OPIM – 5272 – BUSINESS PROCESS MODELING AND DATA MANAGEMENT– SECB17 – 1178
IMPROVE PROCESS EFFICIENY FOR FUND DISBURSEMENT – MAA..PREMA ORGANIZATION

PROJECT REPORT

ABOUT THE ORGANIZATION:

"Maa..Prema" is an NGO based out of Visakhapatnam, India registered under the Society Acts 2001 (Act 35 of 2001). It was founded in December 2010 and has taken up lot of initiatives since it's inception, majorly concentrating on social issues like poverty, illiteracy and unemployment of people below the poverty line. The most important ones being:

a. "Project Vidyamrutham" – SPONSORING OF EDUCATION:
A thorough survey is conducted in the underdeveloped regions of Visakhapatnam and eligible children (children from the below poverty line segment and those who clear the entrance exam) are sponsored education. 236 children are being educated under this project.

b. "Project Premamrutham" – SPONSORING OF MONTHLY RATION:
Sponsoring monthly ration items to the physically challenged people. Currently 108 people are under this scheme.

c. "Youth, We Build the Nation" – REGULAR SOCIAL SERVICE ACTIVITIES:
On the 3rd Sunday of every month, volunteers group together at a place and take up a service activity in areas in and around Visakhapatnam. These activities generally include blood donation camps, plantation drives, food distribution, awareness programs etc.

Over 500 families are being directly benefitted through this NGO and more than 200 volunteers involve selflessly for the upliftment of the organization.

SCOPE OF THE PROJECT:

Identify a critical process in the organization's education sponsorship program and determine the process AS IS. Suggest process improvement and work on the possible RDBMS design for the organization (which is currently unavailable).

PROCESS IN CONSIDERATION:
"Fund Disbursement to newly enrolled student beneficiaries"

ABOUT THE PROCESS:

a. Triggering Event:
February 1st every year which is 4 months prior to the start of academic year in India.

b. Major Interrelated Tasks include:

- VOLUNTEER conducting survey for underprivileged children
- STUDENT giving the entrance exam
- TEACHER evaluating the entrance exam
- DONOR registering themselves as official monthly donors for the organization
- VOLUNTEER mapping students to the donors

- Students enrolling at SCHOOL
- DONOR donating monthly funds to the organization by one of the 3 possible ways
- VOLUNTEER issuing the check to STUDENT
- STUDENT making the fee payment
- SCHOOL issuing the fee receipt
- STUDENT submitting the receipt to VOLUNTEER
- VOLUNTEER updating DONOR about payment by providing receipt as proof.

c. Specific Result:

Disbursal of funds to the newly enrolled student beneficiaries.

d. Customer:

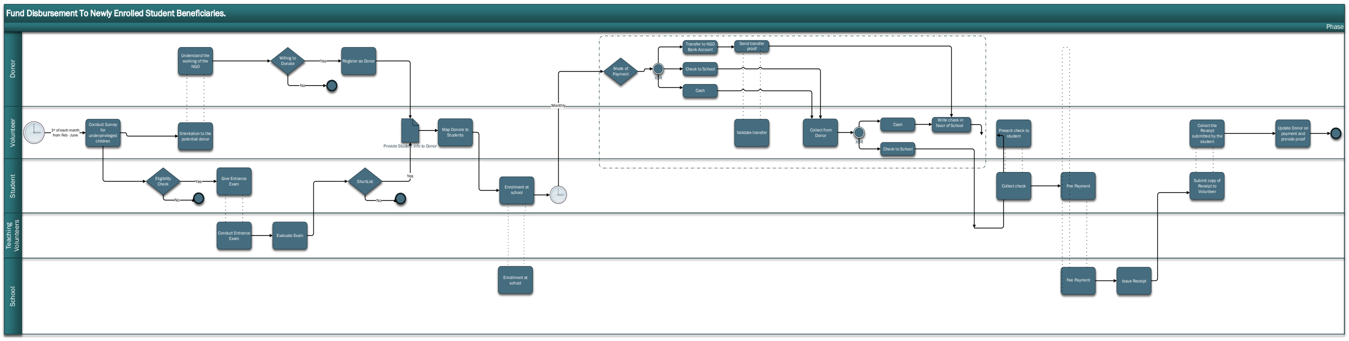
Qualified underprivileged Students registered with the NGO

e. Stakeholders:

