CS4D2A Project

Submitted by-Yash Pandey

17317629

D-Stream

Introduction:

I used to work for a non-government organisation as a volunteer. We focused on eradicating one of the major problems prevalent in the society that is begging. As we started our work we noticed that the two-main reason for begging are lack of awareness amongst people and lack of primary education and hence we divided our work into three sub categories anti-begging awareness, employment generation and primary teaching. The anti-begging team went to various places where the begging activity was more active and collected data about the beggars and children who were in those area, simultaneously employment generation team talked with several shop owners to collaborate with us to provide an employment opportunity to the beggars while teaching team set up schools for free of cost primary teaching. Hence all the three teams worked in hand to hand with each other.

As so much of the data collection is involved hence it is necessary to design and implement a database with the help of data collected by all the three teams can be put together and accessed by everyone as and when needed.

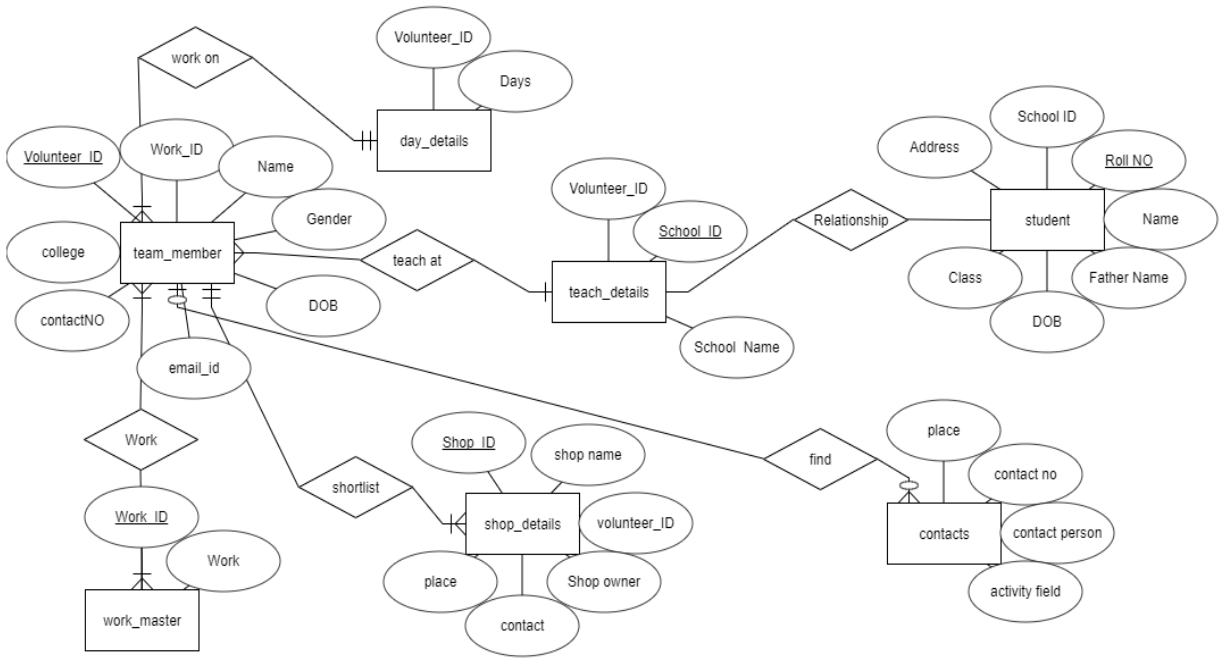
I have created seven entities in the data base which I think will be enough for smooth flowing of work.

All the information related to team members are stored in one entity. The work details that shows the relation between work and work id is stored in one entity. Student table stores all the information related to the students who joined our primary teaching. The days on which every volunteer will be working is stored in one entity. The details of all the schools are stored in one entity. All the information related to shop and its owner who are willing to collaborate with us are stored in one entity and the finally in the last table any kind of contact that any volunteer finds who is willing to help us in any way is stored in the contact table.

Various implicit and explicit constraints such as check constraint used to check gender in male, female or other, are used all over the database. Apart from this, I also tried to implement various procedures and triggers some of which I was able to implement but due to the minor syntax difference between oracle and php I was not able to implement all.

