

Case Study
Students reward system

Name

Mehta Neelkumar Hirenkumar

Limbachiya Om

id

202001435

202001443

Lab group - 9.13



Final SRS

Q1.Understand & Complete the Description of the Case Study/Problem Domain given to you:-

Purpose:-

- ❖ The students reward system is a very important part of the education system. Via positive encouragement, the system tells students what to do, how to do, and to what extent they should do it.
- ❖ Reward is always a central approach to personal training. Therefore, the student reward system is leading the development of study atmosphere and student training.
- ❖ It has clear roles in leading and guiding, and receives a great amount of attention from faculties, students, parents, and the whole society.
- ❖ The purpose of this system is to record student's basic details, result, subject details ,academic and non-academic activities and total points that students earn,etc.

Intended Audience and Reading Suggestions:-

- ❖ Each student can view their own academic and non-academic reward points and results.
- ❖ Professor can view every student's result and reward points.
- ❖ Professors can edit only their own subject's result/point.
- ❖ The Dean had full access to edit and view every student's result/point.

Product Scope:-

- ❖ The scope of this system is to reward top 5 students who get the highest points at the end of semester.
- ❖ Because of this system students will show interest and raise their participation in everyday classroom tasks, responsibilities and learning.
- ❖ Every success story helps students become more self-confident. They are proud and also encouraged to achieve another successful result.
- ❖ It is also a process of helping students build confidence, self-recognition, and identification of the goals of their education.
- ❖ The students reward system is a process of forming an educational guide. By rewarding the best students, establishes a model and benchmark of learning to guide students to learn from the best.
- ❖ Overall, the running of the students reward system is a feedback of the result of education as well as a process of conducting the positive education.
- ❖ The settings of the award programs should be centered at a wide range of quality education and training for students, including the intellectual, physical, moral, and aesthetic aspects.
- ❖ It should not only reward the significant improvement in academics, but also take into account students' personality development. It needs to reward both the outstanding academic performances and the outstanding performances in other areas of moral, physical, aesthetic and other aspects.

Description:-

Requirements:-

- ❖ Our system will keep track of student's basic details, result, subject details ,academic and non-academic activities and total points that students earn,etc.

- ❖ At the end of every semester five students will get a reward which will be given on the basis of academic and non-academic activities points.

The user can get the following details about the given system :

1. Student name
2. Student id
3. Student's branch
4. Academic year
5. Subject name
6. Subject id
7. Professor name
8. Academic Id
9. Quiz point
10. Exam point
11. Bonus point
12. Lab point
13. Subject point
14. Total academic point
15. Non-Academic Id
16. Physical activity point
17. Responsibility point
18. Total non-academic point
19. Total reward point
20. Reward id

This system contains following tables:

- ❖ Student :- It contains basic details of a student like student name, student id, student's branch , academic year.
- ❖ Subject:- It contains details of the subject like subject name, subject id, professor name (who takes that course).
- ❖ Sub points:- All the subject has subject points. It contains details of total academic point which contains student name, Ac id, subject name , subject

id, quiz point, exam point, bonus point, lab point and subject point (total points of a subject)

- ❖ Academic:- It contains details of academic points earned by students which contains student name, Ac id, total academic points. It depends on sub points.
- ❖ Non-academic points :- It contains details of total non-academic points which contains student name, NAc id, physical activity point, responsibility point and total non-academic points.
- ❖ Non-academic:- It contains details of non-academic points earned by students which contains student name, NAc id , total non-academic points. It constructed on the basis of non-academic points.
- ❖ Reward:- By combining total academic points and total non-academic points and adding them we get total reward points. It contains student name, student id, Reward id, total academic point, total non-academic points, total reward points.

2. Document the Requirements Collection/ Fact Finding Phase

ii. Background Reading/s

1. Description of each reading and references:

- Database System Concepts (Seventh edition), by Abraham Silberschatz, Henry F. Korth, S. Sudarshan - From this article, we understood the DBMS concepts like E-R Model, Database designing etc.
- <https://youtube.com/playlist?list=PLxCzCOWd7aiFAN6l8CuViBuCdJgiOkT2Y> - From this video we understood the concepts of dbms.

2. List the combined Requirements gathered from Background Reading:

- A Database management system is required to maintain and update all the

information about details of academic and non-academic points of every student.

- All the basic facilities that are expected to be available like keeping a record of students.
- Students, Professor and Dean will be assigned different types of view permissions to show details about result, academic points and non-academic points ,etc.

(ii) Interviews

a) Student interview

Interview Plan

System : Student Reward System

Project Reference : G6/S9/9.13/2022

Interviewee:

1) Jainam Bhavsar (Role Play)

Designation: Student

Interviewer:

1) Neel Mehta

Designation : Database Developer

2) Om Limbachiya

Designation : Database Developer

Date: 29/9/2022

Time : 22:00

Duration : 45 minutes

Place: HOR Men (Boys Hostel)

Purpose of Interview :

1. Preliminary meeting to identify problems and requirements regarding updating reward points on a regular basis.
2. Any other problems regarding reward system management.

Agenda :

- Introduction and current status
- Initial ideas and follow-up action regarding the student result points management.
- About the needs and complaints of the students and how you handle those issues.
- Activities that students prefer more.

Documents to be brought to the Interview:

- Any documents relating to the current activities and current needs of the students.

Interview Summary:**System : Student Reward System****Project Reference : G6/S9/9.13/2022****Interviewee:**

1)Jainam Bhavsar(Role Play)

Designation: Student**Interviewer:**

1)Neel Mehta

Designation : Database Developer

2)Om Limbachiya

Designation : Database Developer**Date: 29/9/2022****Time : 22:00****Duration : 45 minutes****Place: HOR Men(Boys Hostel)****Purpose of Interview:**

1. Preliminary meeting to identify problems and requirements regarding updating reward points on a regular basis.
2. Any other problems or suggestions regarding reward system management.

Summary of the Interview:

- Sometimes points of one student get into another student's result (updating issue).
- Marks not updated on a regular basis like quizzes and labs marks are not updated regularly (Late updation of points).
- Students prefer exams and labs marking more than quizzes and attendance.
- Ratio of Academic and Non-academic is very low.
- Taking too much time to announce results.

b)TA's interview:-

Interview Plan

System: Student Reward System

Project Reference: G6/S9/9.13/2022

Interviewee:

1)Swapnil Thakkar(Role Play)

Designation: TA(DAIICT)

Interviewer:

1)Neel Mehta

Designation : Business Development Executive

2)Om Limbachiya

Designation : Developer

Date: 29/09/2022 **Time:** 18:00
Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

1. Preliminary meeting to identify problems and requirements regarding updating reward points on a regular basis.
2. Any other problems or suggestions regarding reward system management.

Agenda:

- Introduction and current status
- Regarding Student reward points
- Activities that TA prefer more.
- Suggestion that needs to be improved in this system.

Documents to be brought to the interview:

- Any documents relating to the current activities and current needs of the students.

Interview Summary:

System: Student Reward System

Project Reference : G6/S9/9.13/2022

Interviewee:

1)Swapnil Thakkar(Role Play)

Designation: TA

Interviewer:

1)Neel Mehta

Designation : Business Development Executive

2)Om Limbachiya

Designation : Developer

Date: 29/09/2022 **Time:** 18:00

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

1. Preliminary meeting to identify problems and requirements regarding updating reward points on a regular basis.
2. Any other problems or suggestions regarding reward system management.

Summary of the Interview:

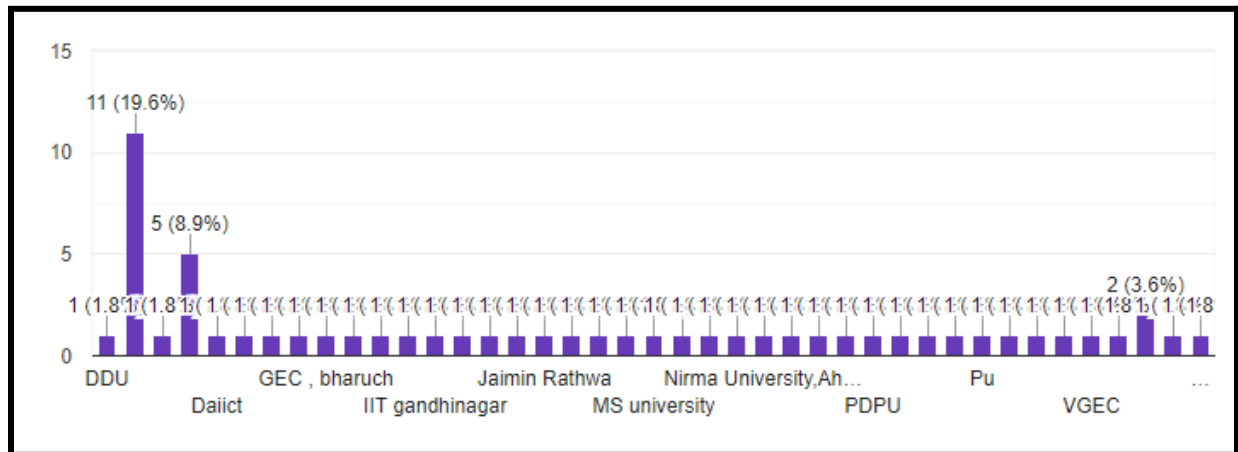
- Regular feedback from TA about students and their works will be taken and improvements will be done accordingly.
- The reward points of Students will be checked from time to time.
- In the marking system, exams had to cover more weightage than quizzes,viva and attendance.
- The weekly quiz or Lab will be taken but not regularly updated the marks of students(update problem).

Combined Requirements gathered from the Interviews:

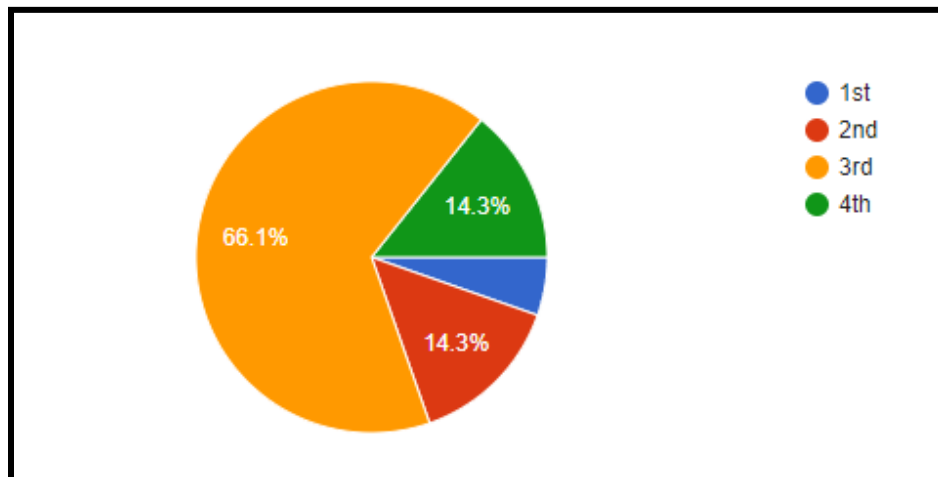
- Feedback should be received from the Students groups about different aspects like update reward points,too much time to announce results,etc.
- For the progress of students, the reward points will be checked on a regular basis.
- Give more preference to exams rather than quizzes and labs.

(iii)Questionnaire/s:-

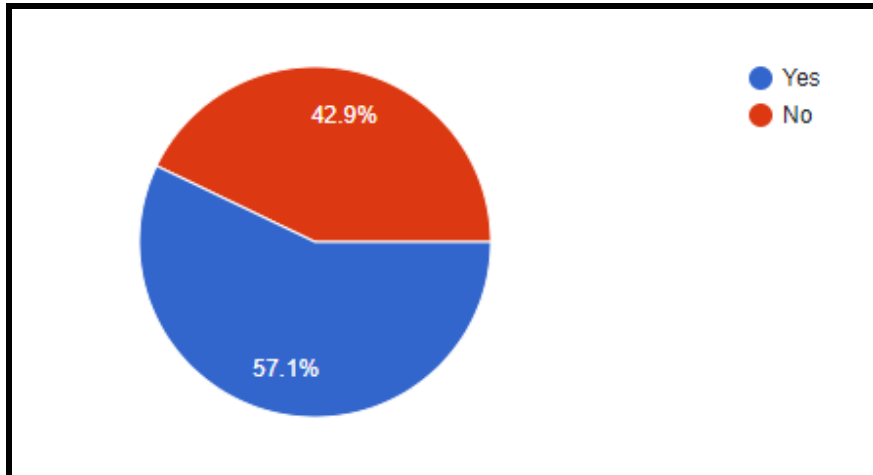
Collage name :-



In which year you are studying?

