ADAMSON UNIVERSITY

900 San Marcelino Street, Ermita, Manila 1000

CPE419AL : DATABASE DESIGN AND DEVELOPMENT LAB

Database and Design Class Student Record Database

Group C

Group Members:

Herrera, Emmanuel Santillan

Maranan, Aron John M.

Pon-an, Jolyn D.

Reyes, Reinaliza P.

Engr. Baluyot, Raffaello Manalaysay

Instructor

Date of Submission: 10 December 2018

Title of Project: Database Design Class Student Record Database

I. Introduction

Recording data of a student in a class plays a key role in the management system in any school but the whole process of it adds heavy workload for the teachers. A database system of a class record is a highly desirable addition to the educational tool-kit particularly when it can provide less effort and a more effective and timely outcome.

The project titled “Database Design Class Record Database” is a class record management database system for monitoring and controlling informations in a class record. The project is developed in structured query language, which mainly focuses on basic operations like adding, reading, updating and removing students and their activities, attendance and merits. It is designed to help users maintain and organize a class record.

II. Objectives

2.1. To develop a Database Design Class Student Record applying Database Programming technique exemplified by MySQL application.

2.2. To create a structure and design to express the visualized concept with the help through software.

2.3. To create UML class diagram of the created database project to demonstrate its component and relationships.

2.4. To apply the lessons learned in this course, Database Design and Development.

III. Features

3.1. Seed Data

3.2. CRUD Functions:

- Student
- Attendance
- Activities
- Merits

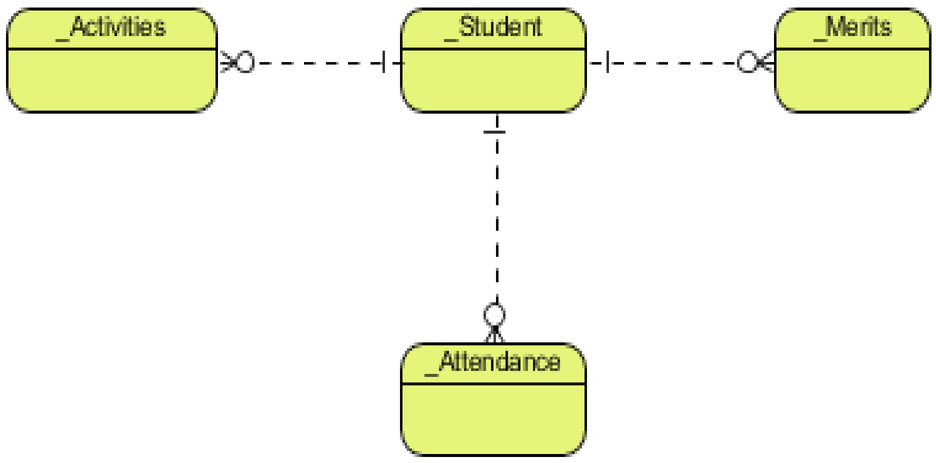
3.3. SQL Queries

- Tables
- Views
- Stored Procedures

IV. ERD Diagrams

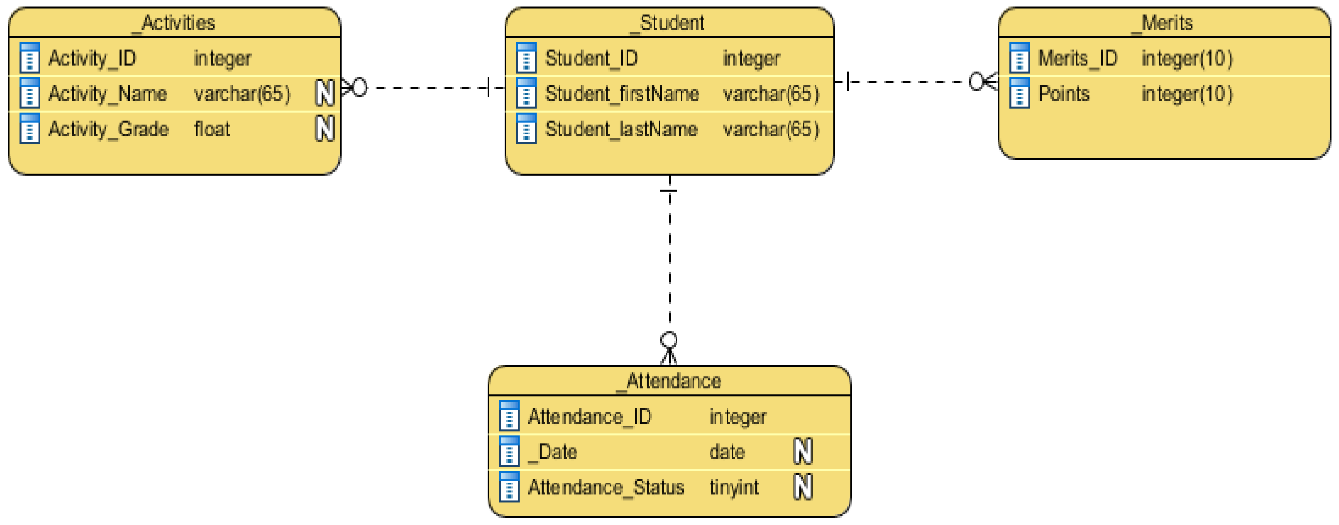
Conceptual ERD:

Conceptual ERD is the simplest type among the three ERDs. This model contains a kind of relationship between entities. In this Conceptual ERD, Student Table has one-to-many relationship with Activities Table, Merits Table, and Attendance Table.



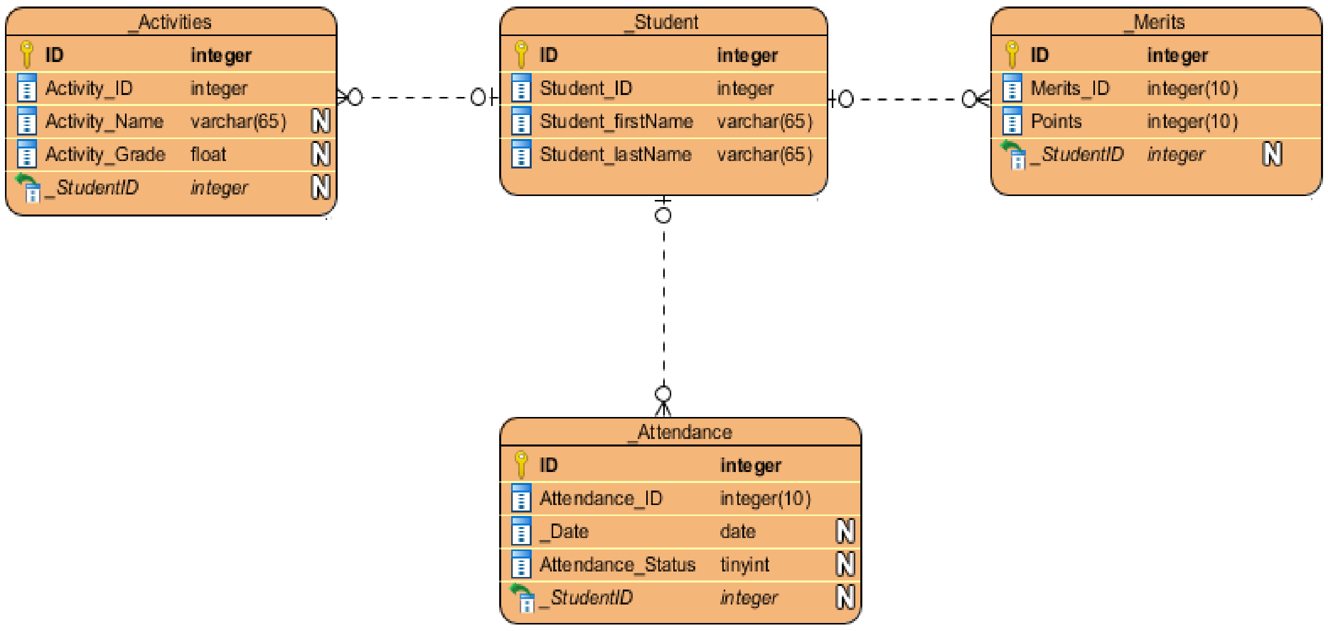
Logical ERD:

Logical ERD is more complex than Conceptual ERD since it contains the columns and its datatypes along with the relationships of the tables. Setting the datatypes for this model is optional. In this Logical ERD, each table contains its own columns with its respective datatypes.