- Donors
- Volunteers
- Students
- Teaching Volunteers
- School

AS IS:

The process AS IS is drafted using Microsoft Visio Professional 2016:



Maa..Prema Fund
Disbursment- Swiml

CHALLENGES IDENTIFIED:

1. All the information related to the organization is currently being maintained in LOG BOOKS which is an inefficient way of data storage and data retrieval.

2. Daunting task for volunteers to continue surveys for 4 months (Surveys generally start during 1st of each

month starting from February and end in May).

3. Overall process time is too long – It takes 7 months for the entire process to complete after initiation.

4. Sometimes fee payment deadlines are being missed as there is an overwhelming load on the volunteer team to coordinate with the donors and beneficiaries.

SOLUTIONS IMPLEMENTED & BENEFITS OBTAINED:

1. Database has been designed and relevant Entity Relationship Diagrams have been drafted.

This solution improves the process in the following manner:

a. Volunteers need not spend more time entering the values in the LOG BOOKS manually.

b. No data duplication. This has been achieved by giving relevant primary key constraints.

c. Data accuracy achieved by implementing relevant checks to each attribute in the tables.

d. Easier data manipulation compared to manually changing information in all logs.

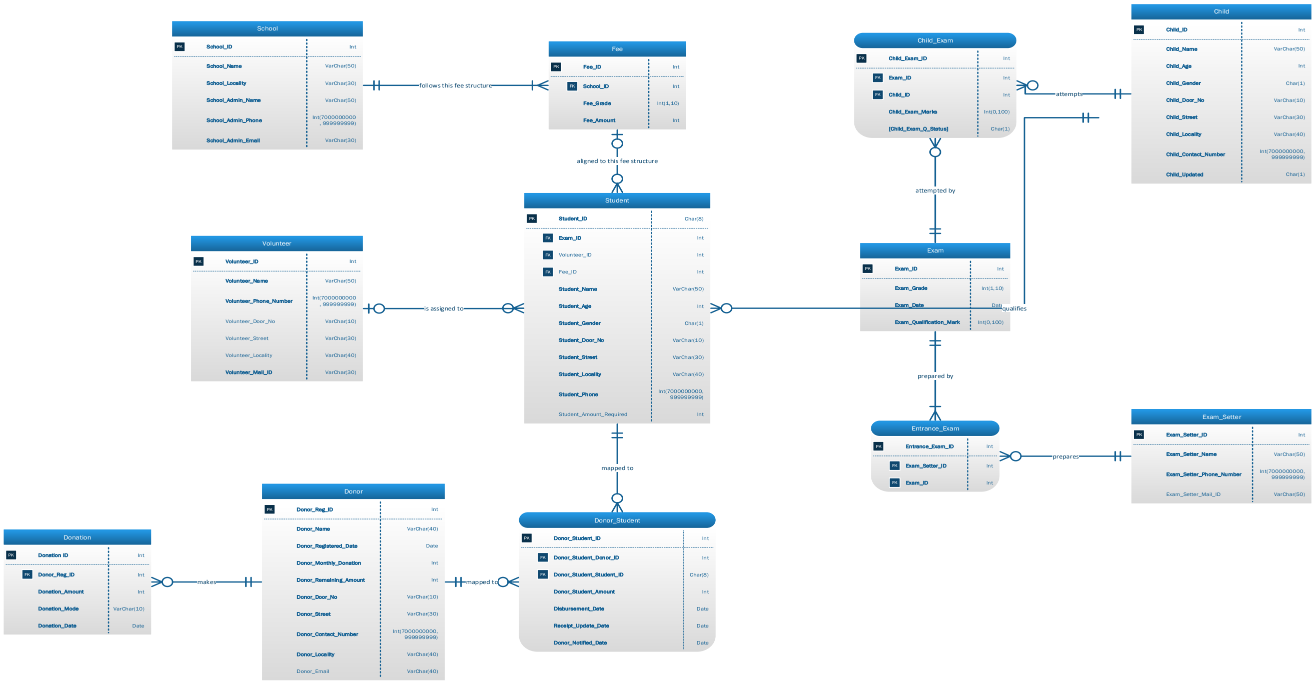
e. Overall reduction in process time

f. Queries can be executed on the database to create reports based on the specific requirements and these can be leveraged for taking important business decisions.

