**DBMS**

**By:**

**Keerthana A**

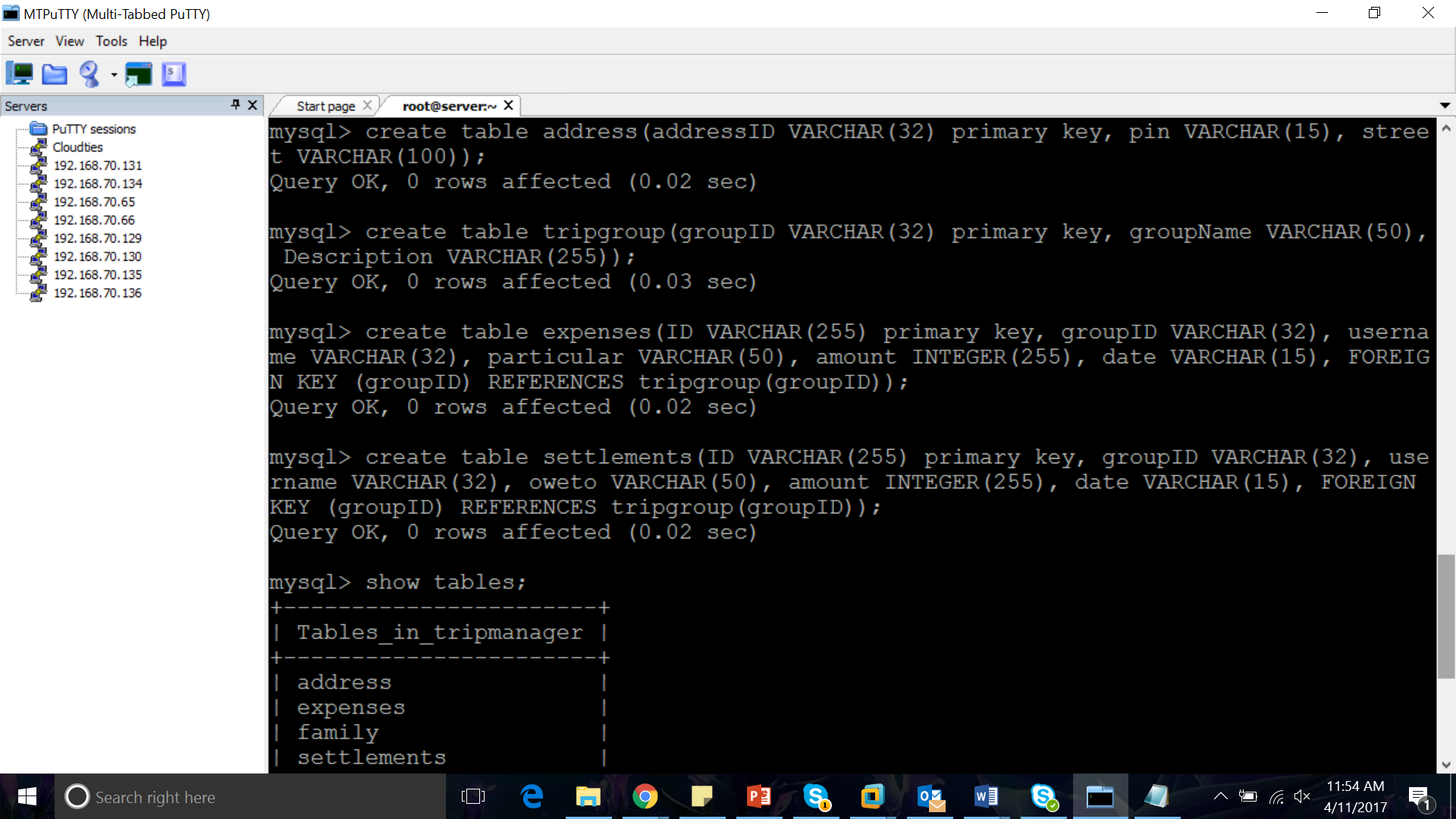
**Documentation to read:**

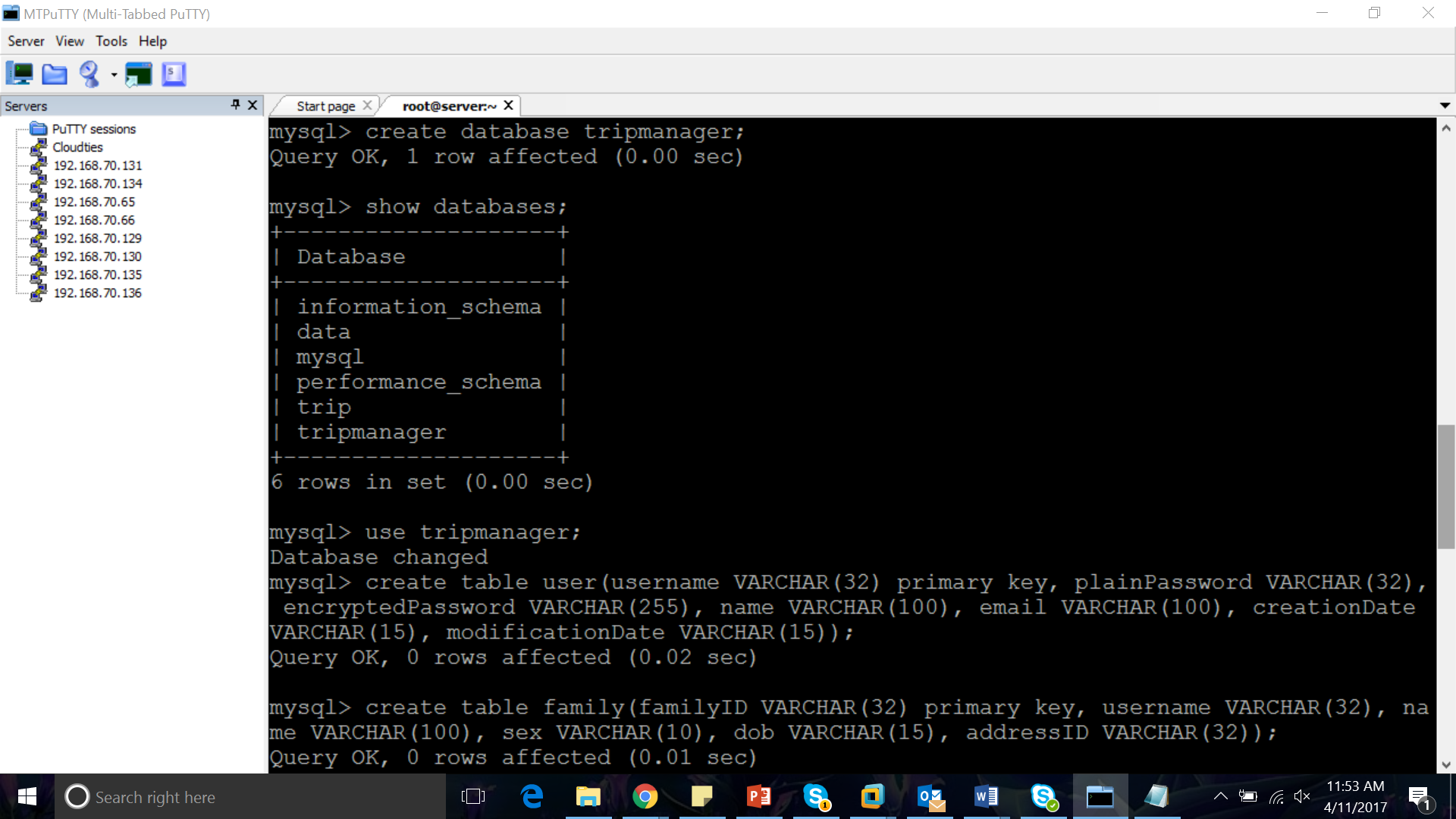
<https://dev.mysql.com/doc/refman/5.7/>

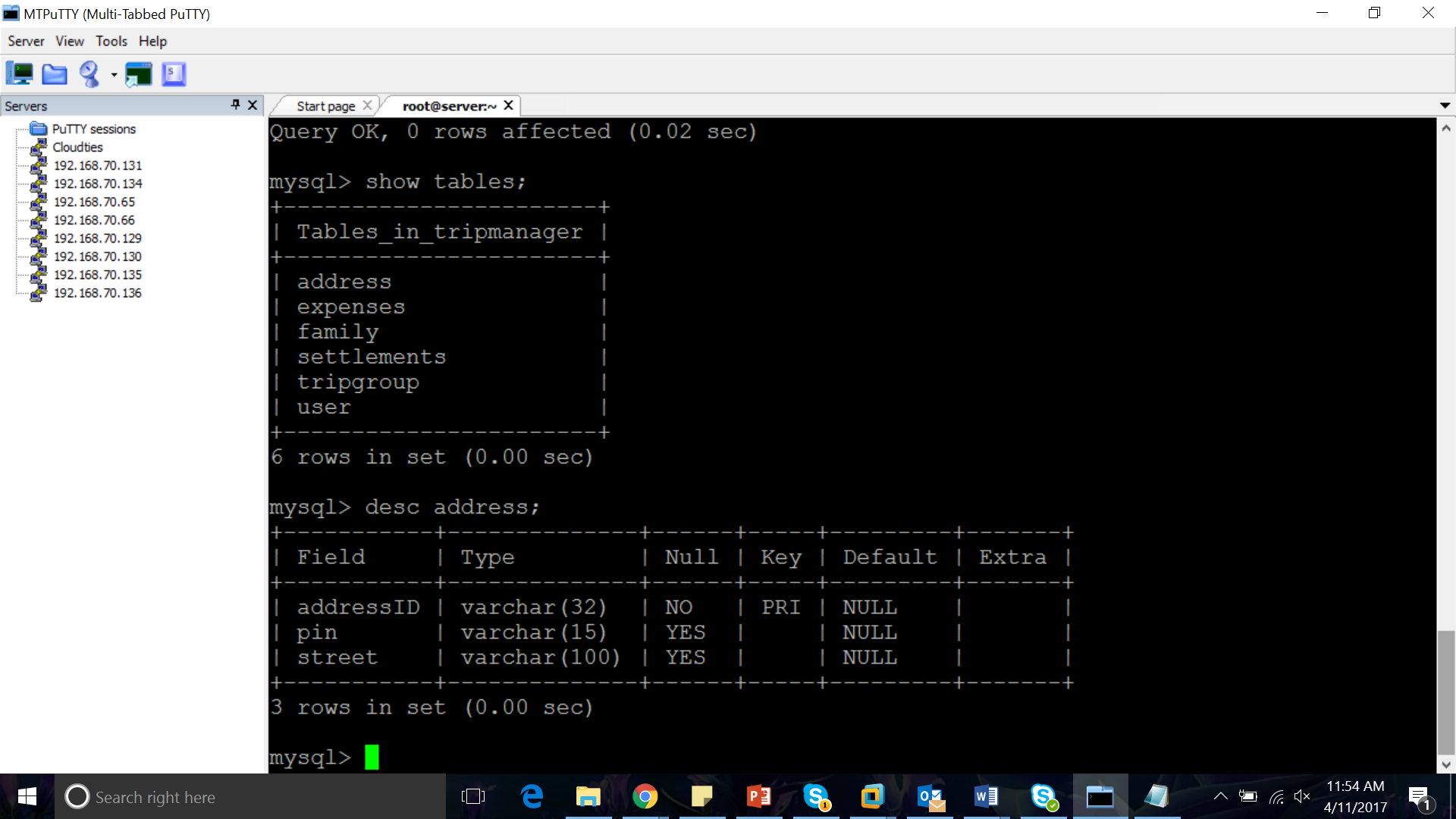
<https://dev.mysql.com/doc/refman/5.7/en/tutorial.html>

**Key sets:**

**Create table query:**







**TABLE CREATION AND DATA INSERTION:**

**USER:**

create table user(username VARCHAR(32) primary key, plainPassword VARCHAR(32), encryptedPassword VARCHAR(255), name VARCHAR(100), email VARCHAR(100), creationDate VARCHAR(15), modificationDate VARCHAR(15));

INSERT INTO user(username, plainPassword, encryptedPassword, name, email, creationDate, modificationDate)VALUES("keerthana","admin123","jdcjdncncs","keerthi","keerthi@gmail.com","01apr2017","11apr2017");

**TABLE DUPLICATE/CLONE:**

create table user\_duplicate(username VARCHAR(32) primary key, plainPassword VARCHAR(32), encryptedPassword VARCHAR(255), name VARCHAR(100), email VARCHAR(100), creationDate VARCHAR(15), modificationDate VARCHAR(15));

INSERT INTO user\_duplicate(username,plainPassword,encryptedPassword,name,email,creationDate, modificationDate)SELECT \* FROM user;

**FAMILY:**

create table family(familyID VARCHAR(32) primary key, username VARCHAR(32), name VARCHAR(100), sex VARCHAR(10), dob VARCHAR(15), addressID VARCHAR(32));

insert into family(familyID,username,name,sex,dob,addressID)values("001",keerthana","keerthi","f","23may1993","chennai");

**ADDRESS:**

create table address(addressID VARCHAR(32) primary key, pin VARCHAR(15), street VARCHAR(100));

insert into address(addressID,pin,street)values("vellore","632007","4thmainroad");

**TRIPGROUP:**

create table tripgroup(groupID VARCHAR(32) primary key, groupName VARCHAR(50), Description VARCHAR(255));

insert into tripgroup(groupID,groupName,Description)values("001","dcjd","cdjnj");

