Hemin Patel I hp465 I Project Submission

```
# Part 1, FDs in Penna

-- a) Specify functional dependencies in Penna
-- FD 1: precinct-> state, geo, locality, filestamp

SELECT DISTINCT 'True' as 'nonempty'
From testDB.penna p1
Where Exists (SELECT * from testDB.penna t1, testDB.penna t2 where t1.precinct =
t2.precinct and (t1.state != t2.state or t1.geo != t2.geo or t1.locality != t2.locality or
t1.filestamp != t2.filestamp))
UNION
SELECT DISTINCT 'False' as 'nonempty'
FROM testDB.penna p1
Where not Exists (SELECT * from testDB.penna t1, testDB.penna t2 where t1.precinct =
t2.precinct and (t1.state != t2.state or t1.geo != t2.geo or t1.locality != t2.locality or
t1.filestamp != t2.filestamp))

-- FD 2: precinct, Timestamp -> totalvotes, Biden, Trump, state, geo, locality, filestamp

SELECT DISTINCT 'True' as 'nonempty'
From testDB.penna p1
Where not Exists (SELECT * from testDB.penna t1, testDB.penna t2 where (t1.precinct
= t2.precinct) and (t1.Timestamp = t2.Timestamp) and (t1.locality != t2.locality) and
(t1.totalvotes != t2.totalvotes) and (t1.Biden != t2.Biden) and (t1.Trump != t2.Trump) and
(t1.filestamp != t2.filestamp))
UNION
SELECT DISTINCT 'False' as 'nonempty'
FROM testDB.penna p1
Where Exists (SELECT * from testDB.penna t1, testDB.penna t2 where (t1.precinct =
t2.precinct) and (t1.Timestamp = t2.Timestamp) and (t1.locality != t2.locality) and
(t1.totalvotes != t2.totalvotes) and (t1.Biden != t2.Biden) and (t1.Trump != t2.Trump) and
(t1.filestamp != t2.filestamp))

#FDs
-- precinct-> state, geo, locality, filestamp
-- precinct, Timestamp -> totalvotes, Biden, Trump, state, geo, locality, filestamp

-- b) Is Penna in BCNF?
-- Location Table
Create table testDB.Location
As Select distinct p.precinct, p.locality, p.geo, p.state From testDB.penna p

-- Votes Table
Create table testDB.Votes
As Select distinct p.precinct, p.totalvotes, p.Biden, p.Trump, p.Timestamp, p.filestamp
```

```
From testDB.penna p


--
--------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------
# Part 2
-- 1) The Precinct
-- a) Winner: who won? Show % of totalvotes to winner. Show final num of totalvotes in
this precinct.
delimiter $$
create procedure WinnerPrecinct(IN precinctName VARCHAR(255))
        BEGIN
                select
                        if(sum(Trump) > sum(Biden), 'Trump', 'Biden') AS Winner,
                        if(sum(Trump) > sum(Biden), sum(Trump), sum(Biden)) AS
totalVotes,
                        if(sum(Trump) > sum(Biden), concat(round(100*sum(Trump)/
sum(totalvotes),2), "%"), concat(round(100*sum(Biden)/sum(totalvotes),2), "%")) AS
Percentage
                from penna
                where precinct = precinctName;
END $$
DELIMITER ;

-- b) RankALL: numerical rank of this precinct in terms of the number of total votes it
received (at the last timestamp) among all precincts in the database
drop procedure if exists RankPrecinct;
delimiter $$
CREATE PROCEDURE RankPrecinct (IN precinctName VARCHAR(255), OUT
rankNum int)
Begin
        Declare MaxTime text;
   Select Max(Timestamp) into MaxTime from penna;
   Drop table if exists RankPrecinct;
   Create table testDB.RankPrecinct as
                SELECT precinct, totalvotes, Timestamp,
                rank() OVER(ORDER BY totalvotes DESC) AS `ranks`
                FROM penna
                where Timestamp = MaxTime; -- this is my latest timestamp in my