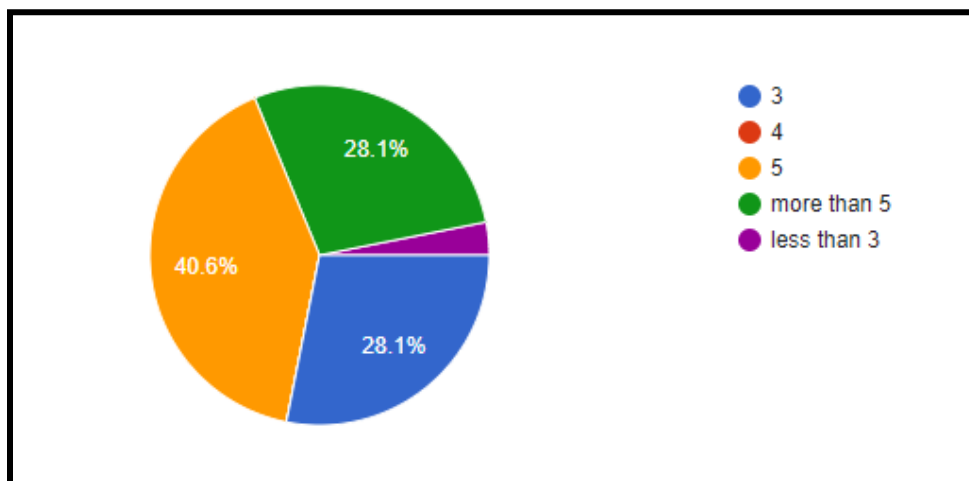


Is there any kind of reward system in your collage?

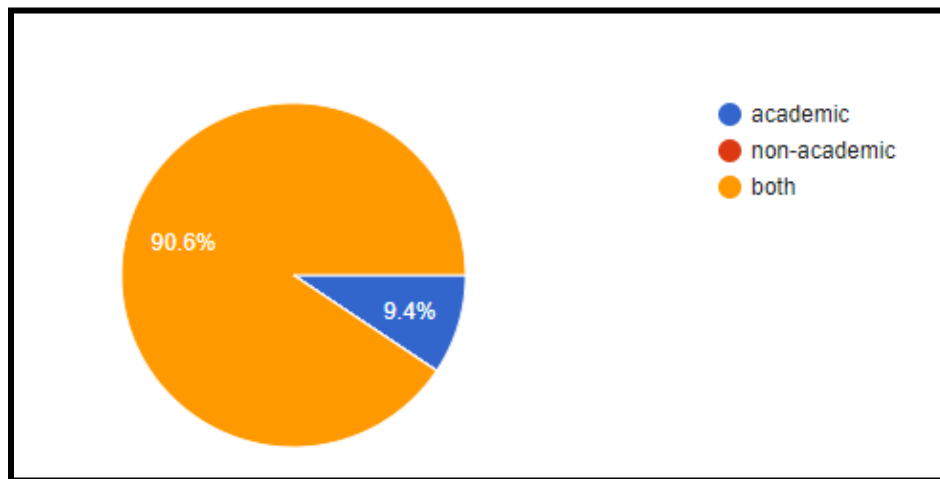


If yes...

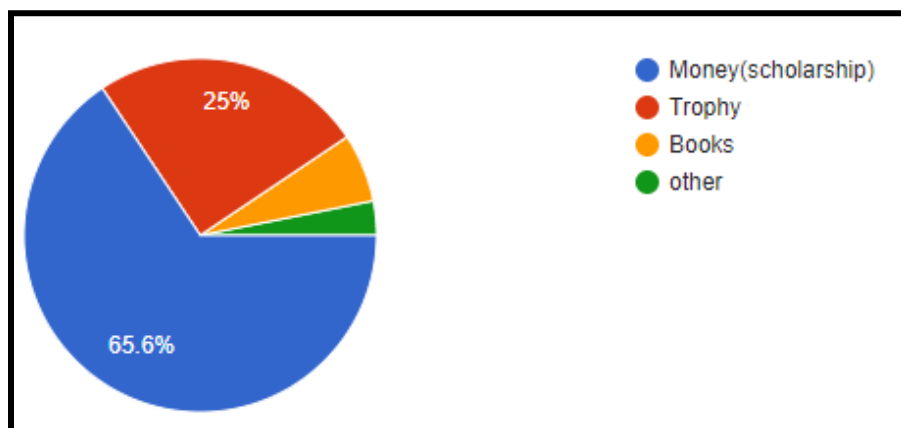
How many students get rewards?



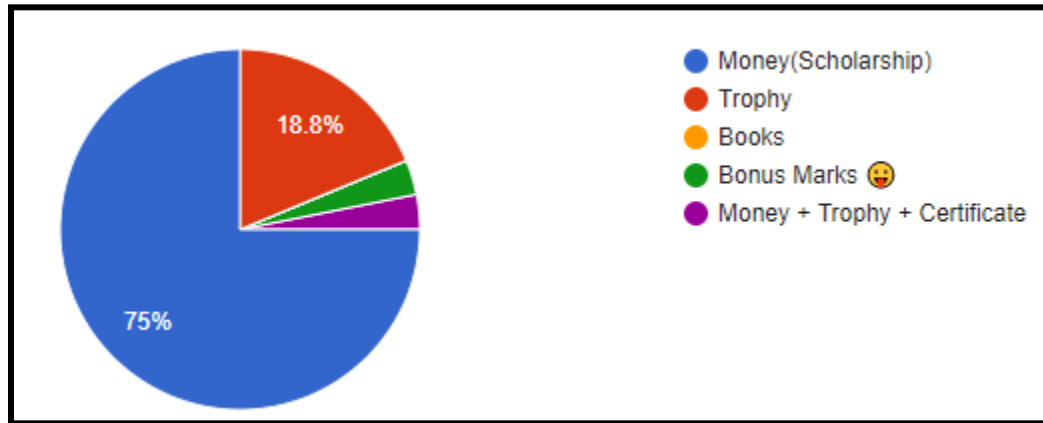
For which activities they give reward?



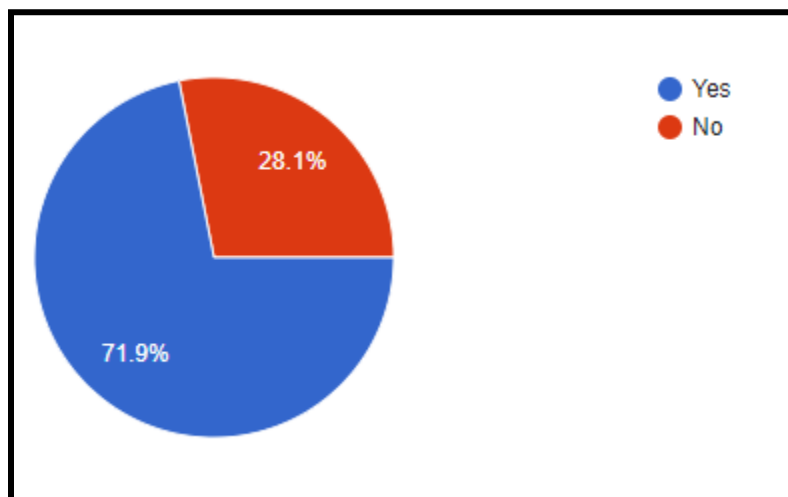
Which type of reward they give?



Which type of reward you like?



Do you get rewards on time after announcing the result?



Suggestion:-

Please note any suggestions to improve reward system...

7 responses

Nothing

.

😞

Have more interesting events to motivate student to participate in. Have a strict rule to give reward just after completing the event.

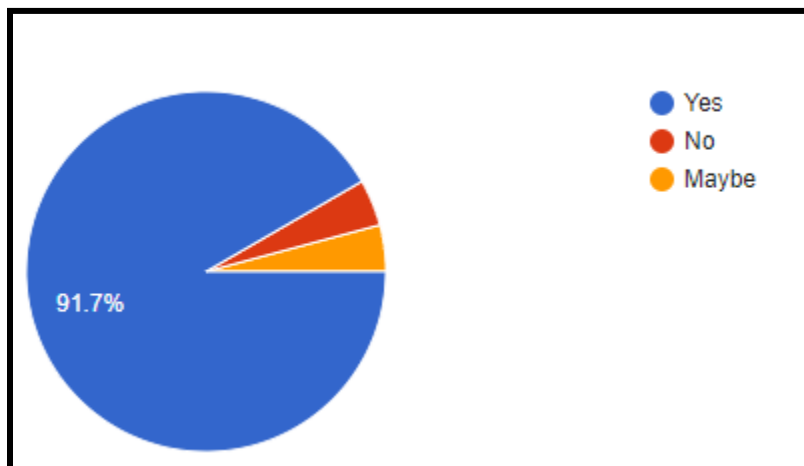
Its all good

After Announcing the result, we don't get reward on time so you have to think on that matter.

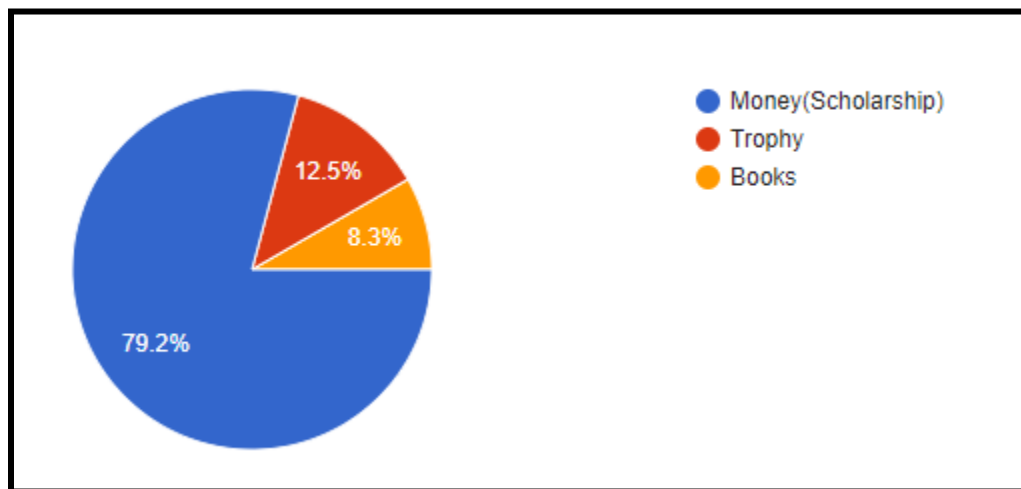
No comments 😊

If no...

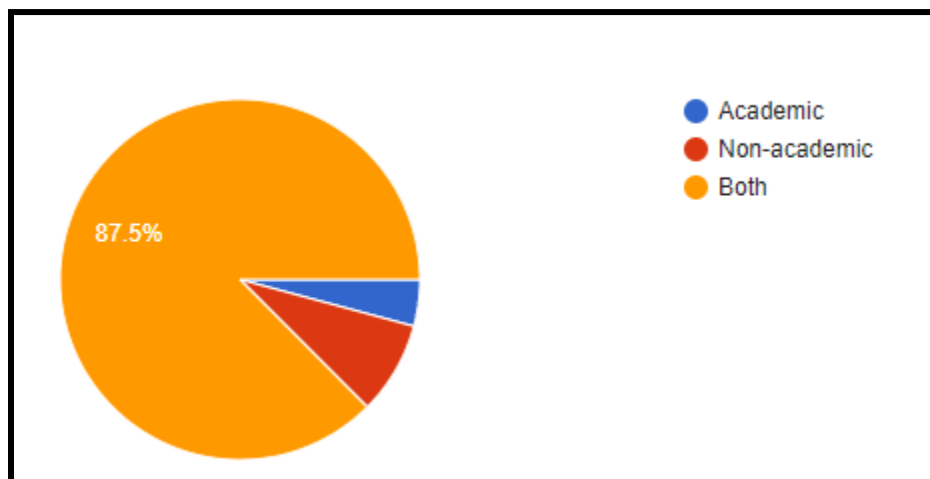
Would you like reward system?



Which type of reward you would like?



For which kind of activities you would like to get reward?



The combined Requirements gathered from Response/s.

- Many colleges haven't developed a student reward system yet and most students want a student reward system , so they should develop this system.
- Some colleges give rewards only on academic achievements but mostly students want rewards on both non-academic and academic basis, so they had to go with both.
- Some colleges give rewards to less than 5 students. They had to increase the number of students who get rewarded to motivate more students.
- As a reward some colleges give books,trophies and other things but mostly students want scholarships as a reward, so colleges had to give scholarships as a reward to motivate students for further studies.