**EXPENSES:**

create table expenses(ID VARCHAR(255) primary key, groupID VARCHAR(32), username VARCHAR(32), particular VARCHAR(50), amount INTEGER(255), date VARCHAR(15), FOREIGN KEY (groupID) REFERENCES tripgroup(groupID));

insert into expenses(ID,groupID,username,particular,amount,date)values("1","001","keerthana","jnsd","500","04apr2017);

**SETTLEMENTS:**

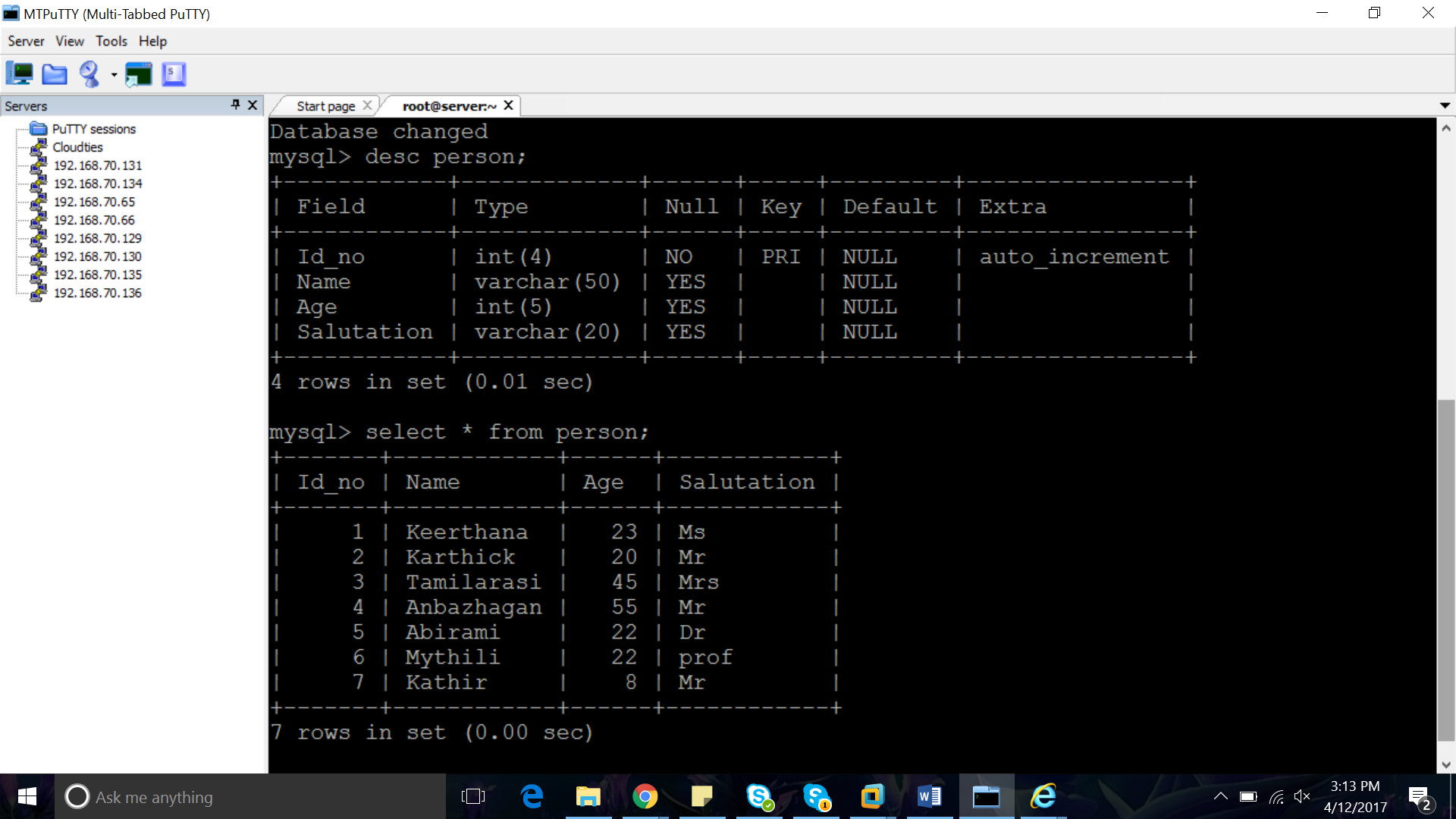
create table settlements(ID VARCHAR(255) primary key, groupID VARCHAR(32), username VARCHAR(32), oweto VARCHAR(50), amount INTEGER(255), date VARCHAR(15), FOREIGN KEY (groupID) REFERENCES tripgroup(groupID));

insert into settlements(ID,groupID,username,oweto,amount,date)values("1","001","keerthana","jnsd","500","04apr2017);

**MySQL SEQUENCE:**

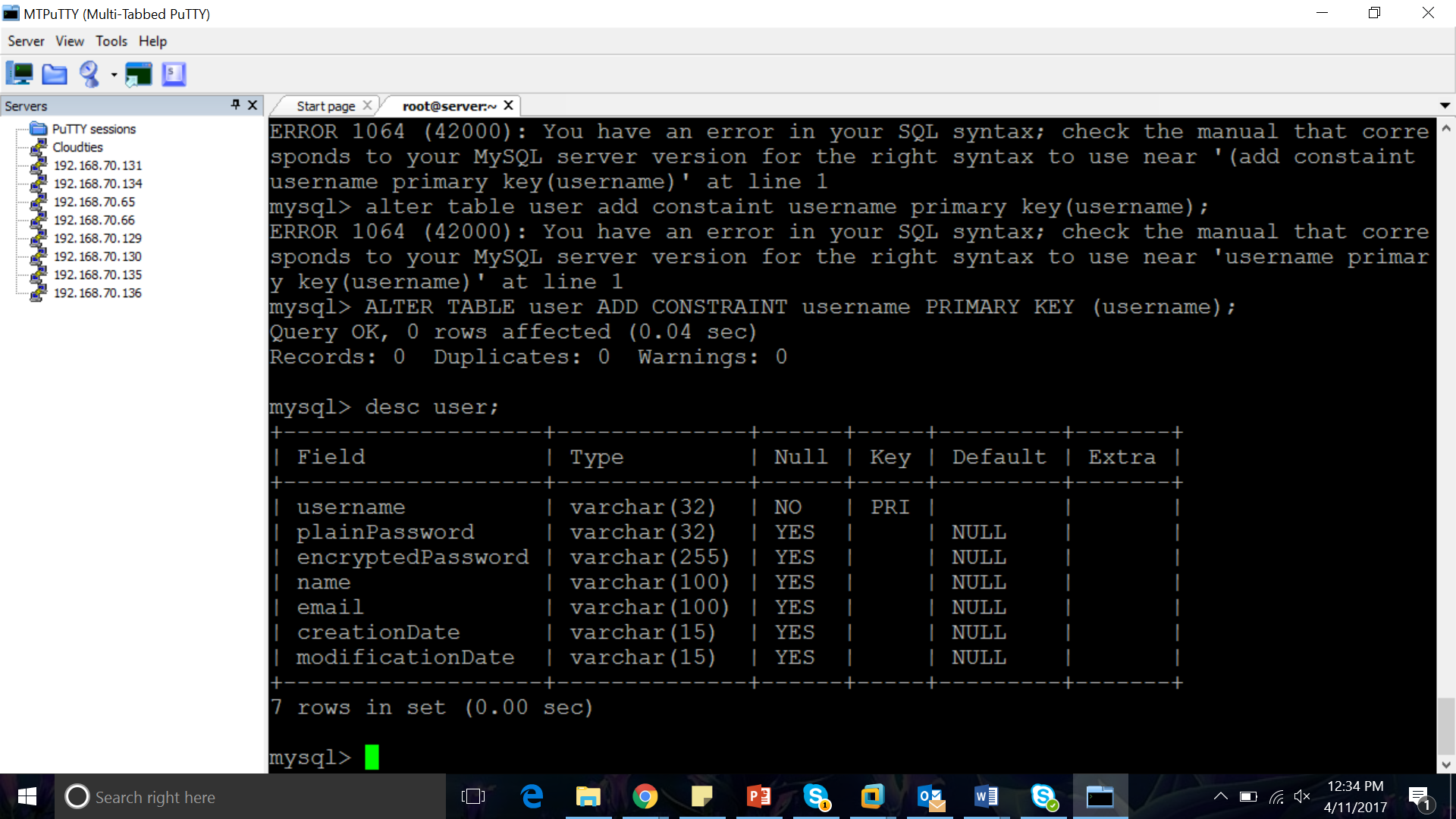
CREATE TABLE person( Id\_no INT(4) AUTO\_INCREMENT PRIMARY KEY,Name VARCHAR(50),Age INT(5), Salutation VARCHAR(50));