Penna.csv (the format also changed when I reimported.. nvm i reimported AGAIN)

      Select ranks into rankNum
      from RankPrecinct
      where precinct= precinctName;
        End $$
DELIMITER ;
```

```
-- c) RankCounty: numerical rank of this precinct in terms of the number of total votes it
received (at the last timestamp) among all precincts in the county this precinct belongs
to
drop procedure if exists RankCounty;
delimiter $$
CREATE PROCEDURE RankCounty (IN precinctName VARCHAR(255), OUT rankNum
int)
Begin
        Declare MaxTime text;
   Select Max(Timestamp) into MaxTime from penna;
   Drop table if exists RankCounty;
   Create table testDB.RankCounty as
                SELECT precinct, locality, totalvotes, Timestamp,
                rank() OVER(Partition BY locality ORDER BY totalvotes DESC) AS
`ranks`
                FROM penna
                where Timestamp = MaxTime; -- this is my latest timestamp in my
Penna.csv (the format also changed when I reimported.. nvm i reimported AGAIN)

      Select ranks into rankNum
      from RankCounty
      where precinct= precinctName;
        End $$
DELIMITER ;

-- d) PlotPrecinct: plot three atrributes on excel, on the doc pdf.
-- Select Timestamp, Biden, Trump, totalvotes
-- from penna
-- where precinct = '02-01';

-- e)  EarliestPrecinct(vote_count) Show the first precinct to reach vote_count (input),
totalvotes, timestamp when it occurred.
--    If multiple precincts reached input @ timestamp, return precinct w/ most totalvotes.
DROP PROCEDURE IF EXISTS EarliestPrecinct;
delimiter $$
Create procedure EarliestPrecinct(in vote_count VARCHAR(255))
Begin
declare counter int;
        Select COUNT(p1.precinct) into counter
        from Penna p1
        where p1.totalvotes >= vote_count and Timestamp = (select min(Timestamp)
        from Penna p2 where p2.totalvotes >= vote_count);
        if counter > 1 THEN
                select distinct p1.Timestamp, p1.precinct, p1.totalvotes
                from Penna p1
                where p1.totalvotes >= vote_count and Timestamp = (select
min(Timestamp)
                from Penna p2 where p2.totalvotes >= vote_count)
      order by totalvotes DESC LIMIT 1;
        elseif counter = 1 THEN
                select distinct p1.Timestamp, p1.precinct, p1.totalvotes
                from Penna p1
                where p1.totalvotes >= vote_count and Timestamp = (SELECT
min(Timestamp)
                from Penna p2 where p2.totalvotes >= vote_count);
        END IF;
END$$
DELIMITER ;

-- 2) The Candidates
-- a) PrecinctsWon: lists precincts of winner, vote difference, totalvotes canidate got
drop procedure if exists PrecinctsWon;
delimiter $$
create procedure PrecinctsWon(IN candidate VARCHAR(255))
        BEGIN
                IF(candidate = 'Biden') THEN
                        Select distinct precinct, Biden, Biden-Trump AS Difference
                        from testDB.penna
                        where Biden > Trump
                        order by Difference DESC;
                END IF;
                IF(candidate = 'Trump') THEN
                        Select distinct precinct, Trump, Trump-Biden AS Difference
                        from testDB.penna
                        where Trump > Biden
                        order by Difference DESC;
                END IF;
        END $$
DELIMITER ;

-- b) PrecinctsWonCount(candidate) Show the count of how many precincts the
candidate won.