g. Fees overdue can be kept in check by executing queries daily and relevant student and donor can be updated on the status.

DATABASE DESIGN:

Entity Relationship Diagram:


School
PK School_ID Int
School_Name VarChar(50)
School_Locality VarChar(30)
School_Admin_Name VarChar(50)
School_Admin_Phone Int(7000000000 , 999999999)
School_Admin_Email VarChar(30)
follows this fee structure
Fee
PK Fee_ID Int
FK School_ID Int
Fee_Grade Int(1,10)
Fee_Amount Int
aligned to this fee structure
Child_Exam
PK Child_Exam_ID Int
FK Exam_ID Int
FK Child_ID Int
Child_Exam_Marks Int(0,100)
[Child_Exam_Q_Status] Char(1)
attempts
Child
PK Child_ID Int
Child_Name VarChar(50)
Child_Age Int
Child_Gender Char(1)
Child_Door_No VarChar(10)
Child_Street VarChar(30)
Child_Locality VarChar(40)
Child_Contact_Number Int(7000000000, 999999999)
Child_Updated Char(1)
attempted by
Volunteer
PK Volunteer_ID Int
Volunteer_Name VarChar(50)
Volunteer_Phone_Number Int(7000000000 , 999999999)
Volunteer_Door_No VarChar(10)
Volunteer_Street VarChar(30)
Volunteer_Locality VarChar(40)
Volunteer_Mail_ID VarChar(30)
is assigned to
Student
PK Student_ID Char(8)
FK Exam_ID Int
FK Volunteer_ID Int
FK Fee_ID Int
Student_Name VarChar(50)
Student_Age Int
Student_Gender Char(1)
Student_Door_No VarChar(10)
Student_Street VarChar(30)
Student_Locality VarChar(40)
Student_Phone Int(7000000000, 999999999)
Student_Amount_Required Int
Exam
PK Exam_ID Int
Exam_Grade Int(1,10)
Exam_Date Date
Exam_Qualification_Mark Int(0,100)
qualifies
prepared by
Entrance_Exam
PK Entrance_Exam_ID Int
FK Exam_Setter_ID Int
FK Exam_ID Int
prepares
Exam_Setter
PK Exam_Setter_ID Int
Exam_Setter_Name VarChar(50)
Exam_Setter_Phone_Number Int(7000000000, 999999999)
Exam_Setter_Mail_ID VarChar(50)
mapped to
Donation
PK Donation ID Int
FK Donor_Reg_ID Int
Donation_Amount Int
Donation_Mode VarChar(10)
Donation_Date Date
makes
Donor
PK Donor_Reg_ID Int
Donor_Name VarChar(40)
Donor_Registered_Date Date
Donor_Monthly_Donation Int
Donor_Remaining_Amount Int
Donor_Door_No VarChar(10)
Donor_Street VarChar(30)
Donor_Contact_Number Int(7000000000, 999999999)
Donor_Locality VarChar(40)
Donor_Email VarChar(40)
mapped to
Donor_Student
PK Donor_Student_ID Int
FK Donor_Student_Donor_ID Int
FK Donor_Student_Student_ID Char(8)
Donor_Student_Amount Int
Disbursement_Date Date
Receipt_Update_Date Date
Donor_Notified_Date Date

ERD_Team 3_BPMDM_Final.vsd

Entity Description based on Business Rules:

The entities and attributes of the database are as follows:

CHILD: An entity type for instances of under-privileged children who were surveyed. Each child takes an exam and needs to qualify in the exam to be registered to the NGO as a STUDENT. Attributes of the child including a unique ID, name, age, gender, address, contact number are to be recorded. We also have a field that needs to be updated when the qualification check has been performed on the CHILD. (Y – update, N-Yet to update). The default value for this attribute is 'N'.