INSERT INTO person(Id\_no,Name,Age,Salutation)VALUES('10','Mithu','7','Ms');



**ALTER TABLE:**

ALTER TABLE user ADD CONSTRAINT username PRIMARY KEY (username);



ALTER TABLE family ADD CONSTRAINT familyID PRIMARY KEY (familyID);

ALTER TABLE address ADD CONSTRAINT addressID PRIMARY KEY (addressID);

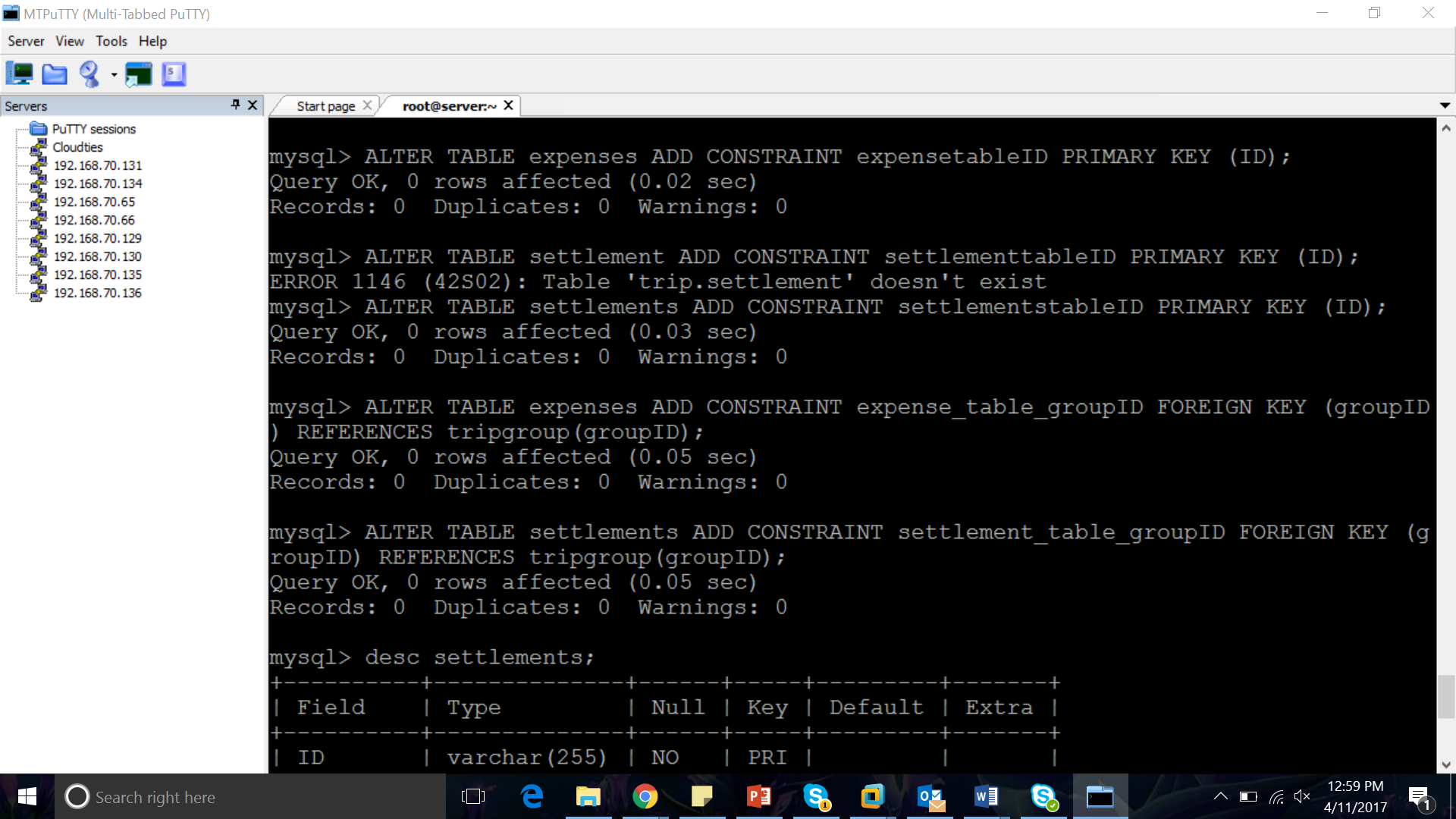
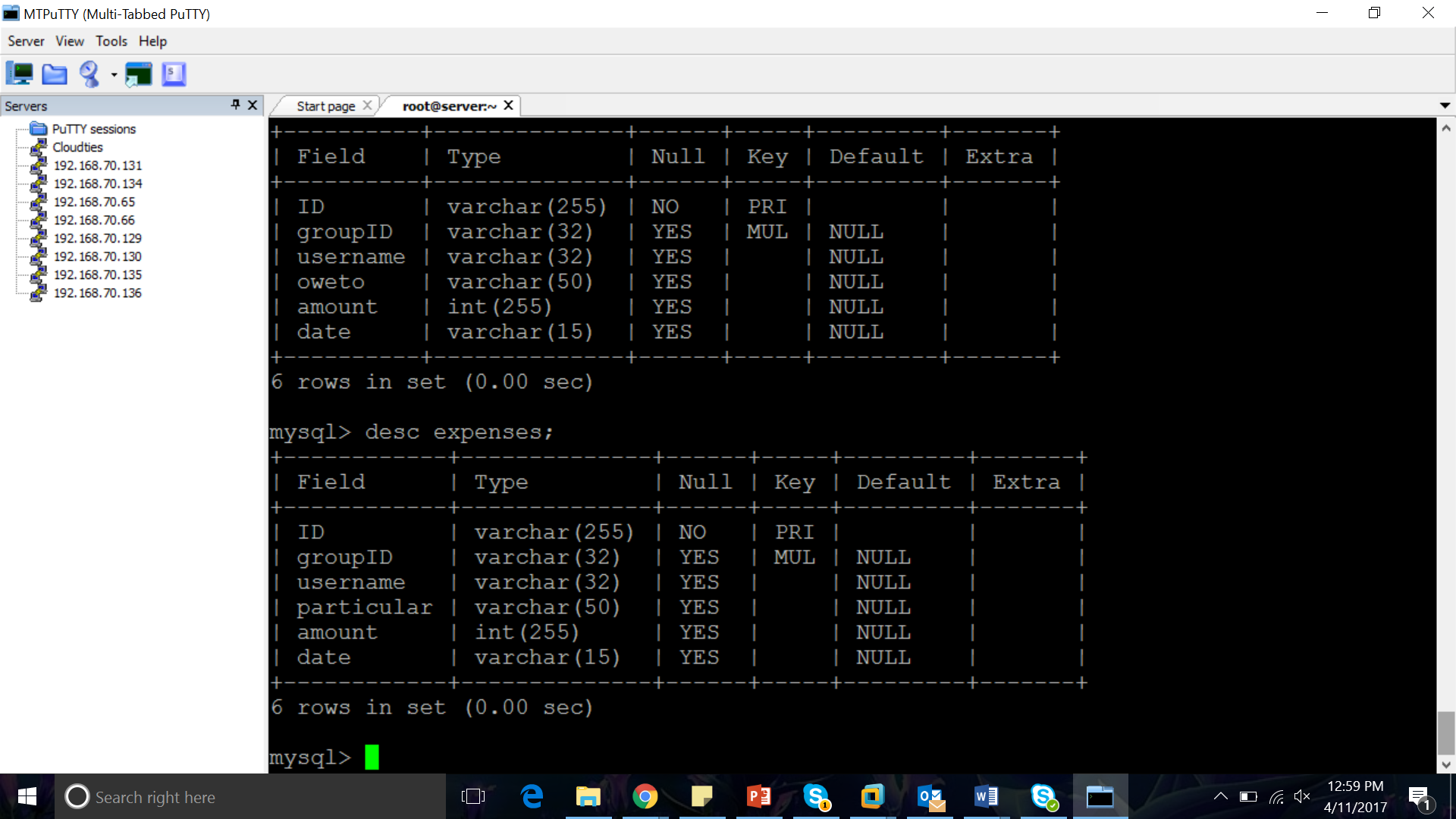
ALTER TABLE tripgroup ADD CONSTRAINT groupID PRIMARY KEY (groupID);