Physical ERD:

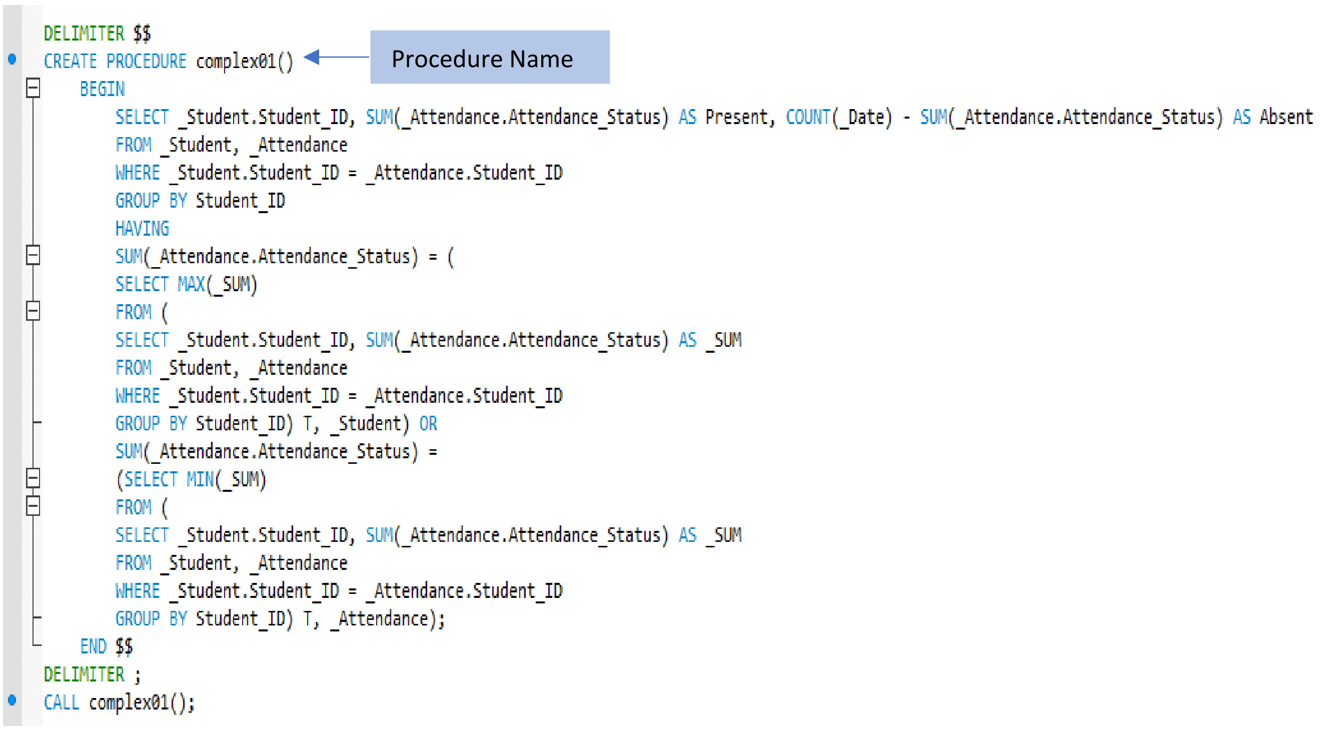
Physical ERD represents the actual design blueprint of a relational database. Accurate datatypes, constraints, primary keys and foreign keys are added to the tables. It is important to add accurate features because it represents the data that should be structured and are related in a specific database management system. In this Physical ERD, the IDs represents the primary key while the foreign keys are _StudentID. The foreign key is a column for Merits, Activities and Attendance tables that references the primary key column in Student table.



V. Stored Procedures

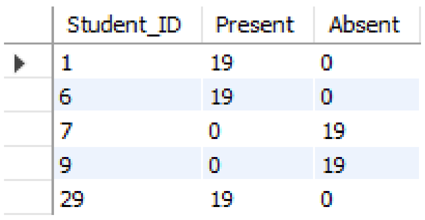
Complex Queries

1. Get the students id with the maximum and minimum absences



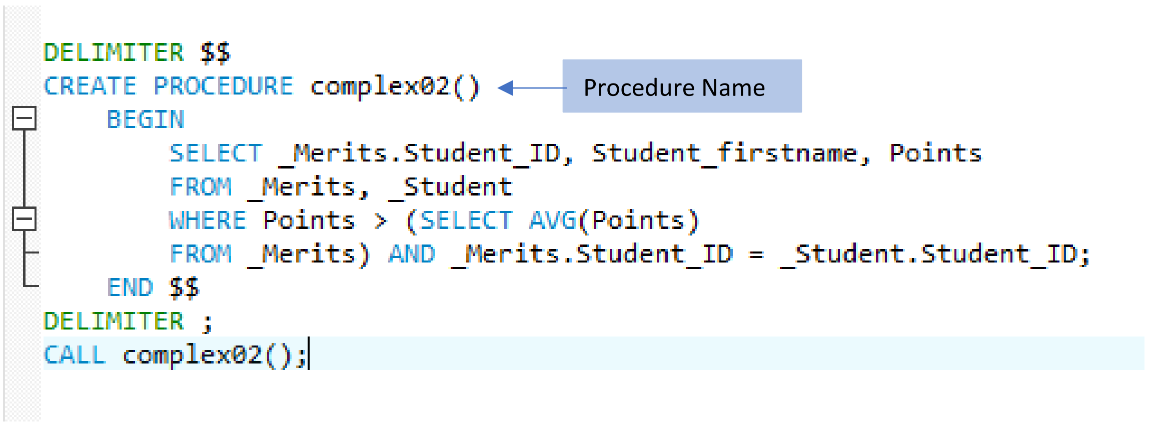
```
DELIMITER $$
CREATE PROCEDURE complex01()
    BEGIN
        SELECT _Student.Student_ID, SUM(_Attendance.Attendance_Status) AS Present, COUNT(_Date) - SUM(_Attendance.Attendance_Status) AS Absent
        FROM _Student, _Attendance
        WHERE _Student.Student_ID = _Attendance.Student_ID
        GROUP BY Student_ID
        HAVING
        SUM(_Attendance.Attendance_Status) = (
        SELECT MAX(_SUM)
        FROM (
        SELECT _Student.Student_ID, SUM(_Attendance.Attendance_Status) AS _SUM
        FROM _Student, _Attendance
        WHERE _Student.Student_ID = _Attendance.Student_ID
        GROUP BY Student_ID) T, _Student) OR
        SUM(_Attendance.Attendance_Status) =
        (SELECT MIN(_SUM)
        FROM (
        SELECT _Student.Student_ID, SUM(_Attendance.Attendance_Status) AS _SUM
        FROM _Student, _Attendance
        WHERE _Student.Student_ID = _Attendance.Student_ID
        GROUP BY Student_ID) T, _Attendance);
    END $$
DELIMITER ;
CALL complex01();
```

Query Returned



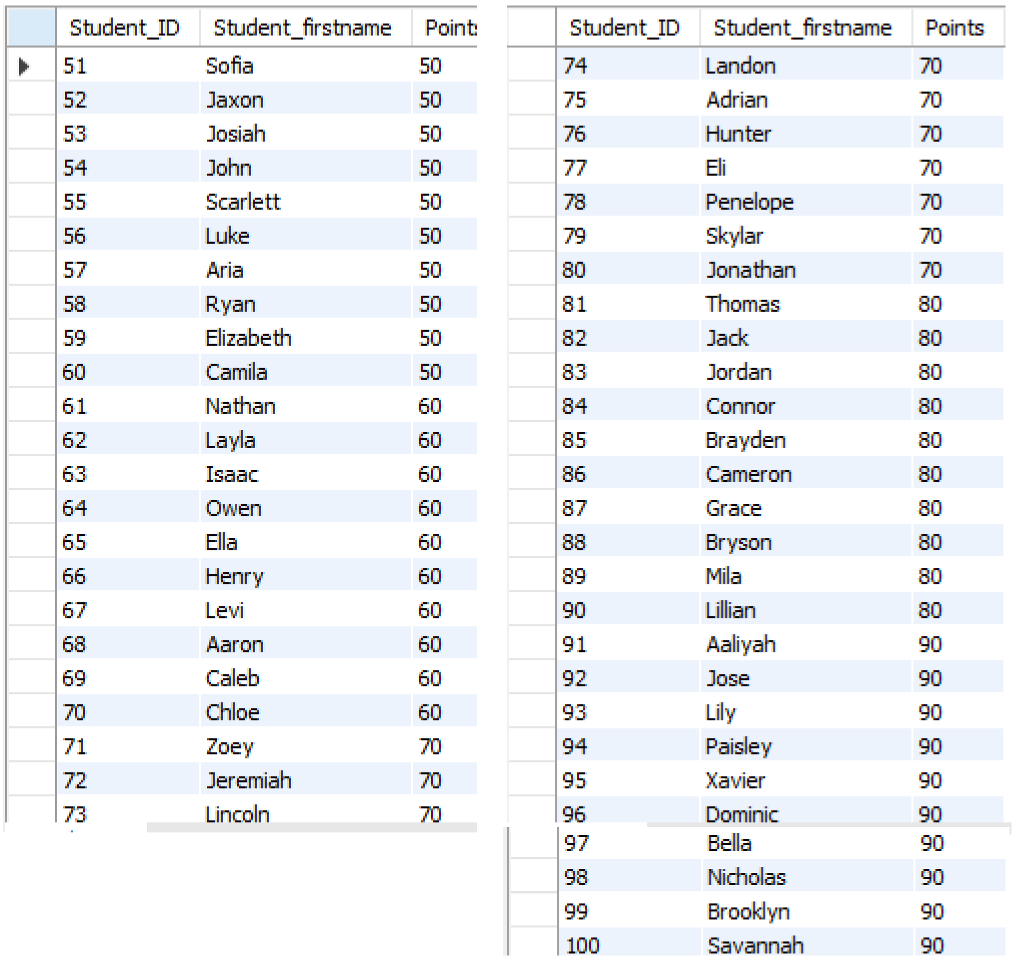
Student_ID	Present	Absent
1	19	0
6	19	0
7	0	19
9	0	19
29	19	0

2. Get the students whose total merit score is more than the average merit score of students.



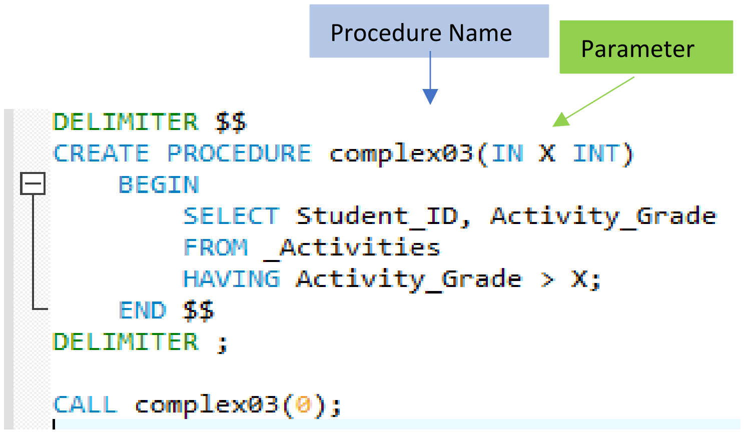
```
DELIMITER $$
CREATE PROCEDURE complex02()
    BEGIN
        SELECT _Merits.Student_ID, Student_firstname, Points
        FROM _Merits, _Student
        WHERE Points > (SELECT AVG(Points)
        FROM _Merits) AND _Merits.Student_ID = _Student.Student_ID;
    END $$
DELIMITER ;
CALL complex02();
```