drop procedure if exists PrecinctsWonCount;
delimiter $$
create procedure PrecinctsWonCount(IN candidate VARCHAR(255))
        BEGIN
                IF(candidate = 'Biden') THEN
                        Select COUNT(*) OVER () as precinct
                        from Penna
                        Group by precinct
                        Having Max(Biden) > Max(Trump) Limit 1;
                END IF;
                IF(candidate = 'Trump') THEN
```

```
                        Select COUNT(*) OVER () as precinct
                        from Penna
                        Group by precinct
                        Having Max(Trump) > Max(Biden) Limit 1;
                END IF;
        END $$
DELIMITER ;

-- c) PrecinctsFullLead(candidate) List precincts which the candidate held a lead for at
every timestamp
-- Biden, return precinct that has Biden leading in that precinct for all timestampos
drop procedure if exists PrecinctsFullLead;
delimiter $$
create procedure PrecinctsFullLead(IN candidate VARCHAR(255))
        BEGIN
                IF(candidate = 'Biden') THEN
                        Select distinct precinct
                        from Penna
                        Where (Biden) > (Trump)
         Group by precinct, Timestamp;
                END IF;
                IF(candidate = 'Trump') THEN
                        Select distinct precinct
                        from Penna
         Where (Trump) > (Biden)
                        Group by precinct, Timestamp;
                END IF;
        END $$
DELIMITER ;

-- d) PlotCandidate(candidate) Show a timeseries plot for the candidate, plot number of
votes that candidate received at each timestamp
-- Select distinct Timestamp, Sum(Biden) as Biden, Sum(Trump) as Trump
-- From penna
-- group by Timestamp
-- order by Timestamp; # export this and plot, submit this graph



-- e) PrecinctsWonTownships - uses all the township precincts. return the name of the
winning candidate, vote difference, the total votes of each candidate

# My previous code, thought had to list winner for each precinct name. HAVE TO list for
OVERALL TOWNSHIP
drop procedure if exists PrecinctsWonTownships;
delimiter $$
create procedure PrecinctsWonTownships()
```

```
BEGIN
   Select Sum(Biden), Sum(Trump), abs(Sum(Biden)-Sum(Trump)) as Difference,
sum(Biden) > sum(Trump) as "Trump: 0 Biden: 1"
        From (Select Timestamp from Penna ORDER BY Timestamp desc LIMIT 1) P1,
Penna P2
   WHERE P2.timestamp = P1.timestamp AND precinct LIKE "%Township%";
END $$
DELIMITER ;

-- 3) The Timestamp
-- a) TotalVotes(timestamp, category) This stored procedure will take a category as input
in the form of either ALL, Trump or Biden.
-- show an list of precincts by either totalvote, Trump, or Biden (based on the input
category) at that Timestamp.
-- 2020-11-04 03:58:36
drop procedure if exists TotalVotes;
delimiter $$
create procedure TotalVotes(in Timestamp text, IN category VARCHAR(255))
        BEGIN
                IF(category = 'Biden') THEN
                        Select distinct precinct
                        from Penna
                        Group by precinct;
                END IF;
                IF(category = 'Trump') THEN
                        Select distinct precinct
                        from Penna
                        Group by precinct;
                END IF;
                IF(category = 'totalvotes') THEN
                        Select distinct precinct
                        from Penna
                        Group by precinct;
                END IF;
        END $$
DELIMITER ;

-- b) GainDelta(timestamp) Using the timestamp preceding the input timestamp, return
DELTA representing the amount of time passed since that preceding timestamp as well
as GAIN,
-- the number of additional votes gained since that preceding timestamp. Also return the
ratio GAIN/DELTA,