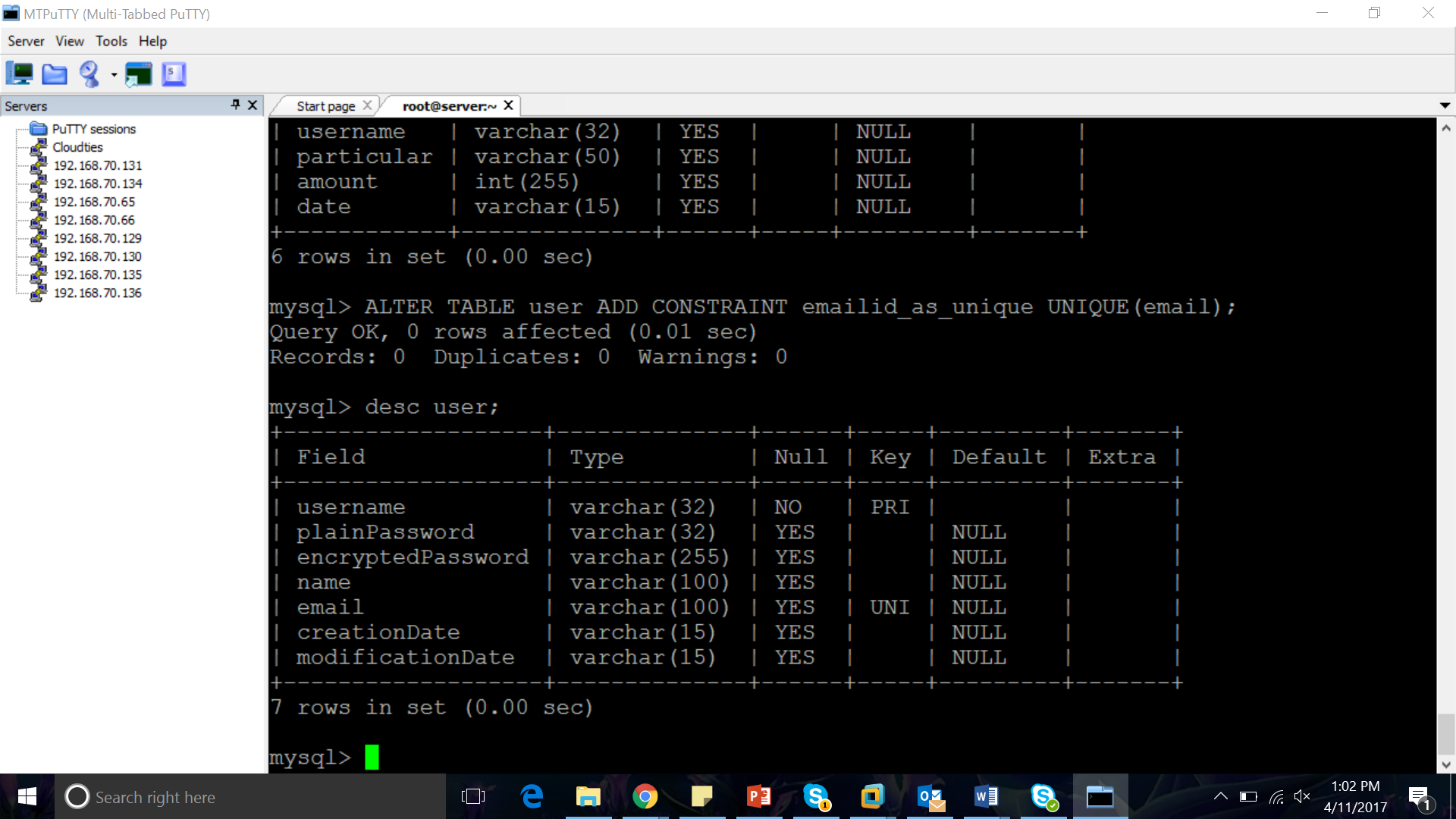
ALTER TABLE expenses ADD CONSTRAINT expensetableID PRIMARY KEY (ID);