Query Returned



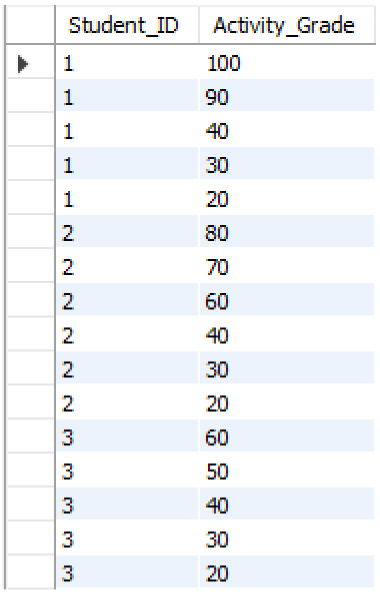
Student_ID	Student_firstname	Points
51	Sofia	50
52	Jaxon	50
53	Josiah	50
54	John	50
55	Scarlett	50
56	Luke	50
57	Aria	50
58	Ryan	50
59	Elizabeth	50
60	Camila	50
61	Nathan	60
62	Layla	60
63	Isaac	60
64	Owen	60
65	Ella	60
66	Henry	60
67	Levi	60
68	Aaron	60
69	Caleb	60
70	Chloe	60
71	Zoey	70
72	Jeremiah	70
73	Lincoln	70
74	Landon	70
75	Adrian	70
76	Hunter	70
77	Eli	70
78	Penelope	70
79	Skylar	70
80	Jonathan	70
81	Thomas	80
82	Jack	80
83	Jordan	80
84	Connor	80
85	Brayden	80
86	Cameron	80
87	Grace	80
88	Bryson	80
89	Mila	80
90	Lillian	80
91	Aaliyah	90
92	Jose	90
93	Lily	90
94	Paisley	90
95	Xavier	90
96	Dominic	90
97	Bella	90
98	Nicholas	90
99	Brooklyn	90
100	Savannah	90

3.Get the students whose percentage activities grade is greater than params.



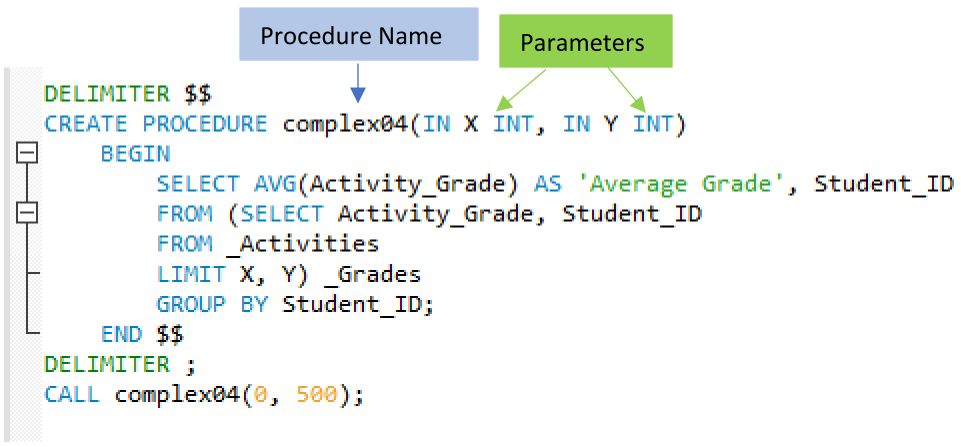
```
DELIMITER $$
CREATE PROCEDURE complex03(IN X INT)
    BEGIN
        SELECT Student_ID, Activity_Grade
        FROM _Activities
        HAVING Activity_Grade > X;
    END $$
DELIMITER ;

CALL complex03(0);
```

Query Returned



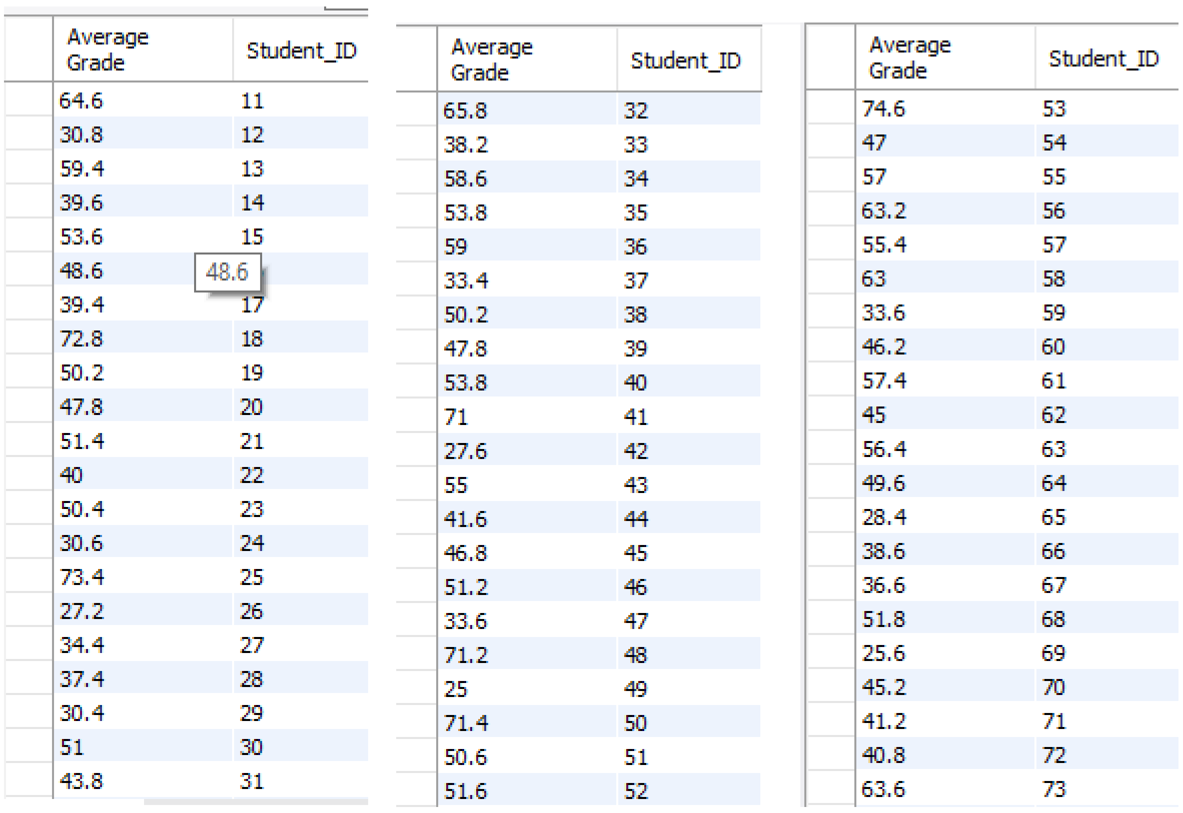
Student_ID	Activity_Grade
1	100
1	90
1	40
1	30
1	20
2	80
2	70
2	60
2	40
2	30
2	20
3	60
3	50
3	40
3	30
3	20

4. Get the average score of the students in their activities given on start_param and end_param

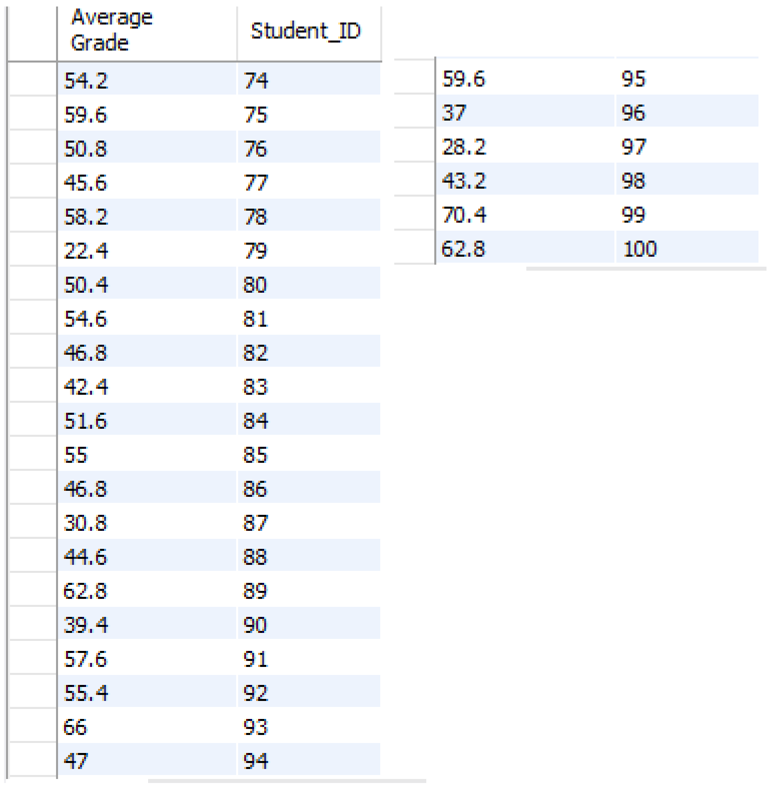


```
DELIMITER $$
CREATE PROCEDURE complex04(IN X INT, IN Y INT)
    BEGIN
        SELECT AVG(Activity_Grade) AS 'Average Grade', Student_ID
        FROM (SELECT Activity_Grade, Student_ID
        FROM _Activities
        LIMIT X, Y) _Grades
        GROUP BY Student_ID;
    END $$
DELIMITER ;
CALL complex04(0, 500);
```

