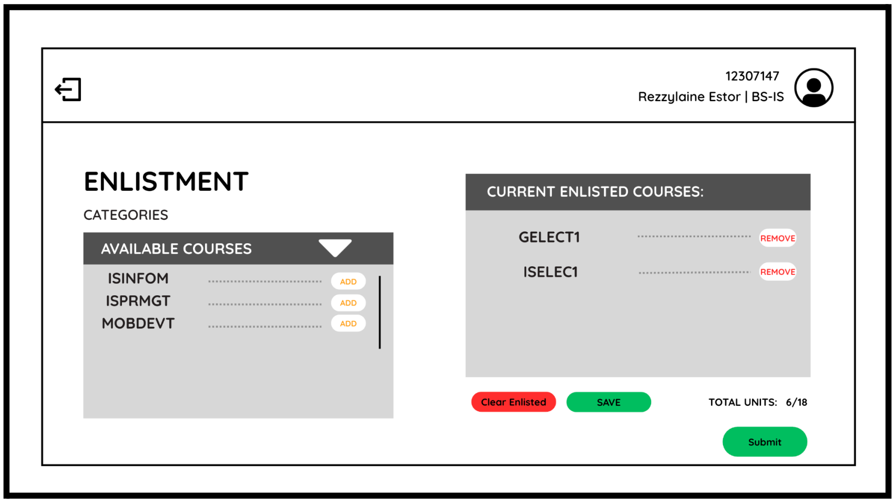
AVAILABLE COURSES

ISINFOM .... ADD
ISELEC1 .... ADD
MOBDEVT .... ADD
ISPRMGT .... ADD
GELECT1 .... ADD

CURRENT ENLISTED COURSES:

Clear Enlisted SAVE TOTAL UNITS: 0/18

Submit



12307147
Rezzylaine Estor | BS-IS

ENLISTMENT

CATEGORIES

AVAILABLE COURSES

ISINFOM .... ADD
ISPRMGT .... ADD
MOBDEVT .... ADD

CURRENT ENLISTED COURSES:

GELECT1 .... REMOVE
ISELEC1 .... REMOVE

Clear Enlisted SAVE TOTAL UNITS: 6/18

Submit