ALTER TABLE settlement ADD CONSTRAINT settlementtableID PRIMARY KEY (ID);

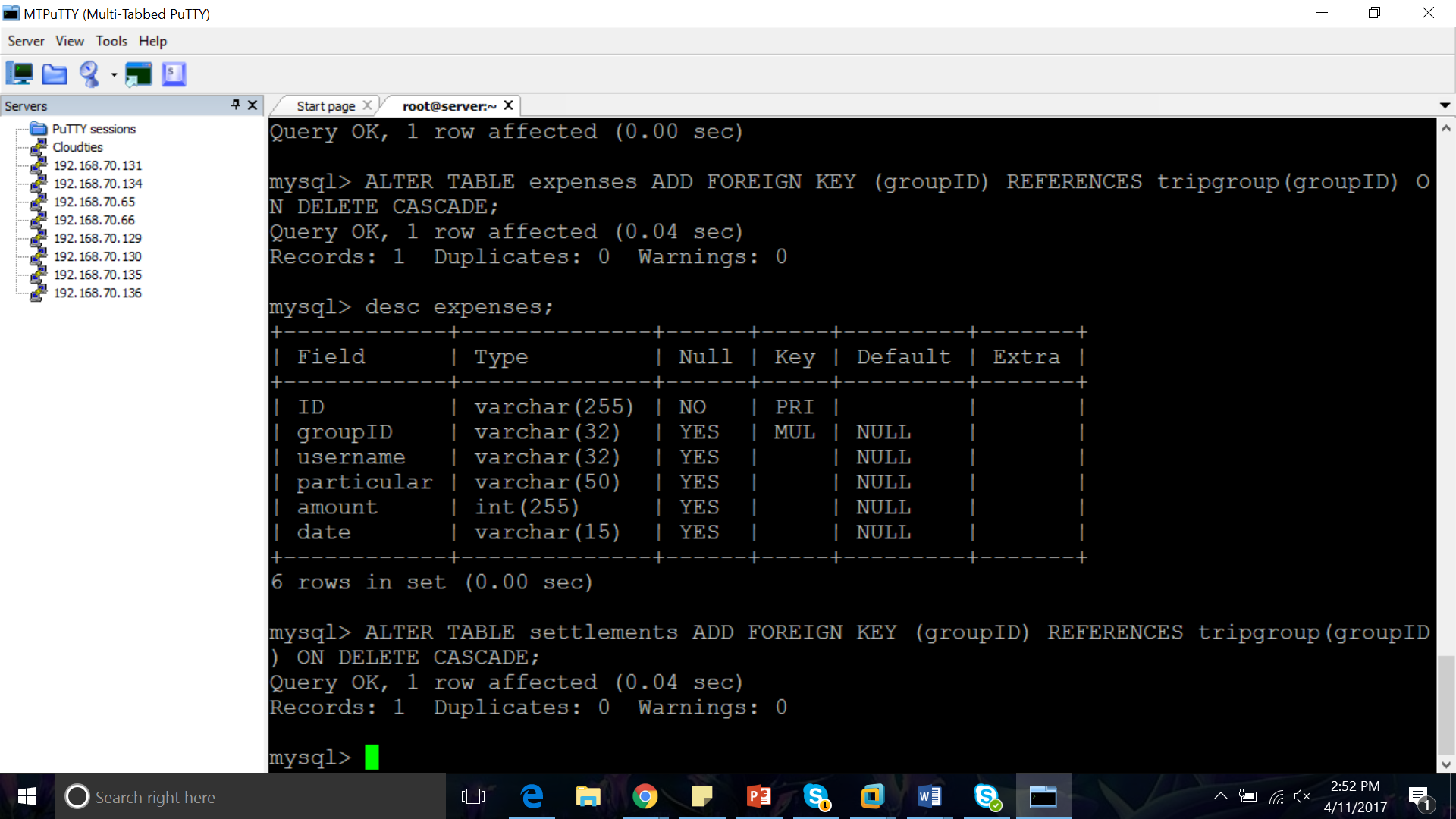
ALTER TABLE expenses ADD CONSTRAINT expense\_table\_groupID FOREIGN KEY (groupID) REFERENCES tripgroup(groupID);