(iv) Observations:-

System : Student Reward System

Project Reference : SF/SJ/2003/12

Observations by: Om Limbachiya (Database Developer)

Date: 30/9/2022

Time: 11:30

Duration: 45 minutes

Place: HOR men

Observations:

- Sometimes points of one student get into another student's result (updating issue).
- Exams had to contain more weightage than any other academic activities.
- Non-academic and academic activities points had to maintain an approx ratio of 3:7 to motivate students for study.
- At least 5 students had to be rewarded.
- Reward points had to be updated from time to time.
- After announcing the result students are not getting reward on time.
- Most students like scholarships as a reward.

(3)Fact Finding Chart:-

Objective	Technique	Subject	Time Commitment
To get background knowledge on the Student reward system	Background Readings	Few similar Projects and requirements of Student rewards	1 day
Preliminary meeting to	Interview	Student(DAIICT)	45 minutes

identify problems and requirements regarding updating reward points on a regular basis			
Any other problems or suggestion regarding reward management system	Interview	TA(DAIICT)	45 minutes
To understand the student's perspective	Observation	Student	1 hour

(4) List Requirements

- For the progress of students, the reward points will be checked on a regular basis.
- Give more preference to exams rather than quizzes and labs.
- Many colleges haven't developed a student reward system yet and most students want a student reward system , so they should develop this system.
- Some colleges give rewards only on academic achievements but mostly students want rewards on both non-academic and academic basis, so they had to go with both.

- Some colleges give rewards to less than 5 students. They had to increase the number of students who get rewarded to motivate more students.
- As a reward some colleges give books, trophies and other things but mostly students want scholarships as a reward, so colleges had to give scholarships as a reward to motivate students for further studies.
- Non-academic and academic activities points had to maintain an approx ratio of 3:7 to motivate students for study.

(5) User Categories and Privileges:-

- **Director**:- They are the admin of the database. They have all the rights and can also add/remove Dean.
- **Dean**:- These will be the main users. They can add/remove students, professors and update any information of the students, etc. In short, they are the manager of the database.
- **Professor**:- They can view every student's result and reward points. They can edit only their own subject's result/point and add/remove TA.
- **TA**:- They can edit the result/point like lab point, quiz point, etc. Which subjects are assigned to them.
- **Student**:- They can view their own academic and non-academic reward points and results.

(6) Assumptions

- Every student has a smartphone/laptop to check their updated points.
- Every student has to participate in reward system.
- Reward event should be done in every semester.
- Point of each student for every event has to be count fairly.

(7) Business Constraints

- Every student's id must be unique and valid.
- Some students don't have any smartphone/laptop so, they face problem to check their updated points.
- Some students may not want to join reward system.
- Sometimes there can be mistake in counting points of students.

Final ERD

Description:-

Requirements:-

- ❖ Our system will keep track of student's basic details, result, subject details ,academic and non-academic activities and total points that students earn,etc.
- ❖ At the end of every semester five students will get a reward which will be given on the basis of academic and non-academic activities points.

The user can get the following details about the given system :

1. Student name
2. Student id
3. Student's branch
4. Academic year
5. Subject name
6. Subject id

7. Professor name
8. Academic Id
9. Quiz point
10. Exam point
11. Bonus point
12. Lab point
13. Subject point
14. Total academic point
15. Non-Academic Id
16. Physical activity point
17. Responsibility point
18. Total non-academic point
19. Total reward point
20. Reward id

This system contains following tables:

- ❖ Student :- It contains basic details of a student like student name, student id, student's branch , academic year.
- ❖ Subject:- It contains details of the subject like subject name, subject id, professor name (who takes that course).
- ❖ Sub points:- All the subject has subject points. It contains details of total academic point which contains student name, Ac id, subject name , subject id, quiz point, exam point, bonus point, lab point and subject point (total points of a subject)
- ❖ Academic:- It contains details of academic points earned by students which contains student name, Ac id, total academic points. It depends on sub points.
- ❖ Non-academic points :- It contains details of total non-academic points which contains student name, NAc id, physical activity point, responsibility point and total non-academic points.
- ❖ Non-academic:- It contains details of non-academic points earned by students which contains student name, NAc id , total non-academic points. It constructed on the basis of non-academic points.
- ❖ Reward:- By combining total academic points and total non-academic points and adding them we get total reward points. It contains student

name, student id, Reward id, total academic point, total non-academic points, total reward points.

Noun Analysis :-

Sr No.	Nouns	Verbs
1	User	Can get
2	Student	contains
3	name	
4	System	Will given
5	Student Id	
6	Student name	
7	Student's branch	
8	Academic year	
9	Subject name	
10	Subject id	
11	Professor name	
12	Point	
13	Exam point	
14	Quiz point	
15	Subject point	
16	Bonus point	
17	Total academic point	
18	Physical activity point	

19	Responsibility point	
20	Total non-academic point	constructed
21	Total reward point	
22	Reward point	add
23	academic	depends
24	non-academic	earned
25	detail	
26	basic	
27	year	
28	course	takes
29	total	
30	Sub points	has
31	Lab points	
32	subject	like
33	reward	earned
34	can	
35	about	
36	Physical	
37	responsibility	
38	five	
39	students	Will get
40	system	Keep track
41	Ac Id	
42	NAc Id	

43	Reward Id	combine
----	-----------	---------

2.Entity-Attribute:-

Sr no.	Candidate entity set	Candidate attribute set	Candidate relationship set
1.	Student	Student name, <u>Student Id</u> ,Student's Branch,Academic year	takes,earned,given
2.	Subject	Subject name, <u>Subject Id</u> ,Professor name	takes,has
3.	Sub points	Student name, <u>Ac Id</u> ,Subject name, <u>Subject Id</u> ,quiz point,exam point,bonus point,lab point,subject point	earned,has,depends
4.	Academic	Student name, <u>Ac id</u> ,total academic point	add,depends,combine
5.	Non-Academic points	Student name, <u>NAC Id</u> ,physical activity point,responsibility point,total non-academic point	earned,construct
6.	Non-Academic	Student name, <u>NAC Id</u> ,total non-academic point	add,construct,combine

7.	Reward	Student name, Student Id, <u>Reward Id</u> , total non-academic point, total academic point, total reward point	given, add, combine
----	--------	---	---------------------

Rejected nouns:-

Noun	Rejection reason
User	General
name	Duplicates
System	General
Student Id	Attributes
Student name	Attributes
Student's branch	Attributes
Academic year	Attributes
Subject name	Attributes
Subject id	Attributes
Professor name	Attributes
Point	Duplicates
Exam point	Attributes
Quiz point	Attributes
Bonus point	Attributes
Total academic point	Attributes
Physical activity point	Attributes

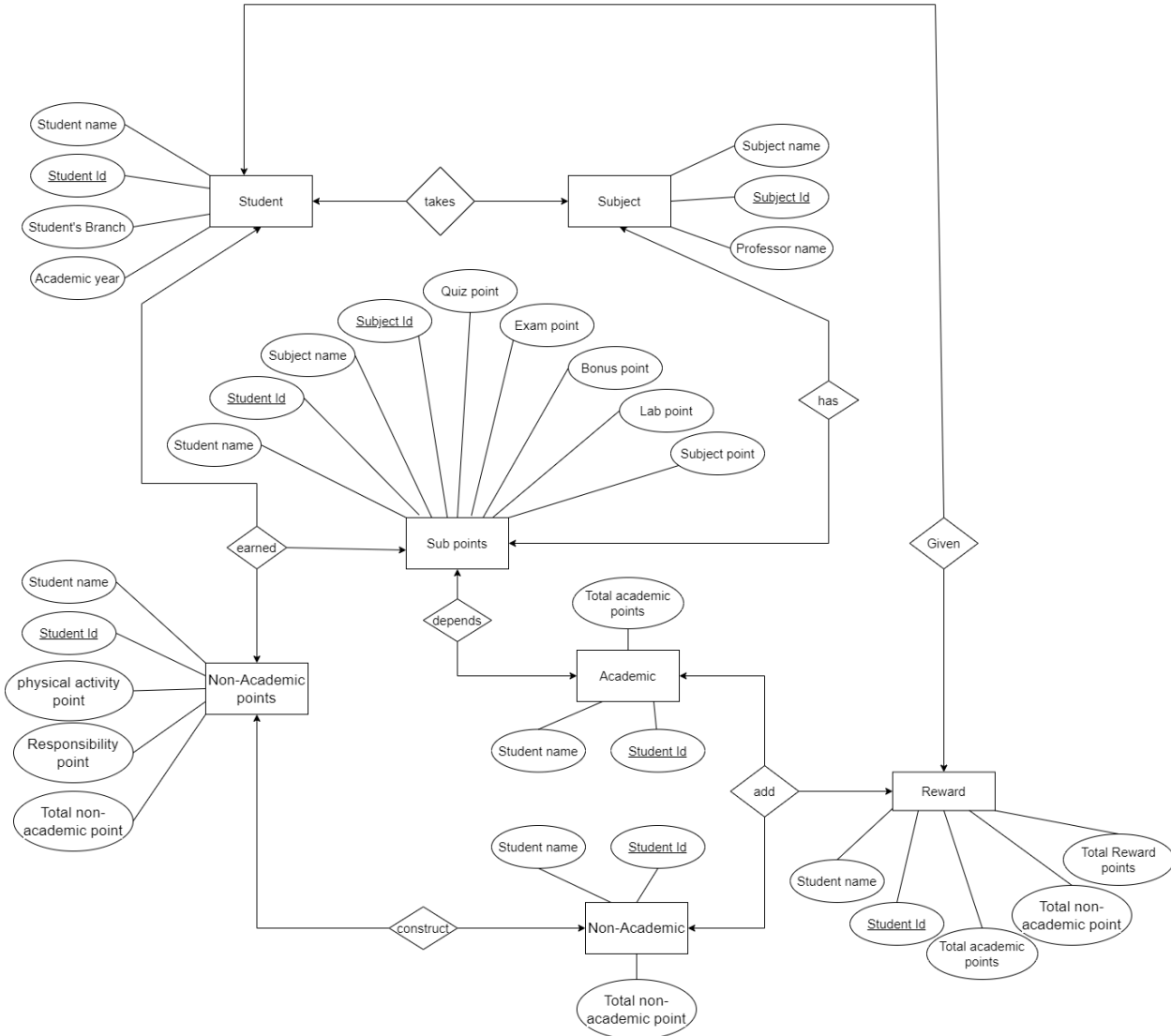
Responsibility point	Attributes
Total non-academic point	Attributes
Total reward point	Attributes
Reward point	Duplicates
details	General
basic	General
year	Duplicates
course	Duplicates
total	Duplicates
Subject point	Attributes
Lab point	Attributes
can	General
about	General
physical	General
responsibility	General
five	General
students	Duplicate
system	Duplicate