# '2020-11-04 01:15:49' # lowest timestamp in my penna
DROP PROCEDURE IF EXISTS GainDelta;
delimiter $$
Create procedure GainDelta(in TimestampIn text)
Begin
```

```
Declare MinVotes int;
Declare MinTime text;

        Drop table if exists Times;
        Create table testdb.Times as (select Timestamp, sum(totalvotes) as totalvotes
from penna Group BY Timestamp Order by Timestamp);

   Select min(totalvotes) into MinVotes from Times;
   Select min(Timestamp) into MinTime from Times;

                select abs(TIMESTAMPDIFF(second, TimestampIn, MinTime)) as
"Seconds",
                abs(MinVotes - totalvotes) as Difference,
      abs(MinVotes - totalvotes)/abs(TIMESTAMPDIFF(second, TimestampIn, MinTime))
as "Gain/Delta"
                FROM Times
      where Timestamp = TimestampIn;
END $$
DELIMITER ;

-- c)  RankTimestamp() Rank all timestamps by the above GAIN/DELTA ratio in
descending order
drop procedure if exists RankTimestamp;
delimiter $$
CREATE PROCEDURE RankTimestamp(in TimestampIn text) #not sure if input is
needed
Begin
        Declare MinVotes int;
        Declare MinTime text;
        Select min(totalvotes) into MinVotes from Times;
   Select min(Timestamp) into MinTime from Times;

   Drop table if exists RankTimestamp;
   Create table testDB.RankTimestamp as
   (Select Timestamp, abs(MinVotes- totalvotes)/abs(TIMESTAMPDIFF(second,
TimestampIn, MinTime)) as "GainDelta"
   From Times Order by Timestamp);

        SELECT Timestamp, GainDelta,
                rank() OVER(ORDER BY GainDelta DESC) AS `Rank`
                FROM RankTimestamp;
        End $$
DELIMITER ;

-- d) VotesPerDay(day) Show votes for Biden, Trump, and total votes that occurred on
just day (i.e., day should be an input between 03 and 11 corresponding to the day of the
timestamp)
```

```
drop procedure if exists VotesPerDay;
delimiter $$
CREATE PROCEDURE VotesPerDay (IN `day` int)
Begin
   drop table if exists DayTime;
        Create table testDB.DayTime as
        select date(substring(timestamp from 1 for 10)) as STime, Biden, Trump,
totalvotes
   from penna;

   Select distinct Biden, Trump, totalvotes
        From DayTime
   WHERE EXTRACT(day FROM Stime) = `day`;
        End $$
DELIMITER ;

-- 4) Suspicious or Interesting Data
-- When I completed Part 3a, I realized that Biden + Trump votes did not equal the
totalvotes count, totalvotes always had more. When I was solving that question, I
created a simple query to see if it macthed and it did not.
-- I assume that maybe totalvotes includes 3rd party candidates, but I believe the
professor would've mentioned that. Or there might've been an error in totalvotes, which
is quite suspicious because then how can we know if the data is accurate.
Select precinct, totalvotes,
(Biden + Trump) as "Biden & Trump" , (totalvotes - (Biden + Trump)) as "Difference in
Vote Count"
From testDB.penna

--
---------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------
# Part 3
-- a) The sum of votes for Trump and Biden cannot be larger than totalvotes
SELECT DISTINCT 'True' as 'nonempty'
From testDB.penna p
Where Exists (SELECT * from testDB.penna where ((Biden + Trump) < totalvotes))
UNION
SELECT DISTINCT 'False' as 'nonempty'
FROM testDB.penna p2
Where not Exists (SELECT * from testDB.penna where ((Biden + Trump) < totalvotes))

-- b) There cannot be any tuples with timestamps later than Nov 11 and earlier than
Nov3
SELECT DISTINCT 'True' as 'nonempty'
From testDB.penna
Where Not Exists (SELECT * from testDB.penna where (Timestamp like
"%2020-11-03%" and "%2020-11-11%"))
```

```
UNION
SELECT DISTINCT 'False' as 'nonempty'
FROM testDB.penna
Where Exists (SELECT * from testDB.penna where (Timestamp like "%2020-11-03%"
and "%2020-11-11%"))

-- c) Neither totalvotes, Trump's votes nor Biden's votes for any precinct and at any
timestamp after 2020-11-05 00:00:00
--    will be smaller than the same attribute at the timestamp 2020-11-05 00:00:00 for
that precinct.