Codes for creation, population and all the procedures and triggers are mentioned at the end of this report.

Entity Relation Diagram



DETAILED DESCRIPTION OF EACH TABLE

1.Team Details->team_members

It contains all the required details of every team member

Volunteer_ID	Number(4)	Primary Key
Work_ID	Number(3)	Foreign Key
Name	Varchar(30)	Not Null
DOB	Date	Not Null
Gender	Char(1)	Check
Email_id	Varchar(50)	Unique
Contact No.	Number(10)	Unique
College	Varchar(60)	

2.Work Details->work_master

It consists of a relation of work_id and work

Work_ID	Number(3)	Primary Key
Work	Varchar(50)	

3.Student Details->student

It stores data about students

School_ID	Varchar(4)	Foreign Key
Roll Number	Number(5)	Primary Key
Name	Varchar(30)	Not Null
Father_Name	Varchar(30)	Not Null
DOB	Date	Not Null
Class	Varchar(6)	Not Null
Address	Varchar(50)	

4.Work_Days->day_details

It assigns work to volunteers

Volunteer_ID	Number(4)	Foreign Key
Days	varchar(30)	Not Null

5.School Details->teach_details

It stores information about volunteers associated with teaching

Volunteer_ID	Number(4)	Foreign Key
School_ID	Varchar(4)	Primary Key
School_Name	Varchar(20)	Not Null

6.Shop Details->shop_details

It stores information about shops involved in employment generation work

Shop_ID	Number(4)	Primary Key
Shop_Name	Varchar(30)	Not Null
Volunteer_ID	Number(4)	Foreign Key
Shop_Owner	Varchar(30)	Not Null
Contact	Number(10)	Unique
Place	Varchar(30)	Not Null

7.Contacts

This table contains contact info

Place	Varchar(30)	Not Null
Contact Person	varchar(30)	Not Null
Contact No	Number(10)	Unique ,Not Null
Activity Field	Varchar(50)	Not Null