Query Returned:

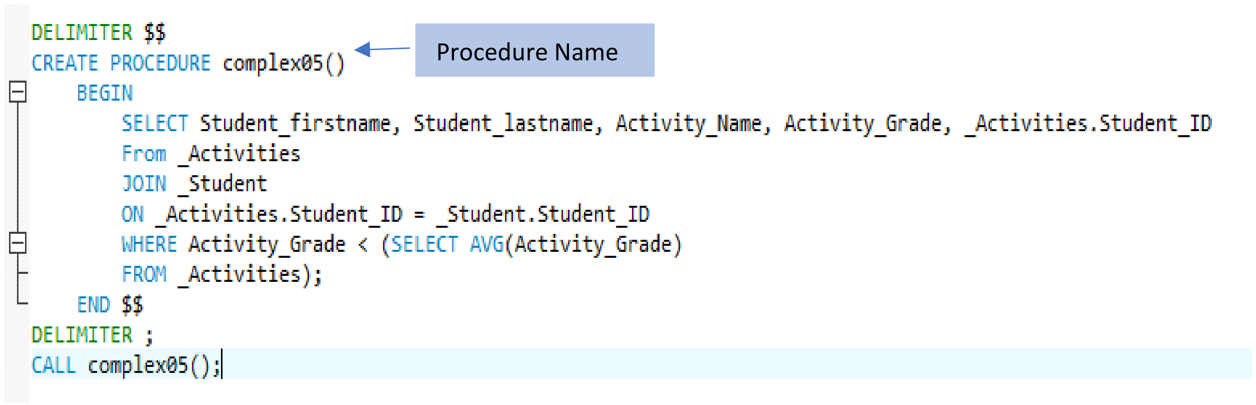


Average Grade	Student_ID
64.6	11
30.8	12
59.4	13
39.6	14
53.6	15
48.6	48.6
39.4	17
72.8	18
50.2	19
47.8	20
51.4	21
40	22
50.4	23
30.6	24
73.4	25
27.2	26
34.4	27
37.4	28
30.4	29
51	30
43.8	31
65.8	32
38.2	33
58.6	34
53.8	35
59	36
33.4	37
50.2	38
47.8	39
53.8	40
71	41
27.6	42
55	43
41.6	44
46.8	45
51.2	46
33.6	47
71.2	48
25	49
71.4	50
50.6	51
51.6	52
74.6	53
47	54
57	55
63.2	56
55.4	57
63	58
33.6	59
46.2	60
57.4	61
45	62
56.4	63
49.6	64
28.4	65
38.6	66
36.6	67
51.8	68
25.6	69
45.2	70
41.2	71
40.8	72
63.6	73



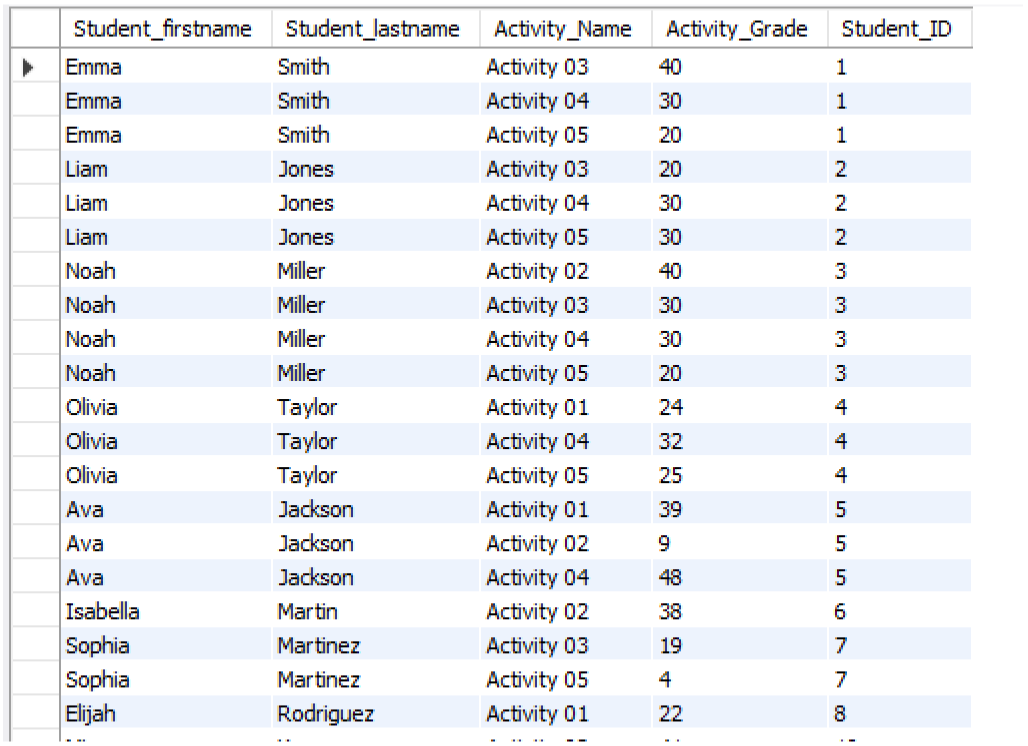
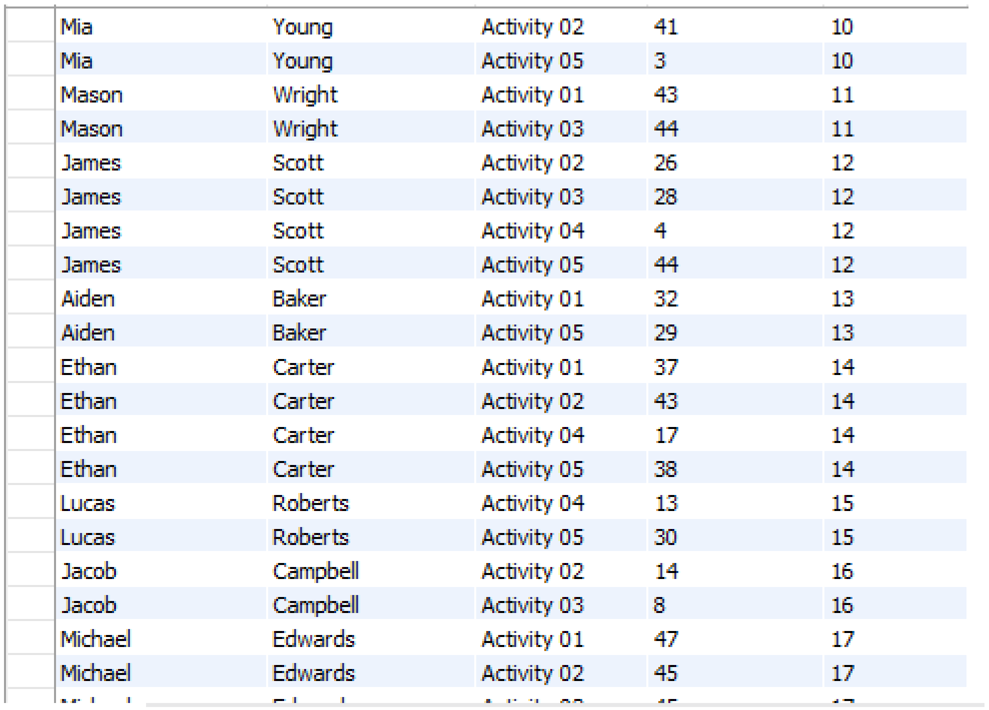
Average Grade	Student_ID
54.2	74
59.6	75
50.8	76
45.6	77
58.2	78
22.4	79
50.4	80
54.6	81
46.8	82
42.4	83
51.6	84
55	85
46.8	86
30.8	87
44.6	88
62.8	89
39.4	90
57.6	91
55.4	92
66	93
47	94
59.6	95
37	96
28.2	97
43.2	98
70.4	99
62.8	100

5. Give the student and activity name whose percentage score is less than the average activity percentage score.

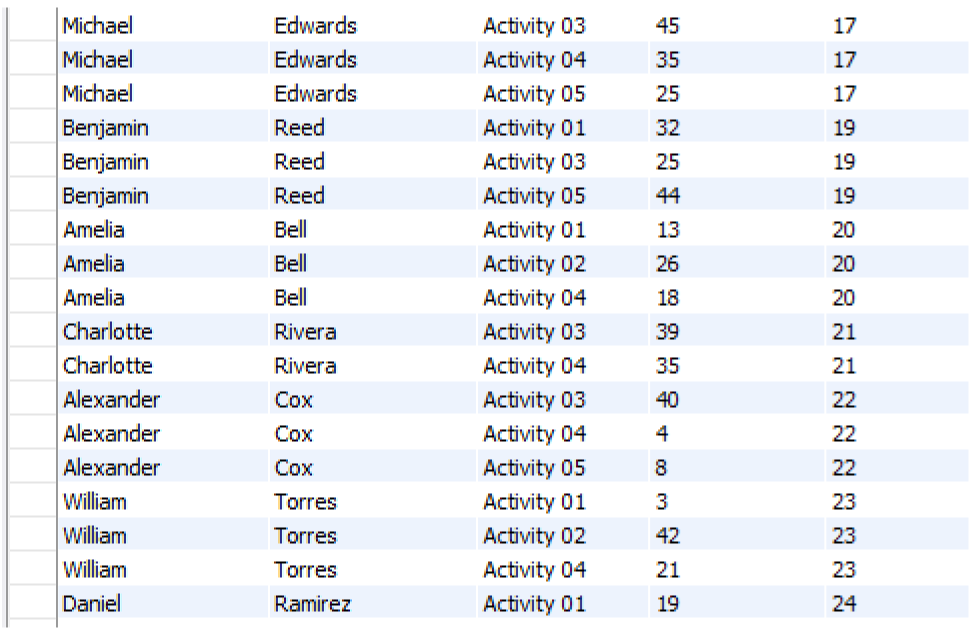


```
DELIMITER $$
CREATE PROCEDURE complex05()
    BEGIN
        SELECT Student_firstname, Student_lastname, Activity_Name, Activity_Grade, _Activities.Student_ID
        From _Activities
        JOIN _Student
        ON _Activities.Student_ID = _Student.Student_ID
        WHERE Activity_Grade < (SELECT AVG(Activity_Grade)
        FROM _Activities);
    END $$
DELIMITER ;
CALL complex05();
```