Select distinct 'True' as 'nonempty'
from penna p
Where Not Exists(select p.totalvotes, p.Biden, p.Trump, p.precinct, p.Timestamp
        from penna p, (select totalvotes, Biden, Trump, precinct, Timestamp from penna
where Timestamp = '2020-11-05 00:00:00') l2
        Where p.Timestamp > '2020-11-05 00:00:00' AND p.totalvotes < l2.totalvotes
AND p.Biden < l2.Biden AND p.Trump < l2.Trump)
UNION
Select distinct 'False' as 'nonempty'
from penna p2
Where Exists(select p.totalvotes, p.Biden, p.Trump, p.precinct, p.Timestamp
        from penna p, (select totalvotes, Biden, Trump, precinct, Timestamp from penna
where Timestamp = '2020-11-05 00:00:00') l2
        Where p.Timestamp > '2020-11-05 00:00:00' AND p.totalvotes < l2.totalvotes
AND p.Biden < l2.Biden AND p.Trump < l2.Trump);
--
---------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------
-- Part 4
-- a) create "Modification" stored procedure, users can modify (Insert/Update/Delete)
any table in my database. Success: show success message, Fail: Violates foreign key

-- 4.1 Triggers and Update driven Stored Procedures
-- a) For each table in database, create three log tables and three triggers. These tables
will be called Updated Tuples, Inserted Tuples and Deleted Tuples.
--    All three tables should have the same schema as the original table and should store
any
--    tuples which were updated (store them as they were before the update), any tuples
which
--    were inserted,  and any tuples which were deleted in their corresponding tables.
--    The triggers should populate these tables upon each update/insertion/deletion.
There will be one
--    trigger for the update operation, one trigger for the insert operation and one trigger
for the delete operation.

# Insert
Create table Inserted_Tuples (precinct VARCHAR(255) NOT NULL, Timestamp text,
totalvotes INT, Biden INT, Trump INT, filestamp VARCHAR(255));
delimiter $$
-- drop trigger Inserted_log
Create Trigger Inserted_log
        After insert on Votes
        For each row
        BEGIN
                Insert into Inserted_Tuples (precinct, Timestamp, totalvotes, Biden, Trump,
filestamp)
                values(new.precinct, new.Timestamp, new.totalvotes, new.Biden,
new.Trump, new.filestamp);
        END$$
DELIMITER ;

# Test
Insert into Location (precinct, locality, geo, state) values('Rutgers Uni', 'Middlesex', 'New
Brunswick', 'NJ');
Insert into Votes (precinct, Timestamp, totalvotes, Biden, Trump, filestamp)
values('Rutgers Uni', '2021-11-30 20:23:10', 500, 250, 250,
'NOVEMBER_30_2022_000000.json');

-- Delete from Location where precinct like "Rutgers Uni"
-- Delete from Votes where precinct like "Rutgers Uni"

select * from testDB.Votes order by Timestamp desc Limit 1;
SELECT * FROM testDB.Inserted_Tuples;

# Update
Create table Updated_Tuples (precinct VARCHAR(255) NOT NULL, Timestamp text,
totalvotes int, Biden int, Trump int, filestamp VARCHAR(255));

delimiter $$
-- drop trigger Updated_log;
Create trigger Updated_log
        Before update on Votes
        for each row
        BEGIN
                Insert into Updated_Tuples (precinct, Timestamp, totalvotes, Biden,
Trump, filestamp)
                values(old.precinct, old.Timestamp, old.totalvotes, old.Biden, old.Trump,
old.filestamp);
        END$$
DELIMITER ;

# Test
-- Set foreign_key_checks=0
update Votes set totalvotes = 800, Biden = 400, Trump = 400 WHERE precinct =
'Rutgers Uni';
select * from testDB.Votes order by Timestamp desc Limit 1;
SELECT * FROM testDB.Updated_Tuples;

# Delete
Create table Deleted_Tuples (precinct VARCHAR(255) NOT NULL, Timestamp text,
totalvotes int, Biden int, Trump int, filestamp VARCHAR(255));
-- drop table Deleted_Tuples;

drop trigger Deleted_log;
delimiter $$
CREATE TRIGGER Deleted_log
	After delete on Votes
	For each row
	BEGIN
		insert into Deleted_Tuples (precinct, Timestamp, totalvotes, Biden, Trump,