Appendix

Codes:

```
-- phpMyAdmin SQL Dump
-- version 4.2.12deb2+deb8u3
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Nov 28, 2018 at 02:18 AM
-- Server version: 5.5.62-0+deb8u1
-- PHP Version: 5.6.38-0+deb8u1

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- Database: `pandeyy_db`
--

-- --------------------------------------------------------

--
-- Table structure for table `Contact`
--

CREATE TABLE IF NOT EXISTS `Contact` (
  `place` varchar(30) NOT NULL,
  `Contact_Person` varchar(30) NOT NULL,
  `ContactNo` int(10) NOT NULL,
  `Activity_Field` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `Contact`
--

INSERT INTO `Contact` (`place`, `Contact_Person`, `ContactNo`, `Activity_Field`) VALUES
('dublin', 'eimar', 855688123, 'employment generation'),
('dublin', 'adam', 855688562, 'Teaching'),
('dublin', 'martin', 855688589, 'Teaching'),
('cork', 'alan', 855688899, 'Teaching'),
('dublin', 'wren', 2147483647, 'anti begging');

--
-- Triggers `Contact`
--
DELIMITER //
CREATE TRIGGER `contins` BEFORE UPDATE ON `Contact`
```

```
 FOR EACH ROW BEGIN
    IF NEW.ContactNo < 0 THEN
       SET NEW.ContactNo = 0;

   END IF;
   END
//
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `day_details`
--

CREATE TABLE IF NOT EXISTS `day_details` (
  `Volunteer_ID` int(11) DEFAULT NULL,
  `Work_ID` int(11) DEFAULT NULL,
  `Days` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `day_details`
--

INSERT INTO `day_details` (`Volunteer_ID`, `Work_ID`, `Days`) VALUES
(1, 1, 'mon tue wed'),
(2, 2, 'tue wed thur'),
(3, 3, 'wed thur fri'),
(4, 1, 'fri sat sun'),
(5, 2, 'sat sun mon');

-- --------------------------------------------------------

--
-- Stand-in structure for view `Schedule`
--
CREATE TABLE IF NOT EXISTS `Schedule` (
`name` varchar(30)
,`work_` varchar(50)
,`value` varchar(30)
);
-- --------------------------------------------------------

--
-- Table structure for table `shop_details`
--

CREATE TABLE IF NOT EXISTS `shop_details` (
  `Shop_ID` int(4) NOT NULL DEFAULT '0',
  `Shop_Name` varchar(30) NOT NULL,
  `Volunteer_ID` int(4) DEFAULT NULL,
  `Shop_Owner` varchar(30) NOT NULL,
  `Contact` int(10) DEFAULT NULL,
  `Place` varchar(30) NOT NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `shop_details`
--

INSERT INTO `shop_details` (`Shop_ID`, `Shop_Name`, `Volunteer_ID`, `Shop_Owner`, `Contact`,
`Place`) VALUES
(100, 'store a', 2, 'Cris', 688526, 'Dublin'),
(101, 'store b', 5, 'Leo', 688589, 'cork'),
(102, 'store c', 2, 'marius', 688547, 'Dublin'),
(103, 'store d', 5, 'darius', 688538, 'cork'),
(104, 'store e', 2, 'dan', 688565, 'Dublin');

-- --------------------------------------------------------

--
-- Table structure for table `student`
--

CREATE TABLE IF NOT EXISTS `student` (
  `School_ID` varchar(4) DEFAULT NULL,
  `Roll_NO` int(11) NOT NULL DEFAULT '0',
  `Name` varchar(30) NOT NULL,
  `Fathers_Name` varchar(30) NOT NULL,
  `DOB` date NOT NULL,
  `Class` varchar(6) NOT NULL,
  `Address` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `student`
--

INSERT INTO `student` (`School_ID`, `Roll_NO`, `Name`, `Fathers_Name`, `DOB`, `Class`,
`Address`) VALUES
('h1', 1, 'Sorcha', 'john', '2005-02-03', 'eight', 'Dublin'),
('h1', 2, 'Jen', 'james', '2007-04-08', 'six', 'Dublin'),
('h1', 3, 'Eilis', 'richard', '2005-08-04', 'seven', 'Dublin'),
('h2', 4, 'Stephen', 'dom', '2004-01-05', 'nine', 'cork'),
('h2', 5, 'Cris', 'dave', '2005-03-09', 'eight', 'cork');

-- --------------------------------------------------------

--
-- Table structure for table `teach_details`
--

CREATE TABLE IF NOT EXISTS `teach_details` (
  `Volunteer_ID` int(11) DEFAULT NULL,
  `School_ID` varchar(4) NOT NULL DEFAULT '',
  `School_Name` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
```

```
-- Dumping data for table `teach_details`
--

INSERT INTO `teach_details` (`Volunteer_ID`, `School_ID`, `School_Name`) VALUES
(3, 'h1', 'hall 1'),
(3, 'h2', 'hall 2');

-- --------------------------------------------------------

--