Rejected verbs:-

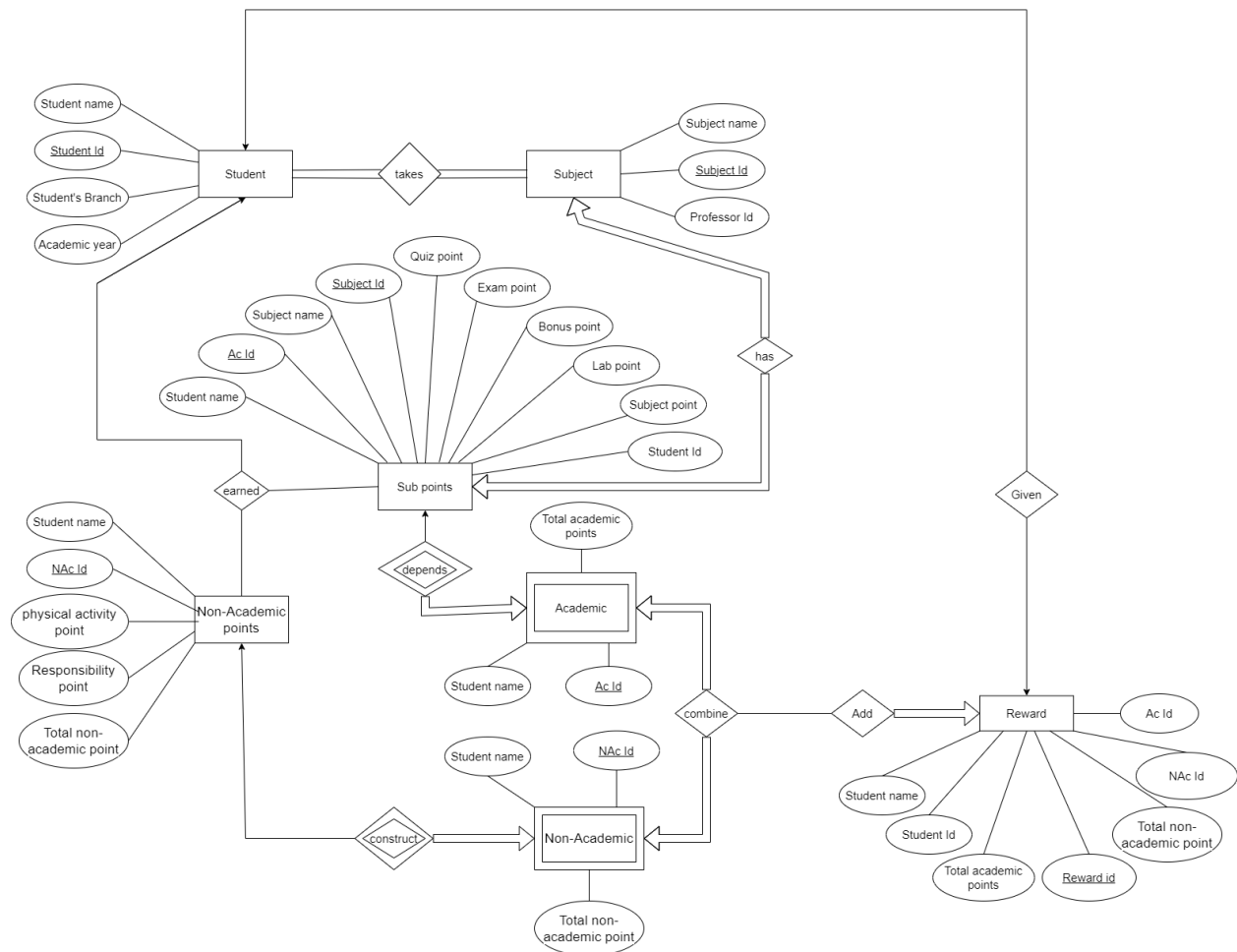
Verbs	Rejection reason
can	General
contains	Duplicates
like	General

keep	Duplicates
track	Irrelevant
be	General

ER-Diagram Version 1:-



ERD-Version-2:-



ERD To Relational Model,DDL

★ Mapping E-R Model to Relational Model

- 1.Student(Student name, Student Id, Student's branch, Academic year)
Primary Key:- Student Id
- 2.Subject(Subject name, Subject Id, Professor name, Professor Id)
Primary Key:- Subject Id

3.Takes(Subject Id, Student Id)

Primary Key:- Subject Id, Student Id

Foreign Key:- Subject Id, Student Id

4.Sub points(Student name, Student Id, Ac Id, Subject Id, Subject name
Quiz point, Exam point, Bonus point, Lab point, Subject point)

Primary Key:- Ac Id, Subject Id

Foreign Key:- Subject Id, Student Id

5.Non-academic point(Student name, Student Id, NAC Id, physical activity
point, Responsibility point, Total non-academic points)

Primary Key:- NAc Id

Foreign Key:- Student Id

6.Academic(Student name, Ac Id, Total academic points)

Primary Key:- Ac Id

7.Non-Academic(Student name, NAC Id, Total non-academic points)

Primary Key:- NAc Id

Foreign Key:- NAc Id

8.Reward(Student name, Student Id, Ac Id, NAc Id, Reward Id, Total
non-academic points, Total academic Points, Total Reward points)

Primary Key:- Reward Id

Foreign Key:- Reward Id, NAc Id, Ac Id

★ DDL Script

Student table:-

```
CREATE TABLE public."Student"
```

```
(  
    "Student Id" bigint NOT NULL,  
    "Student name" character varying NOT NULL,  
    "Student's branch" character varying NOT NULL,  
    "Academic year" integer NOT NULL,  
    PRIMARY KEY ("Student Id")  
);
```

```
ALTER TABLE IF EXISTS public."Student"  
    OWNER to postgres;
```

Subject table:-

```
CREATE TABLE public."Subject"  
(  
    "Subject Id" character varying NOT NULL,  
    "Subject name" character varying NOT NULL,  
    "Professor name" character varying NOT NULL,  
    PRIMARY KEY ("Subject Id")  
);
```

```
ALTER TABLE IF EXISTS public."Subject"  
    OWNER to postgres;
```

Takes table:-

```
CREATE TABLE public."Takes"  
(  
    "Student Id" bigint NOT NULL,  
    "Subject Id" character varying NOT NULL,  
    PRIMARY KEY ("Student Id", "Subject Id"),  
    CONSTRAINT "Student Id" FOREIGN KEY ("Student Id")  
        REFERENCES public."Student" ("Student Id") MATCH SIMPLE
```

```

        ON UPDATE CASCADE
        ON DELETE CASCADE
        NOT VALID,
    CONSTRAINT "Subject Id" FOREIGN KEY ("Subject Id")
        REFERENCES public."Subject" ("Subject Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
        NOT VALID
);

ALTER TABLE IF EXISTS public."Takes"
    OWNER to postgres;

```

Sub point table:-

```

CREATE TABLE IF NOT EXISTS public."Sub points"
(
    "Student Id" bigint NOT NULL,
    "Student name" character varying COLLATE pg_catalog."default" NOT
    NULL,
    "Ac Id" bigint NOT NULL,
    "Subject Id" character varying COLLATE pg_catalog."default" NOT
    NULL,
    "Subject name" character varying COLLATE pg_catalog."default" NOT
    NULL,
    "Quiz point" integer NOT NULL DEFAULT 0,
    "Exam point" integer NOT NULL DEFAULT 0,
    "Bonus point" integer NOT NULL DEFAULT 0,
    "Lab point" integer NOT NULL DEFAULT 0,
    "Subject point" integer NOT NULL DEFAULT 0,
    CONSTRAINT "Sub points_pkey" PRIMARY KEY ("Ac Id", "Subject Id"),
    CONSTRAINT "Student Id" FOREIGN KEY ("Student Id")
        REFERENCES public."Student" ("Student Id") MATCH SIMPLE
        ON UPDATE CASCADE

```



```
        ON DELETE CASCADE,  
CONSTRAINT "Subject Id" FOREIGN KEY ("Subject Id")  
    REFERENCES public."Subject" ("Subject Id") MATCH SIMPLE  
    ON UPDATE CASCADE  
    ON DELETE CASCADE  
)
```

```
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS public."Sub points"  
    OWNER to postgres;
```

Non-academic point table :-

```
CREATE TABLE IF NOT EXISTS public."Non-academic point"  
(  
    "Student name" character varying COLLATE pg_catalog."default" NOT  
NULL,  
    "Student Id" bigint NOT NULL,  
    "NAc Id" bigint NOT NULL,  
    "Physical activity point" integer NOT NULL DEFAULT 0,  
    "Responsibility point" integer NOT NULL DEFAULT 0,  
    "Total non-academic point" integer NOT NULL DEFAULT 0,  
    CONSTRAINT "Non-academic point_pkey" PRIMARY KEY ("NAc Id"),  
    CONSTRAINT "Student Id" FOREIGN KEY ("Student Id")  
        REFERENCES public."Student" ("Student Id") MATCH SIMPLE  
        ON UPDATE CASCADE  
        ON DELETE CASCADE  
)
```

```
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS public."Non-academic point"  
    OWNER to postgres;
```

Academic table:-

```
CREATE TABLE public."Academic"  
(  
    "Student name" character varying NOT NULL,  
    "Ac Id" bigint NOT NULL,  
    "Total academic point" integer NOT NULL DEFAULT 0,  
    PRIMARY KEY ("Ac Id")  
);
```

```
ALTER TABLE IF EXISTS public."Academic"  
    OWNER to postgres;
```

Non-academic table:-

```
CREATE TABLE public."Non Academic"  
(  
    "Student name" character varying NOT NULL,  
    "NAc Id" bigint NOT NULL,  
    "Total non-academic point" integer NOT NULL DEFAULT 0,  
    PRIMARY KEY ("NAc Id"),  
    CONSTRAINT "NAc Id" FOREIGN KEY ("NAc Id")  
        REFERENCES public."Non-academic point" ("NAc Id") MATCH  
SIMPLE  
        ON UPDATE CASCADE  
        ON DELETE CASCADE  
        NOT VALID  
);
```

```
ALTER TABLE IF EXISTS public."Non Academic"  
    OWNER to postgres;
```

Reward table:-

-- Table: public.Reward

-- DROP TABLE IF EXISTS public."Reward";

CREATE TABLE IF NOT EXISTS public."Reward"

(

 "Student name" character varying COLLATE pg_catalog."default" NOT NULL,

 "Student Id" bigint NOT NULL,

 "Ac Id" bigint NOT NULL,

 "NAc Id" bigint NOT NULL,

 "Reward Id" bigint NOT NULL,

 "Total academic point" integer NOT NULL DEFAULT 0,

 "Total non-academic point" integer NOT NULL DEFAULT 0,

 "Total Reward point" integer NOT NULL DEFAULT 0,

 CONSTRAINT "Reward_pkey" PRIMARY KEY ("Reward Id"),

 CONSTRAINT "Reward_Ac Id_fkey" FOREIGN KEY ("Ac Id")

 REFERENCES public."Academic" ("Ac Id") MATCH SIMPLE

 ON UPDATE CASCADE

 ON DELETE CASCADE,

 CONSTRAINT "Reward_NAc Id_fkey" FOREIGN KEY ("NAc Id")

 REFERENCES public."Non Academic" ("NAc Id") MATCH SIMPLE

 ON UPDATE CASCADE

 ON DELETE CASCADE,

 CONSTRAINT "Reward_Student Id_fkey" FOREIGN KEY ("Student Id")

 REFERENCES public."Student" ("Student Id") MATCH SIMPLE

 ON UPDATE CASCADE

 ON DELETE CASCADE

)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Reward"
OWNER to postgres;

Normalization and Schema Refinement

★ Functional Dependencies

1. **Student**(**Student Id** -> Student name, Student's branch, Academic year)
2. **Subject**(**Subject Id** -> Subject name, Professor name, Professor Id ,
Professor Id -> Professor name)
3. **Takes** : No dependency
4. **Sub points**((**Ac Id**, **Subject Id**) -> Student name, Student Id, Subject name, Quiz point, Exam point, Bonus point, Lab point, Subject point ,
(**Quiz point**, **Exam point**, **Bonus point**, **Lab point**) -> Subject point)
5. **Non-academic point**(**NAc Id** -> Student name, Student Id, physical activity point, Responsibility point, Total non-academic points ,
(**physical activity point**, **Responsibility point**) -> Total non-academic points)
6. **Academic**(**Ac Id** -> Student name, Total academic points)
7. **Non-Academic**(**NAc Id** -> Student name, Total non-academic points)
8. **Reward**(**Reward Id** -> Student name, Student Id, Ac Id, NAc Id, Total non-academic points, Total academic Points, Total Reward points ,

(Total non-academic points, Total academic Points) -> Total Reward points)

★ Redundancy and Analysis

- Student(Student name, Student Id, Student's branch, Academic year)
Primary Key:- Student Id
Redundancy :- No redundancy
- Subject(Subject name, Subject Id, Professor name, Professor Id)
Primary Key:- Subject Id
Redundancy :- Transitive Dependencies (Subject Id -> Professor Id , Professor Id -> Professor name implies Subject Id -> Professor name)
-> We removed attribute Professor name. So, there are no transitive dependency.
- Takes(Subject Id, Student Id)
Primary Key:- Subject Id, Student Id
Foreign Key:- Subject Id, Student Id
Redundancy :- No redundancy
- Sub points(Student name, Student Id, Ac Id, Subject Id, Subject name, Quiz point, Exam point, Bonus point, Lab point, Subject point)
Primary Key:- Ac Id, Subject Id
Foreign Key:- Subject Id, Student Id
Redundancy :- Transitive Dependencies ((Ac Id, Subject Id) -> Quiz point, Exam point, Bonus point, Lab point And (Quiz point, Exam point, Bonus point, Lab point) -> Subject Point implies (Ac Id, Subject Id) -> Subject point)
-> We removed attribute Subject point. So, there are no transitive dependency.