EXAM: An entity to store the details of each of the exams conducted. Each exam will have a unique Exam_ID and other details like exam date, grade for which the exam is conducted and exam qualification mark which is used to determine the students who are qualified (Student with Exam marks greater than Qualification Mark is qualified). Each exam may have different qualification mark.

CHILD_EXAM: Associative entity to store the information of all the children who gave the exam and their qualification status for that particular exam along with the marks obtained. This is used to create STUDENT entity which has only qualified children. Child_Exam_Q_Status is a derived attribute obtained by checking whether obtained marks are greater than the qualification marks of that particular exam. CHILD_EXAM_ID is the unique ID for Child and Exam combination.

ENTRANCE_EXAM: An associative entity to store the information about exam setter responsible for preparing and evaluating each exam.

EXAM_SETTER: An entity to store the information of the exam setters who are registered with the organization and are authorized to conduct and evaluate the exams. It consists details of the exam setter which includes exam setter's name, contact number and email id. Exam setter may or may not have an email ID. Other details are required to be stored.

STUDENT: In this entity we store Student details. Students are those instances from CHILD entity who are qualified in the exam and registered with the NGO. The entity stores details of unique student id, student name, age, gender, address, contact number, fee requirement of the student. Once a student registers at the NGO he/she gets a new unique student ID which is different from the child ID.

VOLUNTEER: This entity stores information of each volunteer working for the NGO. It is required to store Name, Phone number and email ID. Additionally, his/her address can also be saved. Every volunteer is identified by a unique ID.

SCHOOL: An entity to store the information of all the schools that are partnered with the NGO. Each school is present in the database with a unique School_ID along with name and address of the school. In addition to this, details of the school admin including name, phone number and email ID are stored in this database all of which are mandatory.

FEE: An entity to store the Fee amount for every grade in each school. The entries in this entity will be for the unique combinations of (School and Grade). Fee_ID has to be in the format 602 – which means School_ID is 6 and 2nd grade for the ease of database operator.

DONOR: An entity is created to save donor information including his/her unique Registration ID, Name, Registered date, Monthly donation Amount, Unassigned Amount, Phone Number and Address all of which are mandatory. Additionally, an email id can also be stored. Unassigned amount for a Donor is the remaining amount of the donation after allocating a part of it to a student.

DONATION: For every donation made each month, a log will be saved in the database. Each donation made will be stored along with the Donor Registration ID, Donated Amount, Date of donation made and the mode of payment. A donor might have made multiple donations or might not have made any donation yet.

DONOR_STUDENT: An associative entity to store the Donor – Student mapping along with the amount associated between both. Once the donation amount is disbursed to a particular student, the latest 'Disbursement_Date' is updated. The date the student updates the NGO about the school fee payment is stored as 'Reciept_Update_Date'. After this the volunteer updates the donor that the student fee payment has been done and a proof is sent to the donor. The date when this update is given is stored in the database as 'Donor_Notified_Date'.

Relationships:

Every child who is surveyed can take multiple exams or might have not attended even one exam. There is one exam conducted for each class and the child who wants to be admitted to that grade may attend that exam.

Each exam is set by one or more exam setters and each exam setter who has agreed to, might or not have set an exam till date.

Every student is mapped to a volunteer who will handle the donations for him/her. But, there may be students who are just registered and do not have volunteer assigned yet. There can be a volunteer who is not mapped to any student. A volunteer can also handle many students. The link between the donor and the volunteer is only through the student assigned.

Once a student is registered at the NGO he/she is to be admitted to school and will be aligned to a specific fee structure based on the school and grade he will be attending. There might be a student who has just registered and has not yet been admitted to a school. A school in the database may not have any students from the NGO at all.