-- Table structure for table `team_members`
--

CREATE TABLE IF NOT EXISTS `team_members` (
  `Volunteer_ID` int(11) NOT NULL DEFAULT '0',
  `Work_id` int(11) DEFAULT NULL,
  `Name` varchar(30) NOT NULL,
  `DOB` date NOT NULL,
  `Gender` varchar(8) DEFAULT NULL,
  `Email_id` varchar(50) DEFAULT NULL,
  `ContactNo` int(11) DEFAULT NULL,
  `College` varchar(60) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `team_members`
--

INSERT INTO `team_members` (`Volunteer_ID`, `Work_id`, `Name`, `DOB`, `Gender`, `Email_id`,
`ContactNo`, `College`) VALUES
(1, 1, 'Yash', '1997-09-15', 'male', 'yash@gmail.com', 851074416, 'TCD'),
(2, 2, 'Ciara', '1997-09-15', 'Female', 'ciara@gmail.com', 851074444, 'UCD'),
(3, 3, 'Jenny', '1997-09-15', 'Female', 'jenny@gmail.com', 851071234, 'TCD'),
(4, 1, 'Aniket', '1997-09-15', 'Female', 'aniket@gmail.com', 851075678, 'TCD'),
(5, 2, 'Sarah', '1997-09-15', 'Female', 'sarah@gmail.com', 851079101, 'UCD');

--
-- Triggers `team_members`
--
DELIMITER //
CREATE TRIGGER `tmins` BEFORE UPDATE ON `team_members`
 FOR EACH ROW BEGIN
    IF NEW.Volunteer_ID = 0 THEN
       SET NEW.Volunteer_ID = 0;

   END IF;
   END
//
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `work_master`
--
```

```
CREATE TABLE IF NOT EXISTS `work_master` (
  `Work_id` int(11) NOT NULL DEFAULT '0',
  `Work_` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `work_master`
--

INSERT INTO `work_master` (`Work_id`, `Work_`) VALUES
(1, 'anti-begging'),
(2, 'employment generation'),
(3, 'teaching');

-- --------------------------------------------------------

--
-- Structure for view `Schedule`
--
DROP TABLE IF EXISTS `Schedule`;

CREATE ALGORITHM=UNDEFINED DEFINER=`pandeyy`@`localhost` SQL SECURITY
DEFINER VIEW `Schedule` AS select `team_members`.`Name` AS `name`,`work_master`.`Work_`
AS `work_`,`day_details`.`Days` AS `value` from ((`team_members` join `work_master`) join
`day_details`);

--
-- Indexes for dumped tables
--

--
-- Indexes for table `Contact`
--
ALTER TABLE `Contact`
 ADD UNIQUE KEY `ContactNo` (`ContactNo`);

--
-- Indexes for table `day_details`
--
ALTER TABLE `day_details`
 ADD KEY `Volunteer_ID` (`Volunteer_ID`), ADD KEY `Work_ID` (`Work_ID`);

--
-- Indexes for table `shop_details`
--
ALTER TABLE `shop_details`
 ADD PRIMARY KEY (`Shop_ID`), ADD UNIQUE KEY `Contact` (`Contact`), ADD KEY
`Volunteer_ID` (`Volunteer_ID`);

--
-- Indexes for table `student`
--
ALTER TABLE `student`
 ADD PRIMARY KEY (`Roll_NO`), ADD KEY `School_ID` (`School_ID`);
```

```
--
-- Indexes for table `teach_details`
--
ALTER TABLE `teach_details`
 ADD PRIMARY KEY (`School_ID`), ADD KEY `Volunteer_ID` (`Volunteer_ID`);

--
-- Indexes for table `team_members`
--
ALTER TABLE `team_members`
 ADD PRIMARY KEY (`Volunteer_ID`), ADD UNIQUE KEY `Email_id` (`Email_id`), ADD
UNIQUE KEY `ContactNo` (`ContactNo`), ADD KEY `Work_id` (`Work_id`);

--
-- Indexes for table `work_master`
--
ALTER TABLE `work_master`
 ADD PRIMARY KEY (`Work_id`);

--
-- Constraints for dumped tables
--

--
-- Constraints for table `day_details`
--
ALTER TABLE `day_details`
ADD CONSTRAINT `day_details_ibfk_1` FOREIGN KEY (`Volunteer_ID`) REFERENCES
`team_members` (`Volunteer_ID`),
ADD CONSTRAINT `day_details_ibfk_2` FOREIGN KEY (`Work_ID`) REFERENCES
`work_master` (`Work_id`);

--
-- Constraints for table `shop_details`
--
ALTER TABLE `shop_details`
ADD CONSTRAINT `shop_details_ibfk_1` FOREIGN KEY (`Volunteer_ID`) REFERENCES
`team_members` (`Volunteer_ID`);

--
-- Constraints for table `student`
--
ALTER TABLE `student`
ADD CONSTRAINT `student_ibfk_1` FOREIGN KEY (`School_ID`) REFERENCES
`teach_details` (`School_ID`);

--
-- Constraints for table `teach_details`
--
ALTER TABLE `teach_details`
ADD CONSTRAINT `teach_details_ibfk_1` FOREIGN KEY (`Volunteer_ID`) REFERENCES
`team_members` (`Volunteer_ID`);

--
```

```
-- Constraints for table `team_members`
--
ALTER TABLE `team_members`
ADD CONSTRAINT `team_members_ibfk_1` FOREIGN KEY (`Work_id`) REFERENCES
`work_master` (`Work_id`);
```