- Non-academic point(Student name, Student Id, NAc Id, physical activity point, Responsibility point, Total non-academic points)
 Primary Key:- NAc Id
 Foreign Key:-Student Id
 Redundancy :- Transitive Dependencies (NAc Id -> physical activity point, Responsibility point And (physical activity point, Responsibility point) -> Total non-academic points implies NAc Id -> Total non-academic points)
 -> We removed the attribute Total non-academic points. So, there is no transitive dependency.
- Academic(Student name, Ac Id, Total academic points)
 Primary Key:- Ac Id
 Redundancy :- No redundancy
- Non-Academic(Student name, NAc Id, Total non-academic points)
 Primary Key:- NAc Id
 Foreign Key:- NAc Id
 Redundancy :- No redundancy
- Reward(Student name, Student Id, Ac Id, NAc Id, Reward Id, Total non-academic points, Total academic Points, Total Reward points)
 Primary Key:- Reward Id
 Foreign Key:- Reward Id, NAc Id, Ac Id
 Redundancy :- Transitive Dependencies (Reward Id -> Total non-academic points, Total academic Points And (Total non-academic points, Total academic Points) ->Total Reward points implies Reward Id-> Total Reward points)
 -> We removed attribute Total Reward points. So, there are no transitive dependency.

★ Normalization upto 3NF/BCNF

- Student(Student name, Student Id, Student's branch, Academic year)
This schema does not have any composite or multi-valued attributes.
It satisfies atomicity.
Hence, It is already in 1NF form.
There is only one attribute (Student_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.
- Subject(Subject name, Subject Id, Professor name, Professor Id)
This schema has a multi-valued attribute which is professor name.
It does not satisfy atomicity.
Hence, It is not in 1NF form.
So, we had to make another schema of (Professor name, Professor Id). So it will satisfy 1NF form.
After make changes,
There is only one attribute (Subject_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.
- Takes(Subject Id, Student Id)
This schema does not have any composite or multi-valued attributes.
It satisfies atomicity.
Hence, It is already in 1NF form.
There are two attributes (Subject_Id, Student_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.
- Sub points(Student name, Student Id, Ac Id, Subject Id, Subject name, Quiz point, Exam point, Bonus point, Lab point, Subject point)

This schema does not have any composite or multi-valued attributes. It satisfies atomicity.

Hence, It is already in 1NF form.

There are two attributes (Ac_Id, Subject_Id) in the candidate key.

There is no partial dependency in this table , hence it is in 2NF form.

There is transitive dependency in this schema. So, we removed the Subject point from this schema.

After making changes,

all the functional dependencies have candidate keys on the left side.

So, it is in BCNF form which implies it is 3NF form.

- Non-academic point(Student name, Student Id, NAC Id, physical activity point, Responsibility point, Total non-academic points)
This schema does not have any composite or multi-valued attributes. It satisfies atomicity.
Hence, It is already in 1NF form.
There is only one attribute (NAC_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
There is transitive dependency in this schema. So, we removed the Total non-academic points from this schema.
After making changes,
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.
- Academic(Student name, Ac Id, Total academic points)
This schema does not have any composite or multi-valued attributes. It satisfies atomicity.
Hence, It is already in 1NF form.
There is only one attribute (Ac_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.

- Non-Academic(Student name, NAc Id, Total non-academic points)
This schema does not have any composite or multi-valued attributes.
It satisfies atomicity.
Hence, It is already in 1NF form.
There is only one attribute (NAc_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.
- Reward(Student name, Student Id, Ac Id, NAc Id, Reward Id, Total non-academic points, Total academic Points, Total Reward points)
This schema does not have any composite or multi-valued attributes.
It satisfies atomicity.
Hence, It is already in 1NF form.
There is only one attribute (Reward_Id) in the candidate key.
There is no partial dependency in this table , hence it is in 2NF form.
There is transitive dependency in this schema. So, we removed the Total Reward points from this schema.
After making changes,
All the functional dependencies have candidate keys on the left side.
So, it is in BCNF form which implies it is 3NF form.

★ **List all final Relations & Schema with all details(final design of database)**

- 1.Student(Student name, Student Id, Student's branch, Academic year)
- 2.Subject(Subject name, Subject Id, Professor Id)
- 3.Takes(Subject Id, Student Id)

4.Professor(Professor Id, Professor name)

5.Sub points(Student name, Student Id, Ac Id, Subject Id, Subject name
Quiz point, Exam point, Bonus point, Lab point, Subject point)

6.Non-academic point(Student name, Student Id, NAc Id, physical activity
point, Responsibility point)

7.Academic(Student name, Ac Id, Total academic points)

8.Non-Academic(Student name, NAc Id, Total non-academic points)

9.Reward(Student name, Student Id, Ac Id, NAc Id, Reward Id, Total
non-academic points, Total academic Points)

10. Reward point (Reward Id, Total Reward points)

★ DDL script

Student:-

-- Table: public.Student

-- DROP TABLE IF EXISTS public."Student";

CREATE TABLE IF NOT EXISTS public."Student"

(

 "Student Id" bigint NOT NULL,

 "Student name" character varying COLLATE pg_catalog."default" NOT NULL,

 "Student's branch" character varying COLLATE pg_catalog."default" NOT
NULL,

 "Academic year" integer NOT NULL,

 CONSTRAINT "Student_pkey" PRIMARY KEY ("Student Id")

)

```
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS public."Student"  
  OWNER to postgres;
```

Subject:-

```
-- Table: public.Subject
```

```
-- DROP TABLE IF EXISTS public."Subject";
```

```
CREATE TABLE IF NOT EXISTS public."Subject"  
(  
  "Subject Id" character varying COLLATE pg_catalog."default" NOT NULL,  
  "Subject name" character varying COLLATE pg_catalog."default" NOT NULL,  
  "Professor Id" character varying COLLATE pg_catalog."default" NOT NULL,  
  CONSTRAINT "Subject_pkey" PRIMARY KEY ("Subject Id"),  
  CONSTRAINT "Subject_Professor Id_fkey" FOREIGN KEY ("Professor Id")  
    REFERENCES public."Professor" ("Professor Id ") MATCH SIMPLE  
    ON UPDATE CASCADE  
    ON DELETE CASCADE  
  NOT VALID  
)
```

```
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS public."Subject"  
  OWNER to postgres;
```

Takes:-

```
-- Table: public.Takes
```

```
-- DROP TABLE IF EXISTS public."Takes";
```

```

CREATE TABLE IF NOT EXISTS public."Takes"
(
    "Student Id" bigint NOT NULL,
    "Subject Id" character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Takes_pkey" PRIMARY KEY ("Student Id", "Subject Id"),
    CONSTRAINT "Student Id" FOREIGN KEY ("Student Id")
        REFERENCES public."Student" ("Student Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT "Subject Id" FOREIGN KEY ("Subject Id")
        REFERENCES public."Subject" ("Subject Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

```

```

TABLESPACE pg_default;

```

```

ALTER TABLE IF EXISTS public."Takes"
    OWNER to postgres;

```

Professor :-

```

-- Table: public.Professor

```

```

-- DROP TABLE IF EXISTS public."Professor";

```

```

CREATE TABLE IF NOT EXISTS public."Professor"
(
    "Professor Id " character varying COLLATE pg_catalog."default" NOT NULL,
    "Professor name" character varying COLLATE pg_catalog."default" NOT
    NULL,
    CONSTRAINT "Professor_pkey" PRIMARY KEY ("Professor Id ")
)

```

```

TABLESPACE pg_default;

```

```
ALTER TABLE IF EXISTS public."Professor"  
OWNER to postgres;
```

Sub points:-

```
-- Table: public.Sub points
```

```
-- DROP TABLE IF EXISTS public."Sub points";
```

```
CREATE TABLE IF NOT EXISTS public."Sub points"  
(  
    "Student Id" bigint NOT NULL,  
    "Student name" character varying COLLATE pg_catalog."default" NOT NULL,  
    "Ac Id" bigint NOT NULL,  
    "Subject Id" character varying COLLATE pg_catalog."default" NOT NULL,  
    "Subject name" character varying COLLATE pg_catalog."default" NOT NULL,  
    "Quiz point" integer NOT NULL DEFAULT 0,  
    "Exam point" integer NOT NULL DEFAULT 0,  
    "Bonus point" integer NOT NULL DEFAULT 0,  
    "Lab point" integer NOT NULL DEFAULT 0,  
    CONSTRAINT "Sub points_pkey" PRIMARY KEY ("Ac Id", "Subject Id"),  
    CONSTRAINT "Student Id" FOREIGN KEY ("Student Id")  
        REFERENCES public."Student" ("Student Id") MATCH SIMPLE  
        ON UPDATE CASCADE  
        ON DELETE CASCADE,  
    CONSTRAINT "Subject Id" FOREIGN KEY ("Subject Id")  
        REFERENCES public."Subject" ("Subject Id") MATCH SIMPLE  
        ON UPDATE CASCADE  
        ON DELETE CASCADE  
)
```

```
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS public."Sub points"  
OWNER to postgres;
```

Non-academic point:-

-- Table: public.Non-academic point

-- DROP TABLE IF EXISTS public."Non-academic point";

CREATE TABLE IF NOT EXISTS public."Non-academic point"

(
 "Student name" character varying COLLATE pg_catalog."default" NOT NULL,
 "Student Id" bigint NOT NULL,
 "NAc Id" bigint NOT NULL,
 "Physical activity point" integer NOT NULL DEFAULT 0,
 "Responsibility point" integer NOT NULL DEFAULT 0,
 CONSTRAINT "Non-academic point_pkey" PRIMARY KEY ("NAc Id"),
 CONSTRAINT "Student Id" FOREIGN KEY ("Student Id")
 REFERENCES public."Student" ("Student Id") MATCH SIMPLE
 ON UPDATE CASCADE
 ON DELETE CASCADE
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Non-academic point"
 OWNER to postgres;

Academic:-

-- Table: public.Academic

-- DROP TABLE IF EXISTS public."Academic";

CREATE TABLE IF NOT EXISTS public."Academic"

(
 "Student name" character varying COLLATE pg_catalog."default" NOT NULL,
 "Ac Id" bigint NOT NULL,
 "Total academic point" integer NOT NULL DEFAULT 0,
 CONSTRAINT "Academic_pkey" PRIMARY KEY ("Ac Id")

)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Academic"
OWNER to postgres;

Non-academic:-

-- Table: public.Non Academic

-- DROP TABLE IF EXISTS public."Non Academic";

CREATE TABLE IF NOT EXISTS public."Non Academic"

(

"Student name" character varying COLLATE pg_catalog."default" NOT NULL,

"NAc Id" bigint NOT NULL,

"Total non-academic point" integer NOT NULL DEFAULT 0,

CONSTRAINT "Non Academic_pkey" PRIMARY KEY ("NAc Id"),

CONSTRAINT "NAc Id" FOREIGN KEY ("NAc Id")

REFERENCES public."Non-academic point" ("NAc Id") MATCH SIMPLE

ON UPDATE CASCADE

ON DELETE CASCADE

)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Non Academic"
OWNER to postgres;

Reward:-

-- Table: public.Reward

-- DROP TABLE IF EXISTS public."Reward";

```

CREATE TABLE IF NOT EXISTS public."Reward"
(
    "Student name" character varying COLLATE pg_catalog."default" NOT NULL,
    "Student Id" bigint NOT NULL,
    "Ac Id" bigint NOT NULL,
    "NAc Id" bigint NOT NULL,
    "Reward Id" bigint NOT NULL,
    "Total academic point" integer NOT NULL DEFAULT 0,
    "Total non-academic point" integer NOT NULL DEFAULT 0,
    CONSTRAINT "Reward_pkey" PRIMARY KEY ("Reward Id"),
    CONSTRAINT "Reward_Ac Id_fkey" FOREIGN KEY ("Ac Id")
        REFERENCES public."Academic" ("Ac Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT "Reward_NAc Id_fkey" FOREIGN KEY ("NAc Id")
        REFERENCES public."Non Academic" ("NAc Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT "Reward_Student Id_fkey" FOREIGN KEY ("Student Id")
        REFERENCES public."Student" ("Student Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

```

```

TABLESPACE pg_default;

```

```

ALTER TABLE IF EXISTS public."Reward"
    OWNER to postgres;

```

Reward point:-

```

-- Table: public.Reward Point

```

```

-- DROP TABLE IF EXISTS public."Reward Point";

```



```
CREATE TABLE IF NOT EXISTS public."Reward Point"
(
    "Reward Id" bigint NOT NULL,
    "Total Reward Points" bigint NOT NULL DEFAULT 0,
    CONSTRAINT "Reward Point_pkey" PRIMARY KEY ("Reward Id"),
    CONSTRAINT "Reward Point_Reward Id_fkey" FOREIGN KEY ("Reward Id")
        REFERENCES public."Reward" ("Reward Id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)
```

```
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS public."Reward Point"
    OWNER to postgres;
```

★ Snapshots of table

1.Student(Student name, Student Id, Student's branch, Academic year)