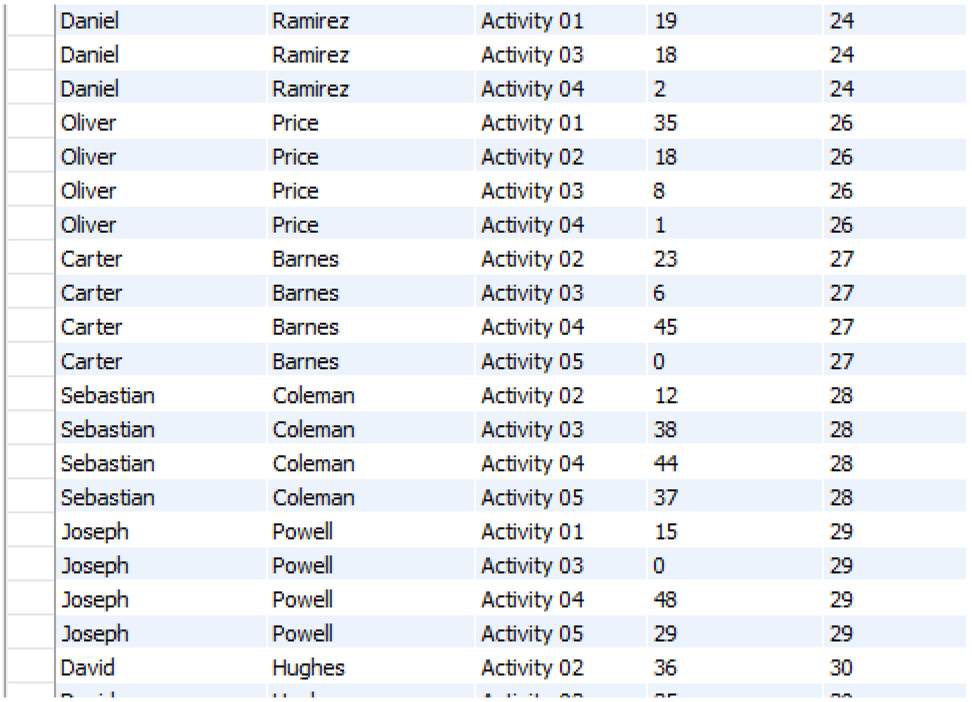
Query Returned:

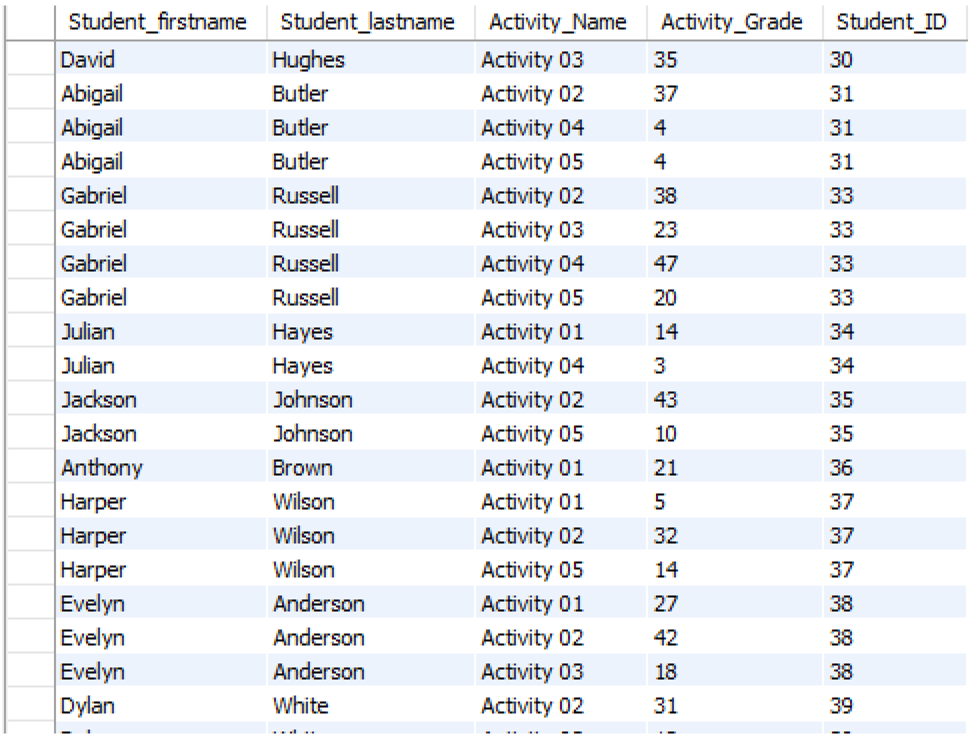
Student_firstname	Student_lastname	Activity_Name	Activity_Grade	Student_ID
Emma	Smith	Activity 03	40	1
Emma	Smith	Activity 04	30	1
Emma	Smith	Activity 05	20	1
Liam	Jones	Activity 03	20	2
Liam	Jones	Activity 04	30	2
Liam	Jones	Activity 05	30	2
Noah	Miller	Activity 02	40	3
Noah	Miller	Activity 03	30	3
Noah	Miller	Activity 04	30	3
Noah	Miller	Activity 05	20	3
Olivia	Taylor	Activity 01	24	4
Olivia	Taylor	Activity 04	32	4
Olivia	Taylor	Activity 05	25	4
Ava	Jackson	Activity 01	39	5
Ava	Jackson	Activity 02	9	5
Ava	Jackson	Activity 04	48	5
Isabella	Martin	Activity 02	38	6
Sophia	Martinez	Activity 03	19	7
Sophia	Martinez	Activity 05	4	7
Elijah	Rodriguez	Activity 01	22	8
Mia	Young	Activity 02	41	10
Mia	Young	Activity 05	3	10
Mason	Wright	Activity 01	43	11
Mason	Wright	Activity 03	44	11
James	Scott	Activity 02	26	12
James	Scott	Activity 03	28	12
James	Scott	Activity 04	4	12
James	Scott	Activity 05	44	12
Aiden	Baker	Activity 01	32	13
Aiden	Baker	Activity 05	29	13
Ethan	Carter	Activity 01	37	14
Ethan	Carter	Activity 02	43	14
Ethan	Carter	Activity 04	17	14
Ethan	Carter	Activity 05	38	14
Lucas	Roberts	Activity 04	13	15
Lucas	Roberts	Activity 05	30	15
Jacob	Campbell	Activity 02	14	16
Jacob	Campbell	Activity 03	8	16
Michael	Edwards	Activity 01	47	17
Michael	Edwards	Activity 02	45	17



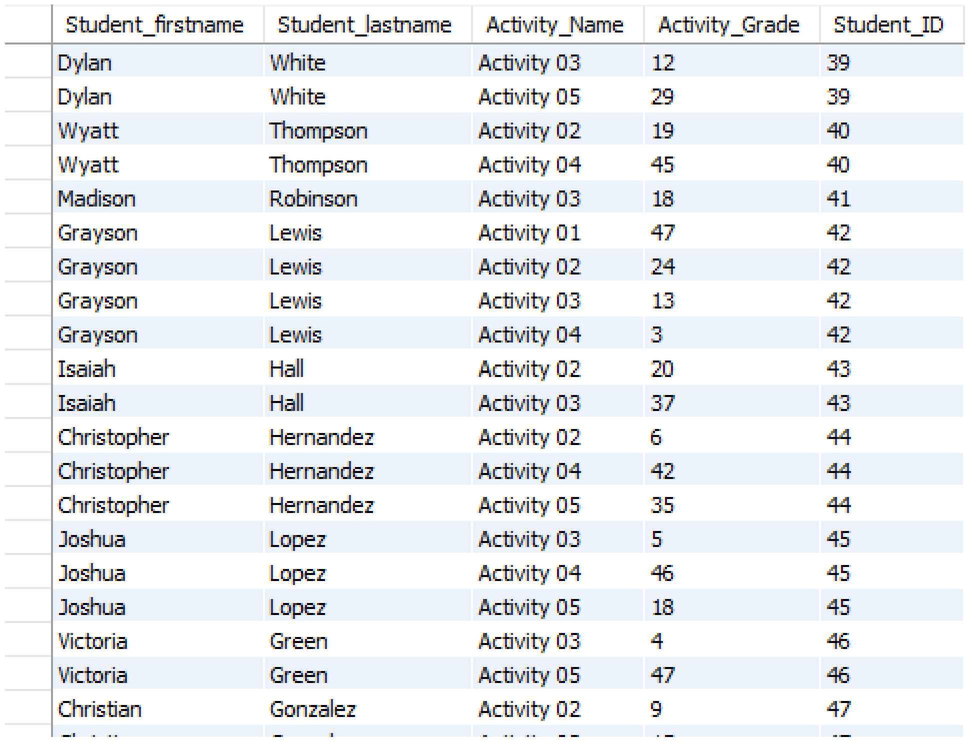
Michael	Edwards	Activity 03	45	17
Michael	Edwards	Activity 04	35	17
Michael	Edwards	Activity 05	25	17
Benjamin	Reed	Activity 01	32	19
Benjamin	Reed	Activity 03	25	19
Benjamin	Reed	Activity 05	44	19
Amelia	Bell	Activity 01	13	20
Amelia	Bell	Activity 02	26	20
Amelia	Bell	Activity 04	18	20
Charlotte	Rivera	Activity 03	39	21
Charlotte	Rivera	Activity 04	35	21
Alexander	Cox	Activity 03	40	22
Alexander	Cox	Activity 04	4	22
Alexander	Cox	Activity 05	8	22
William	Torres	Activity 01	3	23
William	Torres	Activity 02	42	23
William	Torres	Activity 04	21	23
Daniel	Ramirez	Activity 01	19	24