Each grade of the school has a specific fee and fee might differ from one school to the other. Fee for each grade in all the schools are present in the FEE table whether or not a student is enrolled in that particular grade of the school.

Students and donors are mapped based on the fee requirement of the students and the monthly donation amount. One donor can cater to the needs of more than one student. And one student's fee requirement can be fulfilled by one or more donors. Only when a student-donor relationship has been established an entry will be made in the donor-student table. There can be students with no donor mapped to them yet and there can be new donors who have not yet been mapped to any student.

QUERIES TO OBTAIN IMPORTANT BUSINESS REPORTS:

The use of databases facilitates the creation of reports which are useful to define improved business processes. A few reports and their usage have been described below

SET 1:

1. Generate a report which gives a summary of children that gave an exam for each grade conducted in a particular year.

This report provides a grade level summary of the count of students who gave an exam in the specified year that is input. The user needs to enter a specific year and the number of children who attended an exam will be outputted for each grade in descending order of the count of children.

This report is useful to get a picture of how many children have attended an exam of a particular grade and the decisions can be taken on the number of schools to approach or the qualification mark of a particular exam that was conducted.

```
SELECT EXAM_GRADE,COUNT(CHILD_ID) AS ATTENDING_BY
FROM EXAM E LEFT JOIN CHILD_EXAM CE
ON (E.EXAM_ID = CE.EXAM_ID)
WHERE EXTRACT(YEAR FROM E.EXAM_DATE)= &YEAR
GROUP BY EXAM_GRADE
ORDER BY ATTENDING_BY DESC;
```

2. Provide the details of the donors who have not donated from specified number of days.

This report is very useful to follow-up with the registered donors and subtly remind them about the cause. The report provides the Donor ID, Name, and the contact number along with the number of days that elapsed from the last donation. The report outputs donors that have not donated since a longest time on top of the list and so on.

This report makes it more easier to follow-up with donors in an orderly manner than randomly calling and disturbing the ones that have probably made recent donations.

```
CREATE VIEW DON AS
SELECT DT.DONOR_REG_ID, D.DONOR_NAME, D.DONOR_CONTACT_NUMBER,CEIL(SYSDATE - MAX(DONATION_DATE))DAYS
FROM DONATION DT JOIN DONOR D ON DT.DONOR_REG_ID = D.DONOR_REG_ID
GROUP BY DT.DONOR_REG_ID, D.DONOR_NAME,D.DONOR_CONTACT_NUMBER;

SELECT DONOR_NAME,DONOR_REG_ID,DONOR_CONTACT_NUMBER,DAYS "DAYS SINCE LAST DONATION"
FROM DON
WHERE DAYS > &DAYS_AFTER_LAST_DONATION
ORDER BY DAYS DESC ;
```

SET 2:

Mapping of donors to students in a tedious task and the most important one. The NGO maps donors to students based on the monthly donations given made by the donors and the student’s monthly fee requirements. A fraction of the monthly donation can be allotted to a student and a specific amount might be left unassigned. This can be mapped to a new student and a mapping history is stored. Any activity regarding this donor-student pair is taken care by the volunteer associated with the respective student.

Similarly, a single student can be supported by multiple donors and some students might still have a portion of fee left unsupported. This amount can be used to map them to a new donor.

Manually checking for these allotments is tough and randomly choosing the combinations might leave out residual amounts in most cases resulting in a lot of unused but otherwise useful amount. To ease this, one of the reports that can be generated is described below.

3. *For each student whose student ID is entered, to output the donor with the least possible donation amount which is not yet assigned to any other student and is available to independently fulfill the student fee requirement. In case, more than one users have the same amount, output the one whose donation has been least used.*

```
SELECT D.DONOR_REG_ID, D.DONOR_REMAINING_AMOUNT
FROM DONOR D
WHERE D.DONOR_REMAINING_AMOUNT = (SELECT MIN(D1.DONOR_REMAINING_AMOUNT)
                    FROM DONOR D1
                    WHERE D1.DONOR_REMAINING_AMOUNT >= (SELECT S.STUDENT_AMOUNT_REQUIRED
                                        FROM STUDENT S
                                        WHERE STUDENT_ID = '&STUDENT_ID'))
AND ROWNUM =1
ORDER BY DONOR_MONTHLY_DONATION DESC ;
```

Set 3:

The most important operations at the NGO are timely updates of donations made, disbursed and updated. There needs to be a concrete process flow where at any point in time information about pending updates need to be available in order to run any organization smoothly. Checkpoints need to be defined in the process and necessary steps need to be taken when required.