Query	Query History	Messages	Data output	Notifications
1	<code>select * from public."Student";</code>			
	Student Id [PK] bigint	Student name character varying	Student's branch character varying	Academic year integer
1	1	Lexi Walsh	Accounting	2018
2	2	Peyton Brock	Health	2019
3	3	Tyler Parker	Trigonometry	2021
4	4	Paula Hopkinson	Ancient Civilizati...	2021
5	5	Luke Miller	Accounting	2019
6	6	Clint Cowan	Latin	2019
7	7	Danny Ainsworth	Sociology	2021
8	8	Brad Bailey	Physical Education	2017
9	9	Aeris Hunt	French	2019
10	10	Piper Thorne	Resource Program	2017
11	11	Chris Marshall	American Literat...	2021
12	12	Stacy Locke	American Literat...	2017
13	13	Benjamin Grant	Sociology	2019
14	14	Mike Hammond	Grammar	2021
15	15	Brad Dann	Basic Math	2017
16	16	Penelope Redden	History	2020
17	17	Enoch Evans	Grammar	2018

Total rows: 45 of 45 Query complete 00:00:17.742

2. Professor(Professor Id, Professor name)

Query

Query History

1

select * from public."Professor";

Messages

Data output

Notifications

	Professor Id [PK] character varying	Professor name character varying
1	1	Enoch Sloan
2	2	Daniel Jeffery
3	3	Olivia Huggins
4	4	Janelle Richards
5	5	Adeline James
6	6	Alexander Healy
7	7	Gabriel Rossi
8	8	Raquel King
9	9	Carmella Purvis
10	10	Candice Lee
11	11	Christine Rodwell
12	12	Mayleen Mccall
13	13	Joseph Rose
14	14	Jacob Allwood
15	15	Mayleen Whittle
16	16	Andie Irwin
17	17	Jack Huggins

3. Subject(Subject name, Subject Id, Professor Id)

Query

Query History

1

select * from public."Subject";

Messages

Data output

Notifications

	Subject Id [PK] character varying	Subject name character varying	Professor Id character varying
1	1	Resource Program	37
2	2	Economics	12
3	3	German	13
4	4	American Literat...	18
5	5	Instrumental Mu...	39
6	6	Modern Literature	4
7	7	American Literat...	24
8	8	Design and techn...	10
9	9	Spanish	4
10	10	Modern Literature	10
11	11	Language Arts	17
12	12	Basic Math	25
13	13	Modern Literature	38
14	14	Leadership	11
15	15	Latin	5
16	16	Art	19
17	17	Design and techn...	21

4. Takes(Subject Id, Student Id)

Query	Query History	Messages	Data output	Notifications
1	<code>select * from public."Takes";</code>			
	Student Id [PK] bigint	Subject Id [PK] character varying		
1	41	19		
2	12	19		
3	16	5		
4	5	11		
5	38	12		
6	22	18		
7	16	13		
8	27	1		
9	33	9		
10	7	15		
11	38	4		
12	12	1		
13	34	4		
14	14	14		
15	5	3		
16	13	6		
17	37	3		

5. Sub points(Student name, Student Id, Ac Id, Subject Id, Subject name, Quiz point, Exam point, Bonus point, Lab point, Subject point)

Query

Query History

1

select * from public."Sub points";

Messages

Data output

Notifications

	Student Id bigint	Student name character varying	Ac Id [PK] bigint	Subject Id [PK] character varying	Subject name character varying	Quiz point integer	Exam point integer	Bonus point integer	Lab point integer
1	6	Elly Wright	1	5	Latin	1	5	12	15
2	8	Sofia Wooldridge	2	4	Music	2	22	19	3
3	16	Jacob Fox	3	5	Grammar	14	27	10	14
4	25	Ember Kerr	4	16	Modern Literature	4	24	17	15
5	21	Lily Overson	5	10	Economics	17	8	19	9
6	31	Paige Hastings	6	10	Instrumental Mu...	15	10	17	10
7	16	Willow Mould	7	15	Mathematics	3	34	3	3
8	21	Caydence Everett	8	8	Resource Program	12	38	2	15
9	22	Georgia Niles	9	7	Music	4	39	8	1
10	41	Julian Knott	10	10	Ancient Civilizati...	17	34	13	3
11	44	Miley Hobson	11	7	Spanish	1	6	13	13
12	35	Hank Owens	12	11	Language Arts	13	21	11	10
13	7	Peter Dowson	13	16	Sociology	15	12	15	18
14	32	Daron Brooks	14	8	Design and techn...	8	33	15	9

6.Non-academic point(Student name, Student Id, NAc Id, physical activity point, Responsibility point)

Query

Query History








1

select * from public."Non-academic point";

Messages

Data output

Notifications



	Student name character varying	Student Id bigint	NAc Id [PK] bigint	Physical activity point integer	Responsibility point integer
1	Alexander Victor	36	1	7	15
2	Abdul Stanton	25	2	24	8
3	Ember Overson	2	3	15	20
4	Rowan Robe	19	4	4	5
5	Ryan Clarkson	17	5	4	23
6	Jayden Wood	13	6	16	23
7	Regina Nash	33	7	17	21
8	Lillian Adams	19	8	0	4
9	Rufus Wright	16	9	12	17
10	Hayden Lane	2	10	21	18
11	Ruby Styles	30	11	19	16
12	Tony Gray	21	12	22	14
13	Julius Evans	7	13	20	18
14	Bob Robinson	13	14	1	17

7.Academic(Student name, Ac Id, Total academic points)

Query

Query History

1

select * from public."Academic";

Messages

Data output

Notifications

	Student name character varying	Ac Id [PK] bigint	Total academic point integer
1	Sienna Waterson	1	82
2	Renee Dowson	2	41
3	Jolene Upsdell	3	35
4	Michael Addis	4	84
5	Elisabeth King	5	5
6	Cameron Hamilton	6	76
7	Benjamin Wooldr...	7	15
8	William Jarrett	8	42
9	Madelyn Denton	9	98
10	Bryce Rosenbloom	10	31
11	Gil Morrow	11	5
12	Tyson Alldridge	12	7
13	Natalie Nayler	13	26
14	Kurt Nanton	14	76

Total rows: 45 of 45

Query complete 00:00:00.074

8.Non-Academic(Student name, NAc Id, Total non-academic points)

Query		Query History	
1		<code>select * from public."Non Academic";</code>	
Messages		Data output	
Notifications			
	Student name character varying	NAc Id [PK] bigint	Total non-academic point integer
1	Lucy Edler	1	43
2	Julius Rothwell	2	4
3	Fred Archer	3	14
4	Anabelle Spencer	4	21
5	Alice Moran	5	17
6	Kendra Morris	6	11
7	Elijah Neal	7	37
8	Vera Ramsey	8	3
9	Sharon Richardson	9	18
10	Gil Egerton	10	16
11	Harvey Wade	11	9
12	Dakota Brennan	12	15
13	Noah Crawford	13	31
14	Eden Grady	14	20
Total rows: 45 of 45		Query complete 00:00:00.087	

9.Reward(Student name, Student Id, Ac Id, NAc Id, Reward Id, Total non-academic points, Total academic Points)

Query

Query History

1

select * from public."Reward";

Messages

Data output

Notifications

	Student name character varying	Student Id bigint	Ac Id bigint	NAc Id bigint	Reward Id [PK] bigint	Total academic point integer	Total non-academic point integer
1	Crystal Wild	4	9	36	1	95	32
2	Wade Stanley	35	38	35	2	5	33
3	Ron Wright	7	16	28	3	6	21
4	Harvey Skinner	26	36	18	4	22	16
5	Gloria Rixon	25	10	44	5	10	42
6	Anabelle Morris	20	34	15	6	58	27
7	Clint Eddison	5	39	4	7	73	2
8	Jack Lee	17	23	17	8	84	35
9	Johnny Quinton	34	14	22	9	44	4
10	Jessica Allwood	32	21	28	10	66	24
11	Sofia Bullock	13	30	5	11	46	32
12	Faith Simmons	35	33	32	12	33	47
13	Alice Shields	20	13	36	13	24	42
14	Mary Khan	27	12	1	14	58	25









Successfully run. Total query runtime: 71 msec. 45 rows

10. Reward point (Reward Id, Total Reward points)

Query Query History

1 `select * from public."Reward Point";`

Messages Data output Notifications



	Reward Id [PK] bigint	Total Reward Points bigint
1	1	143
2	2	115
3	3	22
4	4	15
5	5	113
6	6	135
7	7	127
8	8	1
9	9	63
10	10	76
11	11	41
12	12	89
13	13	104
14	14	125

★ SQL Queries

1. Display student name with their non academic Id.

-> `select "Student name", "NAc Id" from "Non Academic"`

```
select "Student name", "NAc Id" from "Non Academic"
```

Messages Data output Notifications		
	Student name character varying	NAc Id [PK] bigint
1	Lucy Edler	1
2	Julius Rothwell	2
3	Fred Archer	3
4	Anabelle Spencer	4
5	Alice Moran	5
6	Kendra Morris	6
7	Elijah Neal	7
8	Vera Ramsey	8
9	Sharon Richardson	9
10	Gil Egerton	10
11	Harvey Wade	11

2.Display all student details whose academic year is 2020.

-> select * from "Student" where "Academic year"='2020';

```
1 select * from "Student" where "Academic year"='2020';
```

Messages Data output Notifications				
	Student Id [PK] bigint	Student name character varying	Student's branch character varying	Academic year integer
1	16	Penelope Redden	History	2020
2	18	Javier Ross	Basic Math	2020
3	23	Liam Nash	Mathematics	2020
4	27	Rosalee Murray	Latin	2020
5	29	Nicole Rainford	Basic Math	2020
6	36	Clint Casey	Ancient Civilizati...	2020
7	39	Hannah Woodley	American Literat...	2020
8	40	Percy Brooks	Leadership	2020

3. Display all details of subject whose professor has id 10.

-> select * from "Subject" where "Professor Id"='10';

Query Query History

```
1 select * from "Subject" where "Professor Id"='10';
```

Messages Data output Notifications

	Subject Id [PK] character varying	Subject name character varying	Professor Id character varying
1	8	Design and techn...	10
2	10	Modern Literature	10
3	18	Earth Science	10
4	19	Sociology	10

4. Count the number of students who have taken a subject which has id 4.

-> select count("Student Id") from "Takes" where "Subject Id"='4';

Query Query History

```
1 select count("Student Id") from "Takes" where "Subject Id"='4';
```

Messages Data output Notifications

	count bigint
1	8

5. Display name of the professor who has id 8.

-> select "Professor name" from "Professor" where "Professor Id"='8';


```
1 select "Professor name" from "Professor" where "Professor Id" = '8';
```

Messages Data output Notifications	
<div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> </div>	
	Professor name character varying
1	Raquel King

6. Display the details of students who have more than 20 marks in an exam.

-> select * from "Sub points" where "Exam point" > '20';

Query

Query History

1

select * from "Sub points" where "Exam point" > '20';

Messages

Data output

Notifications

	Student Id bigint	Student name character varying	Ac Id [PK] bigint	Subject Id [PK] character varying	Subject name character varying	Quiz point integer	Exam point integer	Bonus point integer	Lab point integer
1	8	Sofia Wooldridge	2	4	Music	2	22	19	3
2	16	Jacob Fox	3	5	Grammar	14	27	10	14
3	25	Ember Kerr	4	16	Modern Literature	4	24	17	15
4	16	Willow Mould	7	15	Mathematics	3	34	3	3
5	21	Caydence Everett	8	8	Resource Program	12	38	2	15
6	22	Georgia Niles	9	7	Music	4	39	8	1
7	41	Julian Knott	10	10	Ancient Civilizations	17	34	13	3
8	35	Hank Owens	12	11	Language Arts	13	21	11	10
9	32	Daron Brooks	14	8	Design and technolo...	8	33	15	9
10	2	Penny Warden	15	16	Ecology	15	23	16	14

7. Display the details of students who have more than 10 marks in an exam and more than 5 marks in a quiz.

->select * from "Sub points" where "Exam point" > '10' AND "Quiz point" > '5';

Query

Query History

1

select * from "Sub points" where "Exam point" > '10' AND "Quiz point" > '5';

Messages

Data output

Notifications

	Student Id bigint	Student name character varying	Ac Id [PK] bigint	Subject Id [PK] character varying	Subject name character varying	Quiz point integer	Exam point integer	Bonus point integer	Lab point integer
1	16	Jacob Fox	3	5	Grammar	14	27	10	14
2	21	Caydence Everett	8	8	Resource Program	12	38	2	15
3	41	Julian Knott	10	10	Ancient Civilizations	17	34	13	3
4	35	Hank Owens	12	11	Language Arts	13	21	11	10
5	7	Peter Dowson	13	16	Sociology	15	12	15	18
6	32	Daron Brooks	14	8	Design and technolo...	8	33	15	9
7	2	Penny Warden	15	16	Ecology	15	23	16	14
8	16	Nicholas Nobbs	16	19	Sociology	10	39	1	3
9	33	Luke Hunter	17	13	Ecology	10	11	14	13
10	38	Sydney Barrett	18	5	German	16	29	6	6