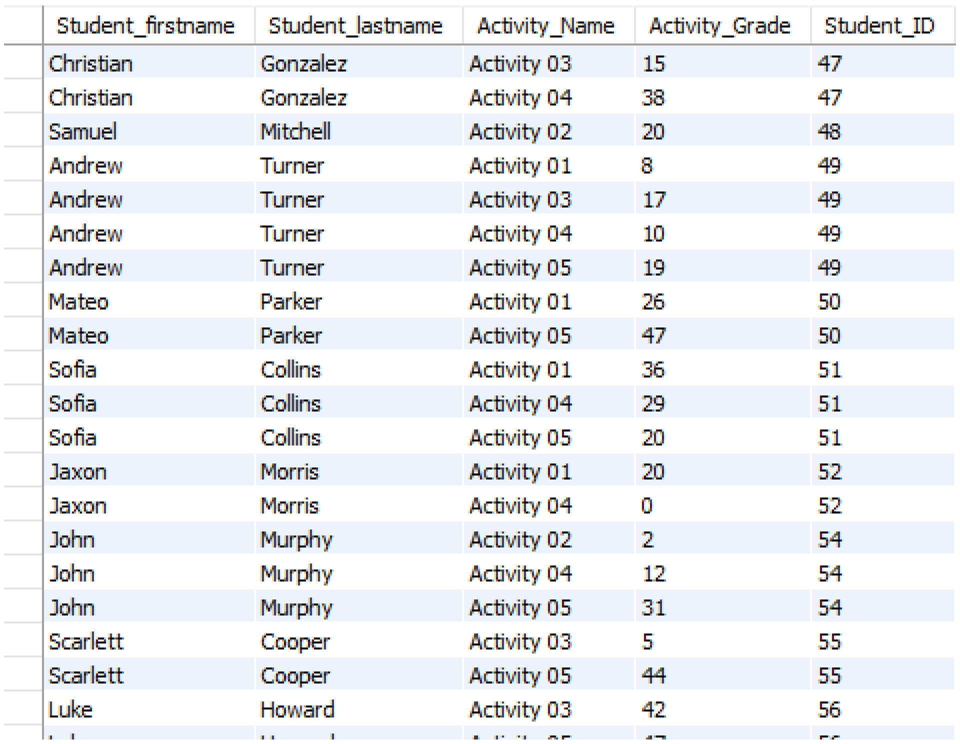
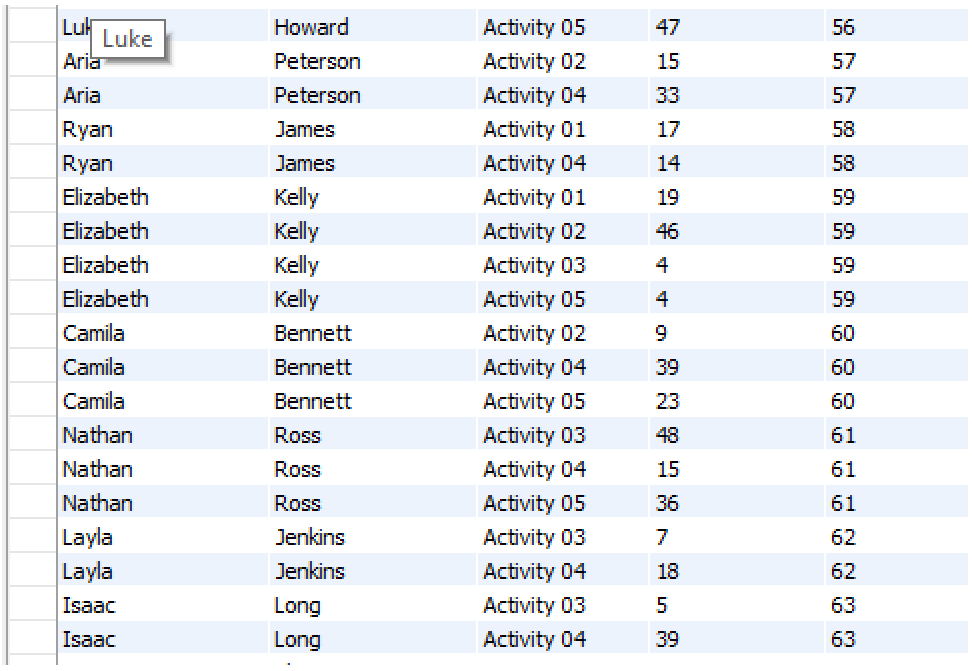
Daniel	Ramirez	Activity 01	19	24
Daniel	Ramirez	Activity 03	18	24
Daniel	Ramirez	Activity 04	2	24
Oliver	Price	Activity 01	35	26
Oliver	Price	Activity 02	18	26
Oliver	Price	Activity 03	8	26
Oliver	Price	Activity 04	1	26
Carter	Barnes	Activity 02	23	27
Carter	Barnes	Activity 03	6	27
Carter	Barnes	Activity 04	45	27
Carter	Barnes	Activity 05	0	27
Sebastian	Coleman	Activity 02	12	28
Sebastian	Coleman	Activity 03	38	28
Sebastian	Coleman	Activity 04	44	28
Sebastian	Coleman	Activity 05	37	28
Joseph	Powell	Activity 01	15	29
Joseph	Powell	Activity 03	0	29
Joseph	Powell	Activity 04	48	29
Joseph	Powell	Activity 05	29	29
David	Hughes	Activity 02	36	30



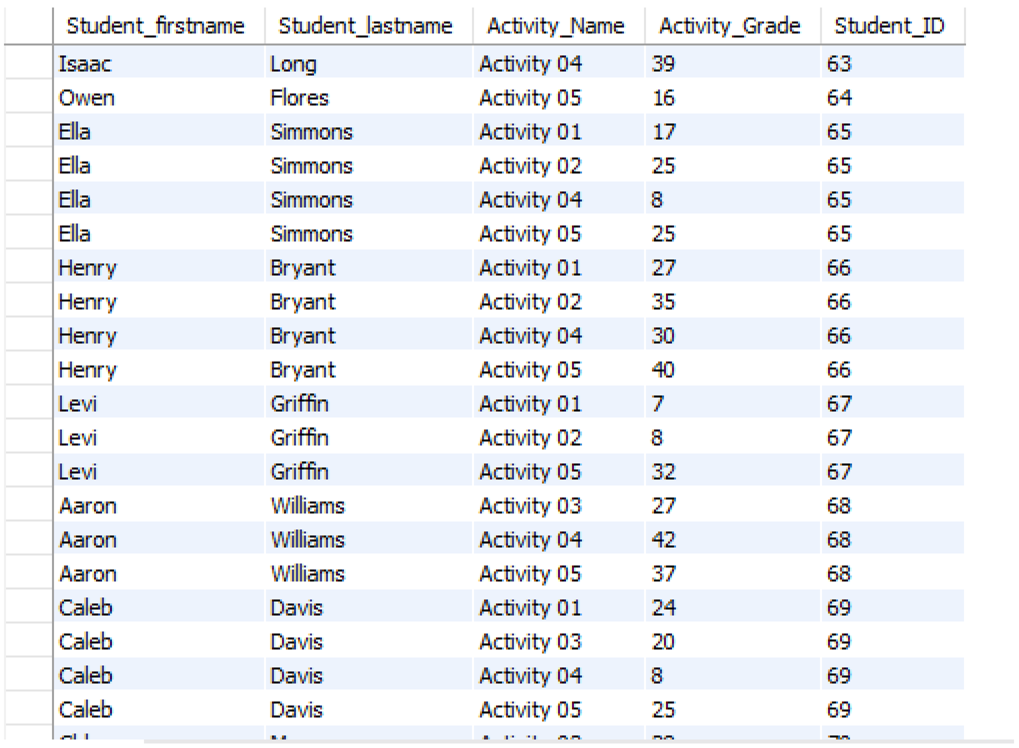
Student_firstname	Student_lastname	Activity_Name	Activity_Grade	Student_ID
David	Hughes	Activity 03	35	30
Abigail	Butler	Activity 02	37	31
Abigail	Butler	Activity 04	4	31
Abigail	Butler	Activity 05	4	31
Gabriel	Russell	Activity 02	38	33
Gabriel	Russell	Activity 03	23	33
Gabriel	Russell	Activity 04	47	33
Gabriel	Russell	Activity 05	20	33
Julian	Hayes	Activity 01	14	34
Julian	Hayes	Activity 04	3	34
Jackson	Johnson	Activity 02	43	35
Jackson	Johnson	Activity 05	10	35
Anthony	Brown	Activity 01	21	36
Harper	Wilson	Activity 01	5	37
Harper	Wilson	Activity 02	32	37
Harper	Wilson	Activity 05	14	37
Evelyn	Anderson	Activity 01	27	38
Evelyn	Anderson	Activity 02	42	38
Evelyn	Anderson	Activity 03	18	38
Dylan	White	Activity 02	31	39