4. All the students who have received a donation but have not updated the NGO yet about the payment of the fee and the respective amount.

Once a donation has been made and disbursed, the student has to pay the fee at school and submit the receipt the receipt at the NGO. A report can be generated to check if this process is intact and all the students have updated the FEE payment event. The date of updation of the last donation will be the saved in the database which can be check against the last disbursement date.

```
SELECT DISTINCT DONOR_STUDENT_STUDENT_ID

FROM DONOR_STUDENT

WHERE  Donor_Student.Receipt_Update_Date < Donor_Student.Disbursement_Date;
```

5. Report to list all the volunteers who have pending updates that need to be made to the donor about receipt of the donations that have been made.

Once the fee update is given by the student, it is the responsibility of the volunteer to update the Donor about the fee payment of that particular student and needs to send a e-copy of the same.

Each volunteer may be assigned to several students and needs to keep a track of all the events and updated being made. He will have tasks to perform accordingly. It is essential for him to know what are the pending tasks for him if any.

```
SELECT VOLUNTEER_NAME
FROM VOLUNTEER V JOIN STUDENT S
ON( V.VOLUNTEER_ID = S.VOLUNTEER_ID)
                JOIN DONOR_STUDENT DS
                ON(  DS. DONOR_STUDENT_STUDENT_ID = S.STUDENT_ID)
WHERE DS.DONOR_NOTIFIED_DATE < DS.RECEIPT_UPDATE_DATE;
```

A code can be run each time a set of children are surveyed and have written an exam to insert the qualified ones as Students in the Student table.

```
--------------------------------------------------------------------------------------------------------------------------------------------------------
---------EACH TIME A NEW SURVEY IS MADE FOR UNDER-PRIVLEDGED CHILDREN AND AN EXAM IS CONDCTED------
----------THE STUDENT TABLE NEEDS TO BE UPDATED BASED ON THE QUALIFICATION MARK FROM THE EXAM------
--------------------------------------------------------------------------------------------------------------------------------------------------------
CREATE OR REPLACE VIEW STUDENT_VIEW AS
SELECT C.CHILD_ID, CE.EXAM_ID, C.CHILD_NAME, C.CHILD_AGE, C.CHILD_GENDER, C.CHILD_DOOR_NO, C.CHILD_STREET,
C.CHILD_LOCALITY,C.CHILD_CONTACT_NUMBER
FROM CHILD C JOIN CHILD_EXAM CE
ON(C.CHILD_ID = CE.CHILD_ID);

INSERT INTO STUDENT (STUDENT_NAME, STUDENT_AGE,
STUDENT_GENDER,STUDENT_DOOR_NO,STUDENT_STREET,STUDENT_LOCALITY,STUDENT_PHONE,EXAM_ID)
SELECT CHILD_NAME, CHILD_AGE, CHILD_GENDER, CHILD_DOOR_NO, CHILD_STREET,
CHILD_LOCALITY,CHILD_CONTACT_NUMBER,EXAM_ID FROM STUDENT_VIEW
WHERE CHILD_ID IN  (SELECT CHILD_ID
 FROM CHILD
 WHERE (CHILD_ID IN (SELECT CHILD_ID
                FROM CHILD_EXAM
                WHERE (CHILD_EXAM_Q_STATUS = 'Y')))
AND (CHILD_UPDATED = 'N'));

UPDATE CHILD
SET CHILD_UPDATED = 'Y';

----------------------------------------------
--END OF STUDENT UPDATION-----
---------------------------------------------
```