8. Show Student name and NAc id in descending order of their total non-academic points.

->select * from "Non Academic" order by "Total non-academic point" desc;

Query

Query History









1

select * from "Non Academic" order by "Total non-academic point" desc;

Messages

Data output

Notifications



	Student name character varying	NAc Id [PK] bigint	Total non-academic point integer
1	Alan Rose	39	48
2	Hope Price	33	47
3	Hayden West	30	46
4	Peter Porter	38	46
5	Rick Underhill	31	46
6	Daron Fox	18	45
7	Jack Uttridge	19	44
8	Jazmin Leslie	16	43
9	Lucy Edler	1	43
10	Mason Selby	23	40

9. Show details of the student who has non-academic points greater than 25 and academic points greater than 50.

->select * from "Reward" where "Total non-academic point" > '25' AND "Total academic point" > '50';

Query

Query History

1

select * from "Reward" where "Total non-academic point" > '25' AND "Total academic point" > '50';

Messages

Data output

Notifications

10. Display the reward id of top 5 students who've scored maximum points.

-> select * from "Reward Point" order by "Total Reward Points" desc limit 5;

Query









Query History

```
1 select * from "Reward Point" order by "Total Reward Points" desc
2 limit 5;
```

Messages

Data output

Notifications



	Reward Id [PK] bigint	Total Reward Points bigint
1	26	147
2	32	144
3	1	143
4	30	139
5	27	138

11. Display all the details of students who have scored maximum points in the exam.

->select * from "Sub points"
 where "Exam point"= (select max("Exam point") from "Sub points");

Query

Query History

1

select * from "Sub points"

2

where "Exam point" = (select max("Exam point") from "Sub points");

information schema

Messages

Data output

Notifications

	Student Id bigint	Student name character varying	Ac Id [PK] bigint	Subject Id [PK] character varying	Subject name character varying	Quiz point integer	Exam point integer	Bonus point integer	Lab point integer
1	22	Georgia Niles	9	7	Music	4	39	8	1
2	16	Nicholas Nobbs	16	19	Sociology	10	39	1	3

12. Display all the details of students who got more than 10 points in the quiz.

->select * from "Student"
 where "Student Id" in (select "Student Id" from "Sub points" where "Quiz point">10);

Query

Query History

```
1 select * from "Student"
2 where "Student Id" in (select "Student Id" from "Sub points" where "Quiz point">10);
```

Messages

Data output

Notifications

</

13. Show all the details of subject in which professor gave more than 13 bonus points.

->select * from "Subject"
 where "Subject Id" = some(select "Subject Id" from "Sub points" where "Bonus point">13);

Query Query History

```

1 select * from "Subject"
2 where "Subject Id" = some(select "Subject Id" from "Sub points" where "Bonus point">13);

```

Messages Data output Notifications

Loading...

	Subject Id [PK] character varying	Subject name character varying	Professor Id character varying
1	1	Resource Program	37
2	2	Economics	12
3	3	German	13
4	4	American Literat...	18
5	5	Instrumental Mu...	39
6	6	Modern Literature	4
7	7	American Literat...	24
8	8	Design and techn...	10

14. Show the details of students and subjects who got more than 8 lab points and taught by a professor who has id 10.

->select * from "Subject" natural join "Sub points"
 where "Lab point" > '8' AND "Professor Id" = '10'

Query Query History

```

1 select * from "Subject" natural join "Sub points"
2 where "Lab point" > '8' AND "Professor Id" = '10'

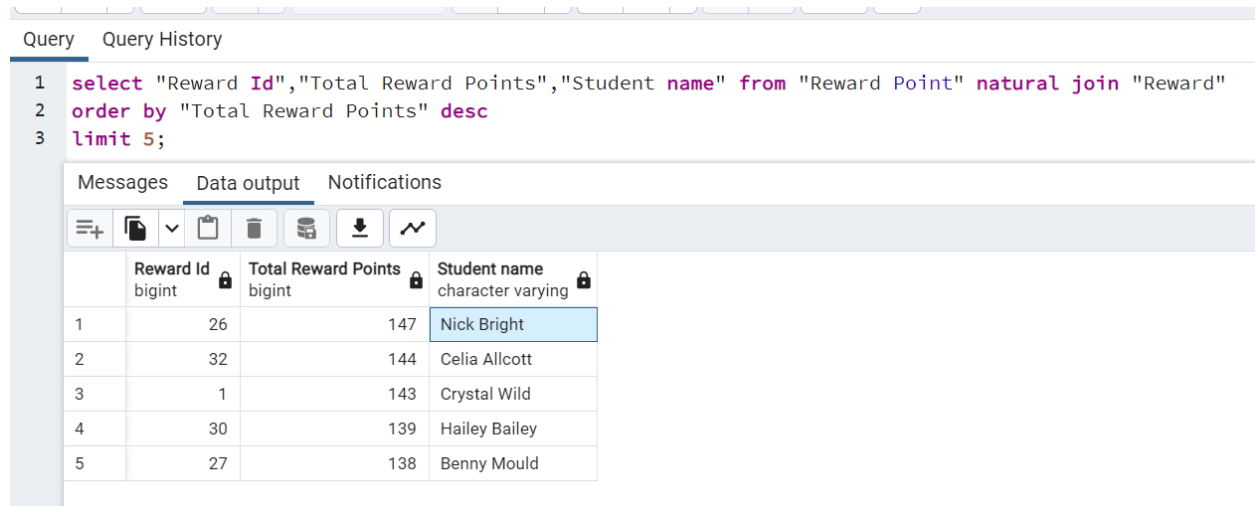
```

Messages Data output Notifications

	Subject Id character varying	Subject name character varying	Professor Id character varying	Student Id bigint	Student name character varying	Ac Id bigint	Quiz point integer	Exam point integer	Bonus point integer	Lab point integer
1		Design and techn...	10	32	Daron Brooks	14	8	33	15	9
2		Modern Literature	10	27	Adalind Morris	48	15	26	1	13
3		Earth Science	10	36	Daniel Ralph	83	2	8	1	12

15. Display Reward id, student name and total reward points of top 5 students who have scored maximum points.

-> select "Reward Id", "Total Reward Points", "Student name" from "Reward Point"
natural join "Reward"
order by "Total Reward Points" desc
limit 5;



The screenshot shows a database query interface with a query editor and a results pane. The query editor contains the following SQL query:

```
1 select "Reward Id", "Total Reward Points", "Student name" from "Reward Point" natural join "Reward"
2 order by "Total Reward Points" desc
3 limit 5;
```

The results pane shows the output of the query, which is a table with 5 rows and 3 columns: Reward Id, Total Reward Points, and Student name. The table is sorted by Total Reward Points in descending order.

	Reward Id bigint	Total Reward Points bigint	Student name character varying
1	26	147	Nick Bright
2	32	144	Celia Allcott
3	1	143	Crystal Wild
4	30	139	Hailey Bailey
5	27	138	Benny Mould

16. Display details of subject points with subject id of the student of academic year 2020.

->select "Student"."Student Id", "Subject Id", "Quiz point", "Exam point", "Bonus point", "Lab point", "Academic year"
from "Student" join "Sub points"
on "Student"."Student Id" = "Sub points"."Student Id"
where "Academic year"='2020';

```

1 select "Student"."Student Id","Subject Id","Quiz point","Exam point","Bonus point","Lab point","Academic year"
2 from "Student" join "Sub points"
3 on "Student"."Student Id" = "Sub points"."Student Id"
4 where "Academic year"='2020';

```

Messages Data output Notifications



	Student Id bigint	Subject Id character varying	Quiz point integer	Exam point integer	Bonus point integer	Lab point integer	Academic year integer
1	16	5	14	27	10	14	2020
2	16	15	3	34	3	3	2020
3	16	19	10	39	1	3	2020
4	39	9	5	31	4	16	2020
5	39	9	7	36	8	18	2020
6	27	10	15	26	1	13	2020
7	18	10	5	23	15	3	2020
8	39	6	8	38	11	16	2020
9	40	14	6	16	16	4	2020
10	36	18	2	8	1	12	2020
11	27	2	10	7	16	12	2020
12	29	6	8	17	10	8	2020

Total rows: 13 of 13 Query complete 00:00:00.051

Ln 4, Col

17. Display details of professor with subject which are taken by students.

```

->select "Professor"."Professor Id ","Professor name","Subject Id","Subject
name" from "Professor" join "Subject"
on "Professor"."Professor Id " = "Subject"."Professor Id"
where "Subject Id" in (select "Takes"."Subject Id" from "Takes" join "Subject"
on "Takes"."Subject Id" = "Subject". "Subject Id")

```

Query

Query History

1

```
select "Professor"."Professor Id ", "Professor name", "Subject Id", "Subject name" from "Professor" join "Subject"
on "Professor"."Professor Id " = "Subject"."Professor Id"|
where "Subject Id" in (select "Takes"."Subject Id" from "Takes" join "Subject"
on "Takes"."Subject Id" = "Subject". "Subject Id")
```

2

Messages

Data output

Notifications

	Professor Id character varying	Professor name character varying	Subject Id character varying	Subject name character varying
1	37	Kieth Neal	1	Resource Program
2	12	Mayleen Mccall	2	Economics
3	13	Joseph Rose	3	German
4	18	Gladys Funnell	4	American Literature
5	39	Anabel Brooks	5	Instrumental Music
6	4	Janelle Richards	6	Modern Literature
7	10	Candice Lee	8	Design and technolo...
8	4	Janelle Richards	9	Spanish
9	17	Jack Huggins	11	Language Arts
10	25	Jack Edley	12	Basic Math
11	38	Charlotte Walter	13	Modern Literature
12	11	Christine Rodwell	14	Leadership

Total rows: 16 of 16

Query complete 00:00:15.624

Ln 2,

18. Show the details of non academic which has same non-academic point as non academic.

->select * from "Non Academic"
where "NAc Id" in (select "NAc Id" from "Non-academic point")

Query

Query History

```

1 select * from "Non Academic"
2 where "NAc Id" in (select "NAc Id" from "Non-academic point")

```

Loading...

Messages

Data output

Notifications

+

📄

▼

📋

🗑️

🗄️

⬇️

📈

	Student name character varying	NAc Id [PK] bigint	Total non-academic point integer
1	Lucy Edler	1	43
2	Julius Rothwell	2	4
3	Fred Archer	3	14
4	Anabelle Spencer	4	21
5	Alice Moran	5	17
6	Kendra Morris	6	11
7	Elijah Neal	7	37
8	Vera Ramsey	8	3
9	Sharon Richardson	9	18
10	Gil Egerton	10	16
11	Harvey Wade	11	9
12	Dakota Brennan	12	15

Total rows: 45 of 45

Query complete 00:02:07.276

19. Find the student details of 2020 batch. Return a temp table with Student id, Student name, Student's branch, Academic year in the result table.

->

```

create or replace function "student_temp"()
returns table (S_Id bigint, S_name character varying, S_branch character
varying, Ac_year integer )
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
R_LIST2 record;
BEGIN
CREATE TEMP TABLE test1 (a1 bigint, b1 character varying, c1 character
varying, d1 integer) ON
COMMIT DROP;

```

```

FOR R_LIST2 in (select "Student Id", "Student name", "Student's branch",
"Academic year" from "Student" where "Academic year"='2020')
loop
Insert into test1 (a1, b1, c1,d1) values (R_LIST2."Student Id",R_LIST2."Student
name", R_LIST2."Student's branch", R_LIST2."Academic year");
end loop;
RETURN QUERY TABLE test1;
END;
$BODY$;

```

```

select "student_temp"();

```

Query

Query History

```

1 create or replace function "student_temp"()
2 returns table (S_Id bigint, S_name character varying, S_branch character varying, Ac_year integer )
3 LANGUAGE 'plpgsql'
4 AS $BODY$
5 DECLARE
6 R_LIST2 record;
7 BEGIN
8 CREATE TEMP TABLE test1 (a1 bigint, b1 character varying, c1 character varying, d1 integer) ON
9 COMMIT DROP;
10 FOR R_LIST2 in (select "Student Id", "Student name", "Student's branch", "Academic year" from "Student" where "Academic year"='2020')
11 loop
12 Insert into test1 (a1, b1, c1,d1) values (R_LIST2."Student Id",R_LIST2."Student name", R_LIST2."Student's branch", R_LIST2."Academic year");
13 end loop;
14 RETURN QUERY TABLE test1;
15 END;
16 $BODY$;
17
18 select "student_temp"();

```

5 DECLARE

Messages

Data output

Notifications

+

📄

▼

📋

🗑️

📦

⬇️

📈

	student_temp record
1	(16,"Penelope Redden",History,2020)
2	(18,"Javier Ross","Basic Math",2020)
3	(23,"Liam Nash",Mathematics,2020)
4	(27,"Rosalee Murray",Latin,2020)
5	(29,"Nicole Rainford","Basic Math",2020)
6	(36,"Clint Casey","Ancient Civilizations",2020)
7	(39,"Hannah Woodley","American Literature",2020)
8	(40,"Percy Brooks",Leadership,2020)

20. Create a column "Is_teaching" in the Professor table. Create a trigger to put the default value Yes for every new entry if nothing is given by the user. Insert new records and check the functionality of Triggers.