Procedures and triggers that were not implemented-

Procedure for insertion in team member

```
CREATE OR REPLACE PROCEDURE INSERTmember (
v_id _team.volunteer_id%TYPE,
w_id _team.work_id%TYPE,
m_name _TEAM.NAME%type,
datebirth _team.dob%TYPE,
m_gender _team.gender%TYPE,
e_id _team.email_id%TYPE,
contact _team.contactno%TYPE,
coll _TEAM.college%TYPE)
IS
BEGIN
INSERT INTO _team_members VALUES(v_id,w_id,'m_name',to_date('datebirth','mm-dd-
yyyy'),'m_gender','e_id',contact,coll);
COMMIT;
END;
/
```

Procedure for insertion in Student

```
CREATE OR REPLACE PROCEDURE INSERTstudent (
s_id STUDENT.SCHOOL_ID%type,
r_no STUDENT.ROLL_NUMBER%type,
s_name STUDENT.NAME%type,
f_name STUDENT.FATHERS_NAME%type,
birthdate STUDENT.DOB%type,
```

```
s_class STUDENT.CLASS%type,
addr STUDENT.ADDRESS%type
)
IS
BEGIN
INSERT INTO STUDENT VALUES('s_id','r_no','s_name','f_name',to_date('birthdate','mm-dd-yyyy'),'s_class','addr');
COMMIT;
END;
/
```

Procedure for inserting in day_details

```
CREATE OR REPLACE PROCEDURE INSERTdays (
v_id DAY_DETAILS.VOLUNTEER_ID%type,
noday DAY_DETAILS.DAYS%type
)
IS
BEGIN
INSERT INTO day_details VALUES(v_id,'noday');
COMMIT;
END;
/
```

Procedure for inserting into teach_details

```
CREATE OR REPLACE PROCEDURE INSERTschool (
v_id TEACH_DETAILS.VOLUNTEER_ID%type,
s_id TEACH_DETAILS.SCHOOL_ID%type,
s_name TEACH_DETAILS.SCHOOL_NAME%type
)
```

```
IS
BEGIN
INSERT INTO teach_details VALUES(v_id,'s_id','s_name');
COMMIT;
END;
/
```

Procedure for inserting into shop details

```
CREATE OR REPLACE PROCEDURE INSERTshop (
v_id SHOP_DETAILS.VOLUNTEER_ID%type,
s_id SHOP_DETAILS.SHOP_ID%type,
s_name SHOP_DETAILS.SHOP_NAME%type,
s_owner SHOP_DETAILS.SHOP_OWNER%type,
cont SHOP_DETAILS.CONTACT%type,
loc SHOP_DETAILS.PLACE%type
)
IS
BEGIN
INSERT INTO SHOP_DETAILS VALUES('s_id','s_name',v_id,'s_owner',cont,'loc');
COMMIT;
END;
/
```

Procedure for inserting in contact

```
CREATE OR REPLACE PROCEDURE INSERTitem (
cname CONTACT.NAME%type,
cper CONTACT.CONTACT_PERSON%type,
cno CONTACT.CONTACTNO%type,
afield CONTACT.ACTIVITY_FIELD%type
```

)

IS

BEGIN

INSERT INTO CONTACT VALUES('cname','cper',cno,’afield’);

COMMIT;

END;

/

Procedure to delete from team

CREATE OR REPLACE PROCEDURE DELETE _member (

p_name IN _team.NAME%TYPE)

IS

BEGIN

DELETE FROM _team WHERE NAME=p_name;

END;

/

Procedure to update team

CREATE OR REPLACE PROCEDURE UPDATE _team (

v_id _TEAM.VOLUNTEER_ID%type,

w_id _TEAM.WORK_ID%type,

uname _TEAM.NAME%type,

dabirth _TEAM.DOB%type,

gen _TEAM.GENDER%type,

email _TEAM.EMAIL_ID%type,

cont _TEAM.CONTACTNO%type,

coll _TEAM.COLLEGE%type)