filestamp)
		values(old.precinct, old.Timestamp, old.totalvotes, old.Biden, old.Trump,
old.filestamp);
	END$$
DELIMITER ;

# Test
delete from testDB.Votes where precinct = 'Rutgers Uni';
select * from testDB.Votes order by Timestamp desc;
SELECT * FROM Deleted_Tuples;

-- 4.2 MoveVotes(Precinct, Timest, CoreCandidate, Number_of_Moved_Votes)

drop procedure if exists MoveVotes;
Delimiter $$
	create procedure MoveVotes(in CorePrecinct VARCHAR(255), in Timest
VARCHAR(255), in CoreCandidate VARCHAR(255), IN Number_of_Moved_Votes
VARCHAR(255))
	BEGIN
		declare condition1 VARCHAR(255);
		declare condition2 VARCHAR(255);
		declare condition3 VARCHAR(255);
		declare condition4 VARCHAR(255);
		declare condition5 VARCHAR(255);
		declare condition6 VARCHAR(255);
		declare removing int;
		declare adding int;

		select if(count(distinct (precinct)) = 1, 'Exists', 'Not Exists') into condition1
from penna where precinct = CorePrecinct;
		select if(Timest in (select distinct Timestamp from penna), 'Exists', 'Not
Exists') INTO condition2;
                select if((Biden > Number_of_Moved_Votes), 'True','False') into condition3
from penna where Timestamp = Timest and precinct = CorePrecinct;
                select if((Trump > Number_of_Moved_Votes), 'True','False') into
condition4 from penna where Timestamp = Timest and precinct = CorePrecinct;

                IF (condition1 = 'Not Exists')
                        then select     'Unknown Precinct' as 'Message';
                ELSEIF (condition2 = 'Not Exists')
                        then select     'Unknown Timestamp' as 'Message';
                ELSEIF (CoreCandidate <> 'Trump') and (CoreCandidate <> 'Biden')
                        then select 'Wrong Candidate' as 'Message';
                ELSEIF (condition3 = 'False' and (CoreCandidate = 'Biden'))
                        then select 'Not enough votes - Biden' AS 'Message';
                ELSEIF (condition4 = 'False' and (CoreCandidate = 'Trump'))
                        then select 'Not enough votes - Trump' AS 'Message';
                ELSEIF (condition3 = 'True' and (CoreCandidate = 'Biden')) THEN
                        select Biden into removing from penna where Timestamp = Timest
and precinct = CorePrecinct;
                        select Trump into adding from penna where Timestamp = Timest
and precinct = CorePrecinct;
                        update penna set Biden = removing - Number_of_Moved_Votes,
Trump = adding + Number_of_Moved_Votes
                        where Timestamp >= Timest AND precinct = CorePrecinct;
                        select * from penna where Timestamp >= Timest AND precinct =
CorePrecinct;
                ELSEIF (condition4 = 'True' and (CoreCandidate = 'Trump')) THEN
                        select Trump into removing from penna where Timestamp = Timest
and precinct = CorePrecinct;
                        select Biden INTO adding from penna where Timestamp = Timest
and precinct = CorePrecinct;
                        update penna set Trump = removing - Number_of_Moved_Votes,
Biden = adding + Number_of_Moved_Votes
                   where Timestamp >= Timest AND precinct = CorePrecinct;
                   select * from penna where Timestamp >= Timest and precinct =
CorePrecinct;
                END IF;
        END $$
DELIMITER ;

-- Test
call MoveVotes("005 ATGLEN","2020-11-04 08:31:05","Biden","200");
call MoveVotes("Seton Hall Uni",'2020-11-04 08:31:05','Biden','200');
call MoveVotes('005 ATGLEN','2020-11-17 23:31:18','Biden','100');

call MoveVotes('Adams Township - Elton Voting Precinct','2020-11-10
23:31:18','Trump','100');
```

```
update Penna SET Biden = 166, Trump = 339 where Timestamp >= '2020-11-10
23:31:18' and precinct = 'Adams Township - Elton Voting Precinct';
Select * from penna where Timestamp like "2020-11-10 23:31:18" and precinct =
'Adams Township - Elton Voting Precinct'
```