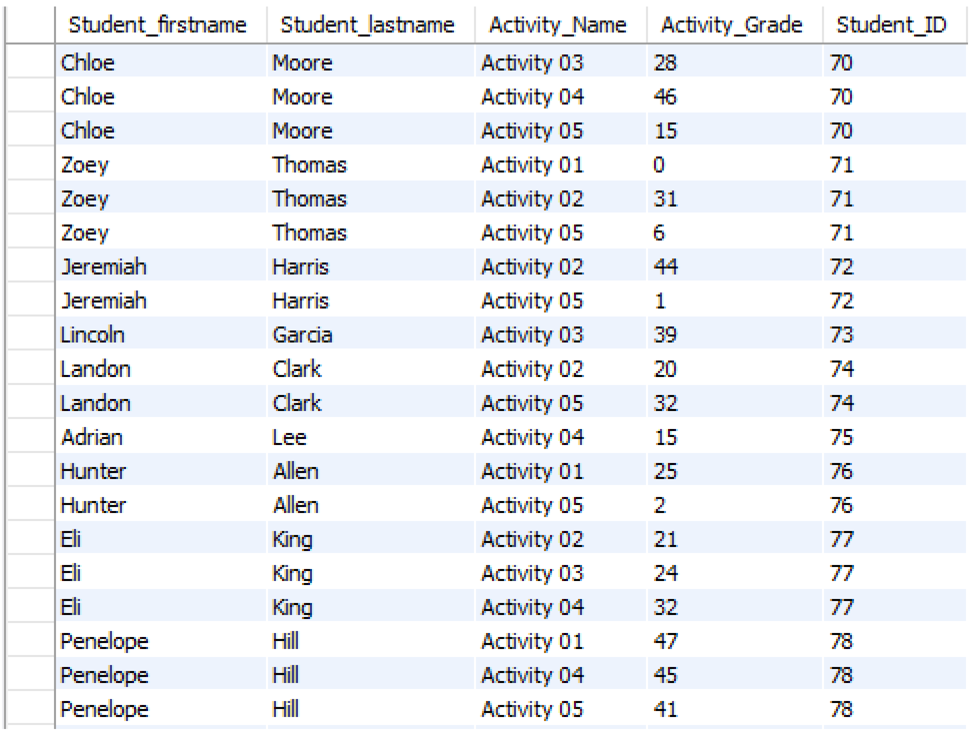
Student_firstname	Student_lastname	Activity_Name	Activity_Grade	Student_ID
Dylan	White	Activity 03	12	39
Dylan	White	Activity 05	29	39
Wyatt	Thompson	Activity 02	19	40
Wyatt	Thompson	Activity 04	45	40
Madison	Robinson	Activity 03	18	41
Grayson	Lewis	Activity 01	47	42
Grayson	Lewis	Activity 02	24	42
Grayson	Lewis	Activity 03	13	42
Grayson	Lewis	Activity 04	3	42
Isaiah	Hall	Activity 02	20	43
Isaiah	Hall	Activity 03	37	43
Christopher	Hernandez	Activity 02	6	44
Christopher	Hernandez	Activity 04	42	44
Christopher	Hernandez	Activity 05	35	44
Joshua	Lopez	Activity 03	5	45
Joshua	Lopez	Activity 04	46	45
Joshua	Lopez	Activity 05	18	45
Victoria	Green	Activity 03	4	46
Victoria	Green	Activity 05	47	46
Christian	Gonzalez	Activity 02	9	47

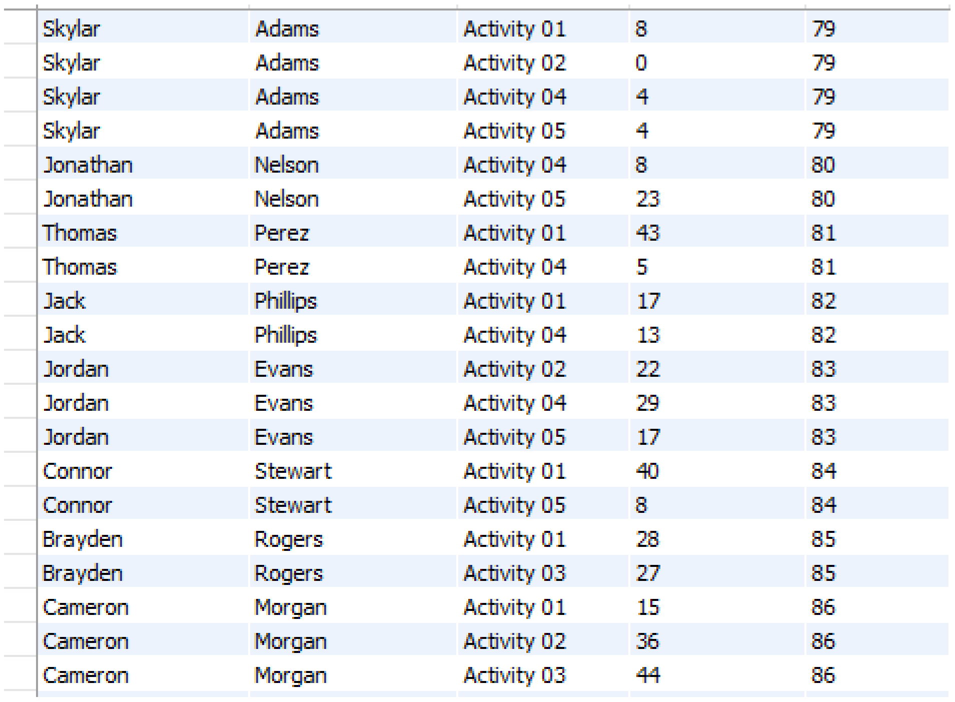
Student_firstname	Student_lastname	Activity_Name	Activity_Grade	Student_ID
Christian	Gonzalez	Activity 03	15	47
Christian	Gonzalez	Activity 04	38	47
Samuel	Mitchell	Activity 02	20	48
Andrew	Turner	Activity 01	8	49
Andrew	Turner	Activity 03	17	49
Andrew	Turner	Activity 04	10	49
Andrew	Turner	Activity 05	19	49
Mateo	Parker	Activity 01	26	50
Mateo	Parker	Activity 05	47	50
Sofia	Collins	Activity 01	36	51
Sofia	Collins	Activity 04	29	51
Sofia	Collins	Activity 05	20	51
Jaxon	Morris	Activity 01	20	52
Jaxon	Morris	Activity 04	0	52
John	Murphy	Activity 02	2	54
John	Murphy	Activity 04	12	54
John	Murphy	Activity 05	31	54
Scarlett	Cooper	Activity 03	5	55
Scarlett	Cooper	Activity 05	44	55
Luke	Howard	Activity 03	42	56
Luke	Howard	Activity 05	47	56
Aria	Peterson	Activity 02	15	57
Aria	Peterson	Activity 04	33	57
Ryan	James	Activity 01	17	58
Ryan	James	Activity 04	14	58
Elizabeth	Kelly	Activity 01	19	59
Elizabeth	Kelly	Activity 02	46	59
Elizabeth	Kelly	Activity 03	4	59
Elizabeth	Kelly	Activity 05	4	59
Camila	Bennett	Activity 02	9	60
Camila	Bennett	Activity 04	39	60
Camila	Bennett	Activity 05	23	60
Nathan	Ross	Activity 03	48	61
Nathan	Ross	Activity 04	15	61
Nathan	Ross	Activity 05	36	61
Layla	Jenkins	Activity 03	7	62
Layla	Jenkins	Activity 04	18	62
Isaac	Long	Activity 03	5	63
Isaac	Long	Activity 04	39	63



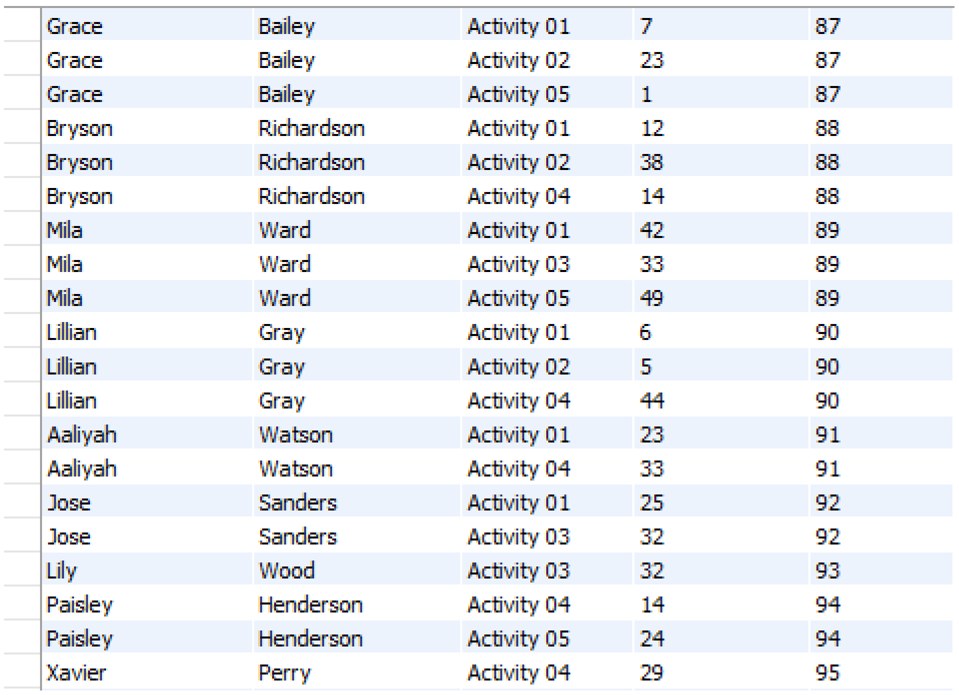
Student_firstname	Student_lastname	Activity_Name	Activity_Grade	Student_ID
Isaac	Long	Activity 04	39	63
Owen	Flores	Activity 05	16	64
Ella	Simmons	Activity 01	17	65
Ella	Simmons	Activity 02	25	65
Ella	Simmons	Activity 04	8	65
Ella	Simmons	Activity 05	25	65
Henry	Bryant	Activity 01	27	66
Henry	Bryant	Activity 02	35	66
Henry	Bryant	Activity 04	30	66
Henry	Bryant	Activity 05	40	66
Levi	Griffin	Activity 01	7	67
Levi	Griffin	Activity 02	8	67
Levi	Griffin	Activity 05	32	67
Aaron	Williams	Activity 03	27	68
Aaron	Williams	Activity 04	42	68
Aaron	Williams	Activity 05	37	68
Caleb	Davis	Activity 01	24	69
Caleb	Davis	Activity 03	20	69
Caleb	Davis	Activity 04	8	69
Caleb	Davis	Activity 05	25	69