IS

BEGIN

```
UPDATE  _team SET NAME=uname,
GENDER=gen,WORK_ID=w_id,dob=dabirth,email_id=email,contactno=cont,college=coll
WHERE volunteer_id=v_id;

COMMIT;

END;

/
```

Procedure to show team details

```
CREATE OR REPLACE PROCEDURE team_details

IS

CURSOR team_cur IS

SELECT name, volunteer_id, work_id,email_id FROM  _team;

team_rec team_cur%rowtype;

BEGIN

open team_cur;

FOR team_rec in team_cur

LOOP

dbms_output.put_line('name: '||team_rec.name || ' volunteer id: ' ||team_rec.volunteer_id || '
work id: ' ||team_rec.work_id||' email id: '||team_rec.email_id);

END LOOP;

close team_cur;

END;

/
```

Exception where volunteer id is asked and name and email address are returned if that exists ,else different errors are shown

```
SET SERVEROUTPUT ON;

DECLARE

  k_id  _TEAM.VOLUNTEER_ID%type := &kt_id;

  k_name  _TEAM.NAME%type;

  k_email  _TEAM.EMAIL_ID%type;


  -- user defined exception

  ex_invalid_id  EXCEPTION;
```

```
BEGIN
  IF k_id <= 0 THEN
    RAISE ex_invalid_id;
  ELSE
    SELECT  name, email_id INTO  k_name, k_email
    FROM  _team
    WHERE VOLUNTEER_ID = k_id;

    DBMS_OUTPUT.PUT_LINE ('Name: '||  k_name);
    DBMS_OUTPUT.PUT_LINE ('Email Address: ' || k_email);
  END IF;
EXCEPTION
  WHEN ex_invalid_id THEN
    dbms_output.put_line('ID must be greater than zero!');
  WHEN no_data_found THEN
    dbms_output.put_line('No such   team member!');
  WHEN others THEN
    dbms_output.put_line('Error!');
END;
/
```

Triggers

Trigger of contact table after insertion

```
create or replace trigger contprim
after insert on contact
for each row
begin
dbms_output.put_line('A row has been inserted');
end;
/
```

Trigger of shop_details table after insertion

```
create or replace trigger shopins
after insert on shop_details
for each row
begin
dbms_output.put_line('A row has been inserted');
end;
/
```

Trigger of teach_details table after insertion

```
create or replace trigger teachins
after insert on teach_details
for each row
begin
dbms_output.put_line('A row has been inserted');
end;
/
```

Trigger of team table after insertion

```
create or replace trigger in
after insert on _team
for each row
begin
dbms_output.put_line('A row has been inserted');
end;
/
```

Trigger of student table after insertion

```
create or replace trigger studentin
after insert on student
```

```
for each row
begin
dbms_output.put_line('A row has been inserted');
end;
/
```

Trigger of day_details table after insertion

```
create or replace trigger workdayin
after insert on day_details
for each row
begin
dbms_output.put_line('A row has been inserted');
end;
/
```

Trigger of contact table after updation

```
create or replace trigger contupd
after update on contact
for each row
begin
dbms_output.put_line('A row has been updated');
end;
/
```

Trigger of shop_details table after updation

```
create or replace trigger shopupd
after update on shop_details
for each row
begin
dbms_output.put_line('A row has been updated');
end;
/
```

Trigger of teach_details table after updation

```
create or replace trigger teachupd
after update on teach_details
for each row
begin
dbms_output.put_line('A row has been updated');
end;
/
```

Trigger of _team table after updation

```
create or replace trigger upd
after update on _team
for each row
begin
dbms_output.put_line('A row has been updated');
end;
/
```

Trigger of student table after updation

```
create or replace trigger studentupd
after update on student
for each row
begin
dbms_output.put_line('A row has been updated');
end;
/
```

Trigger of day_details table after updation

```
create or replace trigger workdayupd
after update on day_details
for each row
begin
```

```
dbms_output.put_line('A row has been updated');
end;
```