->

```
ALTER TABLE "Professor"
ADD COLUMN "Is_teaching" VARCHAR;
```

```
CREATE OR REPLACE FUNCTION "put"()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
if New."Is_teaching" is NULL THEN
UPDATE "Professor" SET "Is_teaching"='YES'
WHERE NEW."Professor Id "="Professor Id ";
End if;
RETURN NEW;
END
$BODY$;
CREATE TRIGGER Trigger_insert_2
AFTER INSERT
ON "Professor"
FOR EACH ROW
EXECUTE PROCEDURE "put"();
```

```
Insert into "Professor"
values(50,'Jim kurose');
```

```
select *
from "Professor"
where "Professor Id "='50'
```

```
Insert into "Professor"
values(51,'David marker','NO');
```

```
select *
from "Professor"
where "Professor Id "='51'
```

```

22
23 Insert into "Professor"
24 values(50,'Jim kurose');
25
26
27 select *
28 from "Professor"
29 where "Professor Id" ='50'
30

```

Messages Data output Notifications



	Professor Id [PK] character varying	Professor name character varying	Is_teaching character varying
1	50	Jim kurose	YES

Total rows: 1 of 1 Query complete 00:00:00.263

```

32 Insert into "Professor"
33 values(51,'David marker','NO');
34
35
36 select *
37 from "Professor"
38 where "Professor Id" ='51'
39
40
41
42

```

Messages Data output Notifications

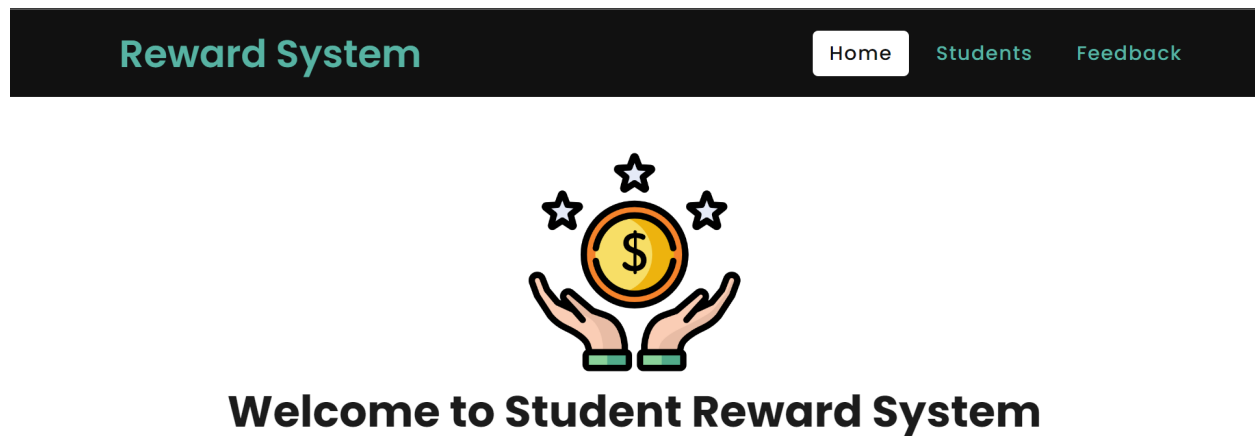


	Professor Id [PK] character varying	Professor name character varying	Is_teaching character varying
1	51	David marker	NO

Total rows: 1 of 1 Query complete 00:00:15.162

★ Screenshots of the website that connect the database:-

1. Home page:-



2. List of students:-

List of students				
For new Registration Student's list of year 2020				
Student id	Student name	Student's branch	Academic year	
3	Tyler Parker	Trigonometry	2021	Edit Delete
4	Paula Hopkinson	Ancient Civilizations	2021	Edit Delete
5	Luke Miller	Accounting	2019	Edit Delete
6	Clint Cowan	Latin	2019	Edit Delete
7	Danny Ainsworth	Sociology	2021	Edit Delete
8	Brad Bailey	Physical Education	2017	Edit Delete
9	Aeris Hunt	French	2019	Edit Delete
10	Piper Thorne	Resource Program	2017	Edit Delete
11	Chris Marshall	American Literature	2021	Edit Delete
12	Stacy Locke	American Literature	2017	Edit Delete
13	Benjamin Grant	Sociology	2019	Edit Delete
14	Mike Hammond	Grammar	2021	Edit Delete
15	Brad Dann	Basic Math	2017	Edit Delete
16	Penelope Redden	History	2020	Edit Delete
17	Enoch Evans	Grammar	2018	Edit Delete

17	Enoch Evans	Grammar	2018	Edit	Delete
18	Javier Ross	Basic Math	2020	Edit	Delete
19	Hope Mcleod	Mathematics	2021	Edit	Delete
20	Margot Terry	Modern Literature	2018	Edit	Delete
21	Chris Mackenzie	LOGIC	2019	Edit	Delete
22	Audrey Boden	Art	2021	Edit	Delete
23	Liam Nash	Mathematics	2020	Edit	Delete
24	Denis Booth	Instrumental Music	2021	Edit	Delete
25	Phillip Harrington	Modern Literature	2018	Edit	Delete
26	Alexander Sherry	Ancient Civilizations	2017	Edit	Delete
27	Rosalee Murray	Latin	2020	Edit	Delete
28	Erick Vincent	English IV	2019	Edit	Delete
29	Nicole Rainford	Basic Math	2020	Edit	Delete
30	Cedrick Gates	Dramatics	2019	Edit	Delete
31	Bree Rothwell	American Literature	2018	Edit	Delete
32	Tara Preston	Ecology	2017	Edit	Delete
33	Paula Locke	Design and technology	2017	Edit	Delete
34	Joy Cowan	Trigonometry	2021	Edit	Delete
35	Tom Doherty	Science	2021	Edit	Delete
36	Clint Casey	Ancient Civilizations	2020	Edit	Delete
37	Kurt Rust	Spanish	2021	Edit	Delete
38	Adelaide Booth	Dramatics	2019	Edit	Delete

38	Adelaide Booth	Dramatics	2019	Edit	Delete
39	Hannah Woodley	American Literature	2020	Edit	Delete
40	Percy Brooks	Leadership	2020	Edit	Delete
41	Parker Wood	Speech	2021	Edit	Delete
42	Rosalee Webster	Music	2018	Edit	Delete
43	Renee Donnelly	Instrumental Music	2021	Edit	Delete
44	Harvey Archer	Geography	2018	Edit	Delete
45	Adalind Stevens	Science	2019	Edit	Delete
1	Lexi Walsh	Accounting	2019	Edit	Delete
2	Peyton Brock	Health	2018	Edit	Delete

[Go to home](#)

3. When we click on new registration this window will be open so that you can insert new data of students.

Insert the data

Student_id	Enter Student Id..
Student_name	Enter Student name..
Students_branch	Enter Students branch..
Academic_year	Enter Academic year..
Insert	

[Go back](#)

After inserting one student details:-

Insert the data

Student_id	Enter Student Id..
Student_name	Enter Student name..
Students_branch	Enter Students branch..
Academic_year	Enter Academic year..
Insert	Student Neel Mehta is Saved successfully..!

[Go back](#)

When we click go back and check the list of students we found one added entry which is shown below.(last one)

39	Hannah Woodley	American Literature	2020	Edit	Delete
40	Percy Brooks	Leadership	2020	Edit	Delete
41	Parker Wood	Speech	2021	Edit	Delete
42	Rosalee Webster	Music	2018	Edit	Delete
43	Renee Donnelly	Instrumental Music	2021	Edit	Delete
44	Harvey Archer	Geography	2018	Edit	Delete
45	Adalind Stevens	Science	2019	Edit	Delete
1	Lexi Walsh	Accounting	2019	Edit	Delete
2	Peyton Brock	Health	2018	Edit	Delete
46	Neel Mehta	Cs_Ict	2020	Edit	Delete

[Go to home](#)

4.We updated one student's detail whose id is 46.

Edit details

Student Id	46
Student name	Om Limbachiya
Students_branch	Cs
Academic_year	2021
Update Record	Record Updated Successfully..!

[Go back](#)

When we click go back and check the list of students we found one updated entry which is shown below.(last one)

39	Hannah Woodley	American Literature	2020	Edit	Delete
40	Percy Brooks	Leadership	2020	Edit	Delete
41	Parker Wood	Speech	2021	Edit	Delete
42	Rosalee Webster	Music	2018	Edit	Delete
43	Renee Donnelly	Instrumental Music	2021	Edit	Delete
44	Harvey Archer	Geography	2018	Edit	Delete
45	Adalind Stevens	Science	2019	Edit	Delete
1	Lexi Walsh	Accounting	2019	Edit	Delete
2	Peyton Brock	Health	2018	Edit	Delete
46	Om Limbachiya	Cs	2021	Edit	Delete

[Go to home](#)

5. When we click on delete then the browser will ask us are you sure want to delete the record?

127.0.0.1:8000 says

Are You Sure want to Delete the Record ?

OK Cancel

30				Edit Delete
31				Edit Delete
32				Edit Delete
33				Edit Delete
34	Joy Cowan	Trigonometry	2021	Edit Delete
35	Tom Doherty	Science	2021	Edit Delete
36	Clint Casey	Ancient Civilizations	2020	Edit Delete
37	Kurt Rust	Spanish	2021	Edit Delete
38	Adelaide Booth	Dramatics	2019	Edit Delete
39	Hannah Woodley	American Literature	2020	Edit Delete
40	Percy Brooks	Leadership	2020	Edit Delete
41	Parker Wood	Speech	2021	Edit Delete
42	Rosalee Webster	Music	2018	Edit Delete
43	Renee Donnelly	Instrumental Music	2021	Edit Delete
44	Harvey Archer	Geography	2018	Edit Delete
45	Adalind Stevens	Science	2019	Edit Delete
1	Lexi Walsh	Accounting	2019	Edit Delete
2	Peyton Brock	Health	2018	Edit Delete
46	Om Limbachiya	Cs	2021	Edit Delete

127.0.0.1:8000/Delete/46

[Go to home](#)

After clicking on OK button the entry will be removed(Deleted) from table.

33	Paula Locke	Design and technology	2017	Edit Delete
34	Joy Cowan	Trigonometry	2021	Edit Delete
35	Tom Doherty	Science	2021	Edit Delete
36	Clint Casey	Ancient Civilizations	2020	Edit Delete
37	Kurt Rust	Spanish	2021	Edit Delete
38	Adelaide Booth	Dramatics	2019	Edit Delete
39	Hannah Woodley	American Literature	2020	Edit Delete
40	Percy Brooks	Leadership	2020	Edit Delete
41	Parker Wood	Speech	2021	Edit Delete
42	Rosalee Webster	Music	2018	Edit Delete
43	Renee Donnelly	Instrumental Music	2021	Edit Delete
44	Harvey Archer	Geography	2018	Edit Delete
45	Adalind Stevens	Science	2019	Edit Delete
1	Lexi Walsh	Accounting	2019	Edit Delete
2	Peyton Brock	Health	2018	Edit Delete

[Go to home](#)

★ Query Run:

When we click on student's list of year 2020, query run in the backend and it shows the output of students of batch 2020.

Student's of 2020 batch			
Student id	Student name	Student's branch	Academic year
16	Penelope Redden	History	2020
18	Javier Ross	Basic Math	2020
23	Liam Nash	Mathematics	2020
27	Rosalee Murray	Latin	2020
29	Nicole Rainford	Basic Math	2020
36	Clint Casey	Ancient Civilizations	2020
39	Hannah Woodley	American Literature	2020
40	Percy Brooks	Leadership	2020
Go back			