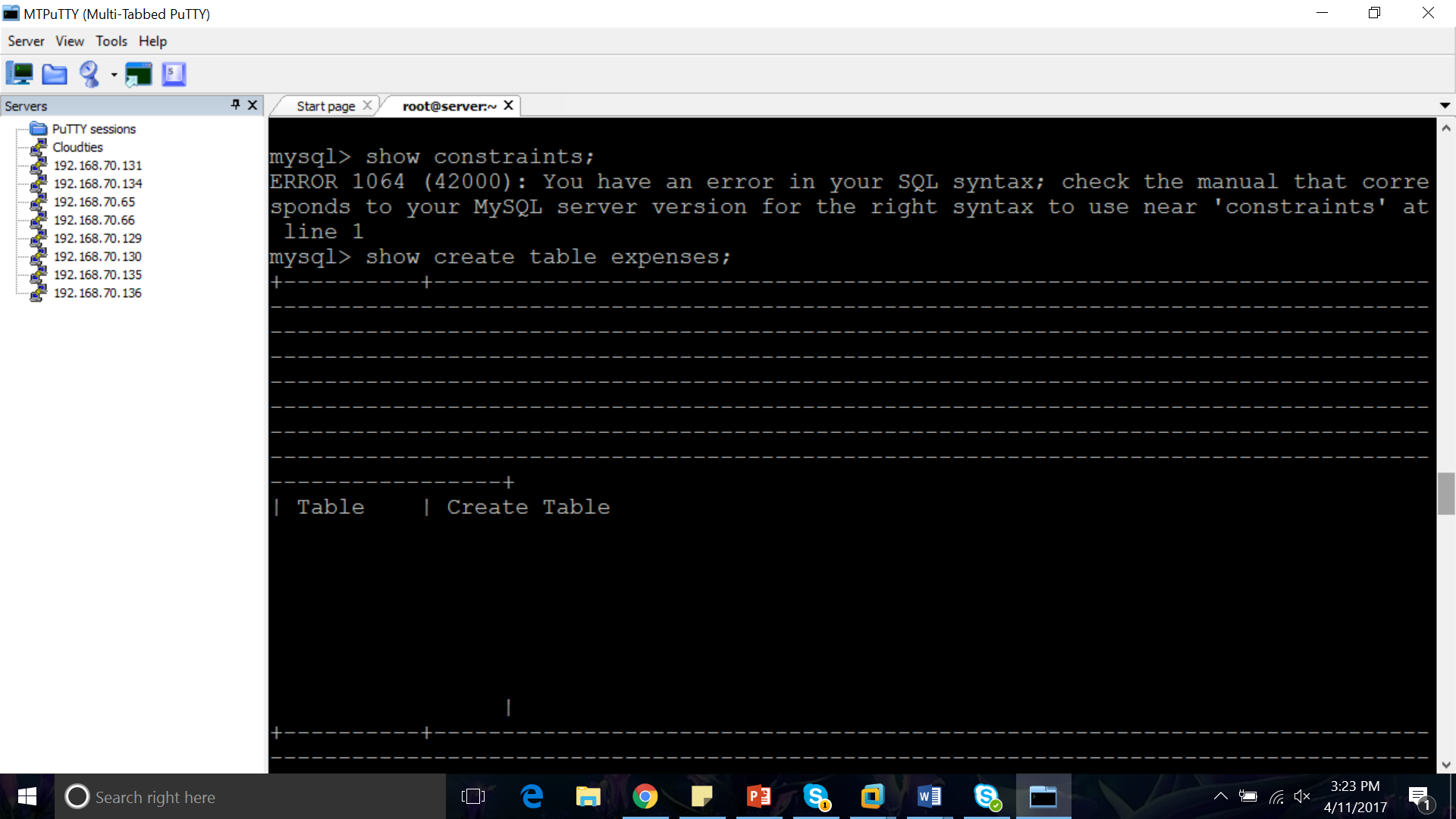
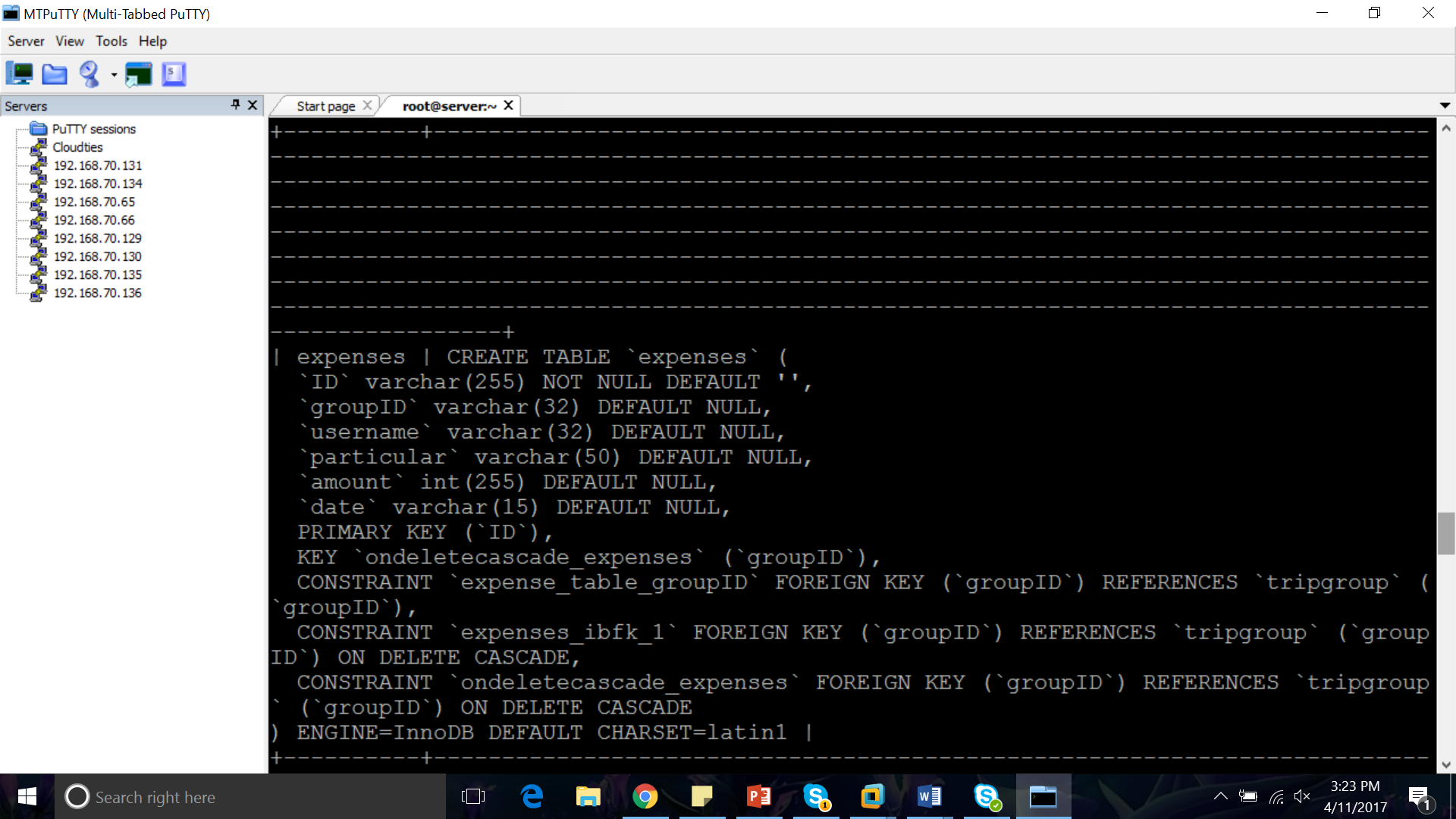
ALTER TABLE settlements ADD CONSTRAINT settlement\_table\_groupID FOREIGN KEY (groupID) REFERENCES tripgroup(groupID);