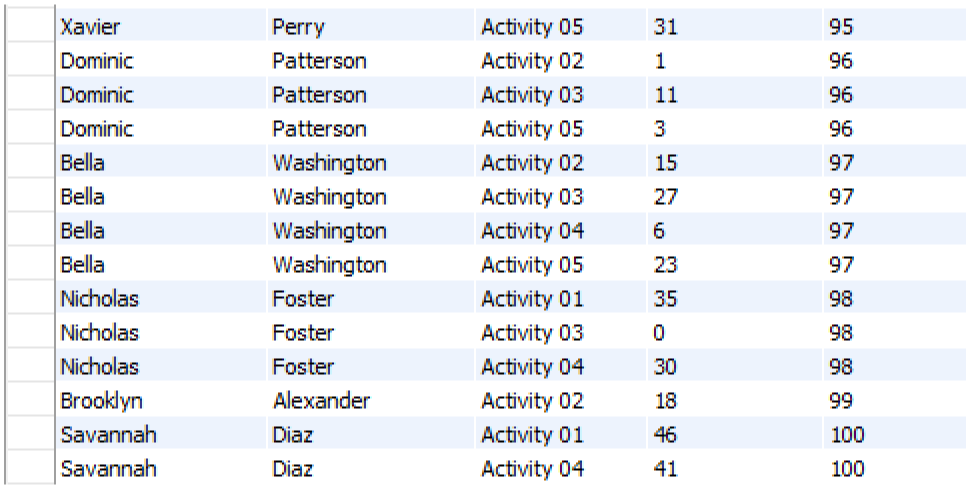
Student_firstname	Student_lastname	Activity_Name	Activity_Grade	Student_ID
Chloe	Moore	Activity 03	28	70
Chloe	Moore	Activity 04	46	70
Chloe	Moore	Activity 05	15	70
Zoey	Thomas	Activity 01	0	71
Zoey	Thomas	Activity 02	31	71
Zoey	Thomas	Activity 05	6	71
Jeremiah	Harris	Activity 02	44	72
Jeremiah	Harris	Activity 05	1	72
Lincoln	Garcia	Activity 03	39	73
Landon	Clark	Activity 02	20	74
Landon	Clark	Activity 05	32	74
Adrian	Lee	Activity 04	15	75
Hunter	Allen	Activity 01	25	76
Hunter	Allen	Activity 05	2	76
Eli	King	Activity 02	21	77
Eli	King	Activity 03	24	77
Eli	King	Activity 04	32	77
Penelope	Hill	Activity 01	47	78
Penelope	Hill	Activity 04	45	78
Penelope	Hill	Activity 05	41	78



Skylar	Adams	Activity 01	8	79
Skylar	Adams	Activity 02	0	79
Skylar	Adams	Activity 04	4	79
Skylar	Adams	Activity 05	4	79
Jonathan	Nelson	Activity 04	8	80
Jonathan	Nelson	Activity 05	23	80
Thomas	Perez	Activity 01	43	81
Thomas	Perez	Activity 04	5	81
Jack	Phillips	Activity 01	17	82
Jack	Phillips	Activity 04	13	82
Jordan	Evans	Activity 02	22	83
Jordan	Evans	Activity 04	29	83
Jordan	Evans	Activity 05	17	83
Connor	Stewart	Activity 01	40	84
Connor	Stewart	Activity 05	8	84
Brayden	Rogers	Activity 01	28	85
Brayden	Rogers	Activity 03	27	85
Cameron	Morgan	Activity 01	15	86
Cameron	Morgan	Activity 02	36	86
Cameron	Morgan	Activity 03	44	86



Grace	Bailey	Activity 01	7	87
Grace	Bailey	Activity 02	23	87
Grace	Bailey	Activity 05	1	87
Bryson	Richardson	Activity 01	12	88
Bryson	Richardson	Activity 02	38	88
Bryson	Richardson	Activity 04	14	88
Mila	Ward	Activity 01	42	89
Mila	Ward	Activity 03	33	89
Mila	Ward	Activity 05	49	89
Lillian	Gray	Activity 01	6	90
Lillian	Gray	Activity 02	5	90
Lillian	Gray	Activity 04	44	90
Aaliyah	Watson	Activity 01	23	91
Aaliyah	Watson	Activity 04	33	91
Jose	Sanders	Activity 01	25	92
Jose	Sanders	Activity 03	32	92
Lily	Wood	Activity 03	32	93
Paisley	Henderson	Activity 04	14	94
Paisley	Henderson	Activity 05	24	94
Xavier	Perry	Activity 04	29	95



Xavier	Perry	Activity 05	31	95
Dominic	Patterson	Activity 02	1	96
Dominic	Patterson	Activity 03	11	96
Dominic	Patterson	Activity 05	3	96
Bella	Washington	Activity 02	15	97
Bella	Washington	Activity 03	27	97
Bella	Washington	Activity 04	6	97
Bella	Washington	Activity 05	23	97
Nicholas	Foster	Activity 01	35	98
Nicholas	Foster	Activity 03	0	98
Nicholas	Foster	Activity 04	30	98
Brooklyn	Alexander	Activity 02	18	99
Savannah	Diaz	Activity 01	46	100
Savannah	Diaz	Activity 04	41	100

STORED PROCEDURES CRUD

A. Procedure Name: GetNumberofStudents

B. Parameters: None
C. Query Returned: Number of students per section

1. A. Procedure Name: GetAllStudentInformation
   B. Parameters: None
   C. Query Returned: Students first and last name

2. A. Procedure Name: CreateStudent
   B. Parameters: [IN] X (VARCHAR (65)), [IN] Y (VARCHAR(65))
   C. Query Returned: ID of the created student

3. A. Procedure Name: ReadStudent
   B. Parameters: [IN] X (INT)
   C. Query Returned: Returns student ID, and first and last name.

4. A. Procedure Name: UpdateStudent
   B. Parameters: (IN X INT, IN Y VARCHAR(65), IN Z VARCHAR(65))
   C. Query Returned: None

5. A. Procedure Name: DeleteStudent
   B. Parameters: (IN X INT)
   C. Query Returned: None

6. A. Procedure Name: InsertMerit
   B. Parameters: (IN X INT, IN Y INT)
   C. Query Returned: None

7. A. Procedure Name: UpdateMerit
   B. Parameters: (IN X INT, IN Y INT)
   C. Query Returned: None

8. A. Procedure Name: ViewMerits()
   B. Parameters: None
   C. Query Returned: Returns student ID and Points

9. A. Procedure Name: DeleteMerit
   B. Parameters: (IN X INT)
   C. Query Returned: None

10. A. Procedure Name: InsertAttendance
    B. Parameters: (IN X INT)
    C. Query Returned: None

11. A. Procedure Name: ViewStudentsAttendance
    B. Parameters: (IN X INT)
    C. Query Returned: Returns student ID, first name, Date and attendance status.

12. A. Procedure Name: UpdateStudentAttendance
    B. Parameters: (IN X INT, IN A BOOLEAN, IN B BOOLEAN, IN C BOOLEAN,
    IN D BOOLEAN, IN E BOOLEAN, IN F BOOLEAN, IN G BOOLEAN, IN H BOOLEAN, IN I BOOLEAN,
    IN J BOOLEAN, IN K BOOLEAN, IN L BOOLEAN, IN M BOOLEAN, IN N BOOLEAN, IN O BOOLEAN,
    IN P BOOLEAN, IN Q BOOLEAN, IN R BOOLEAN, IN S BOOLEAN)
    C. Query Returned: None

13. A. Procedure Name: DeleteAttendance
    B. Parameters: (IN Y INT)
    C. Query Returned: None

14. A. Procedure Name: ViewStudentWhoisPresent
    B. Parameters: (IN X Date)
    C. Query Returned: Returns student who is present in the given date.

15. A. Procedure Name: InsertActivities
    B. Parameters: (IN X INT, IN Y VARCHAR(65), IN Z INT)
    C. Query Returned: Activity ID of the inserted activity

16. A. Procedure Name: ReadStudentActivities

B. Parameters: (IN X INT)
C. Query Returned: Returns activities of a given student ID.

17. A. Procedure Name: ReadActivity
    B. Parameters: (IN X VARCHAR(65))
    C. Query Returned: Returns student ID and grade for a given activity name.

18. A. Procedure Name: UpdateActivity
    B. Parameters: (IN Z INT, IN X VARCHAR(65), IN Y INT)
    C. Query Returned: None

19. A. Procedure Name: DeleteActivity
    B. Parameters: (IN X INT)
    C. Query Returned: None