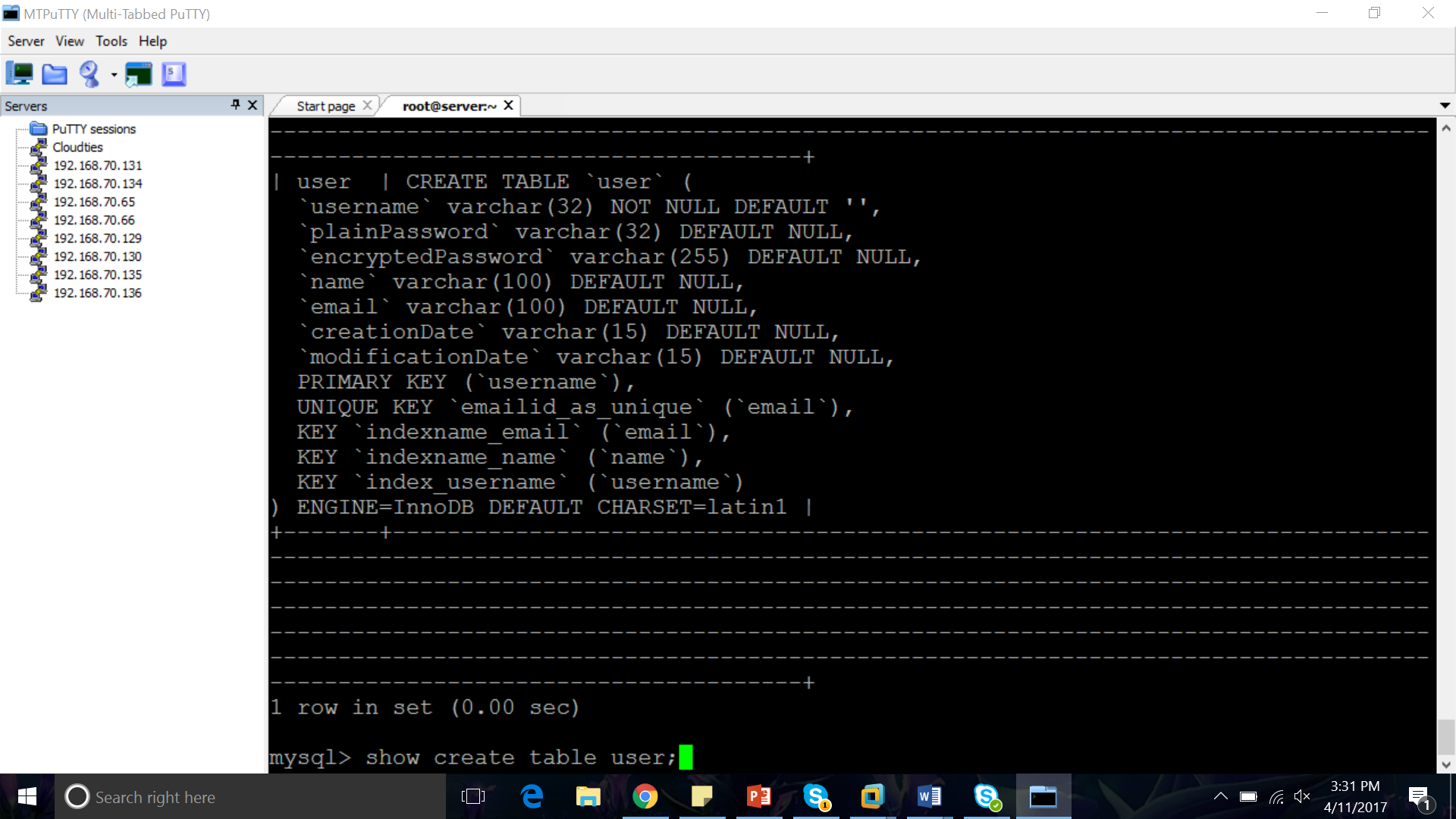
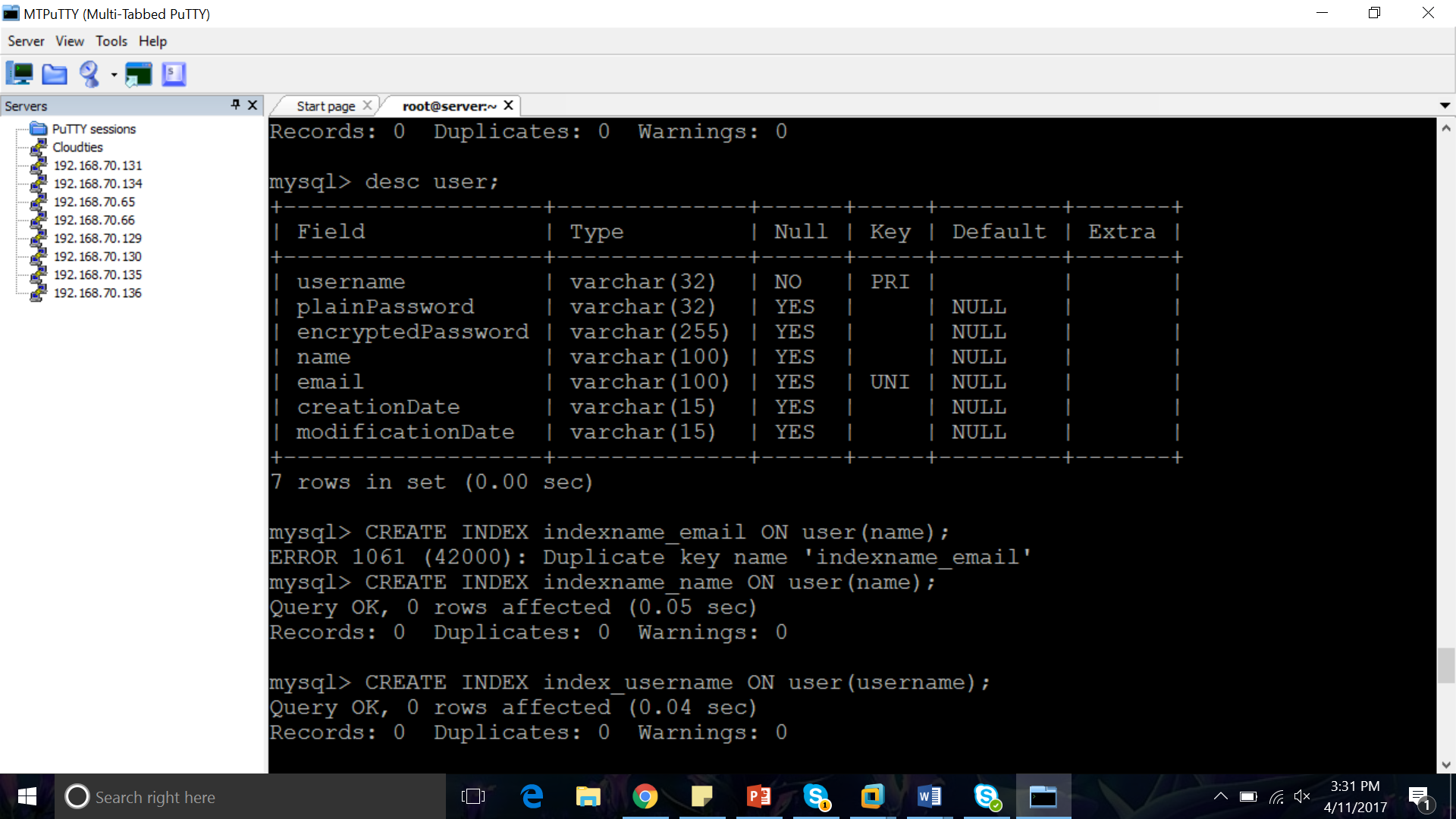
 

ALTER TABLE user ADD CONSTRAINT emailid\_as\_unique UNIQUE(email); 

ALTER TABLE expenses ADD FOREIGN KEY (groupID) REFERENCES tripgroup(groupID) ON DELETE CASCADE;



**PL/SQL:**

PL/SQL is a combination of SQL along with the procedural features of programming languages.

**FUNCTIONS AND PROCEDURE WITHOUT AND WITH PARAMETER:**

DELIMITER $$

CREATE PROCEDURE HelloWorld()

BEGIN

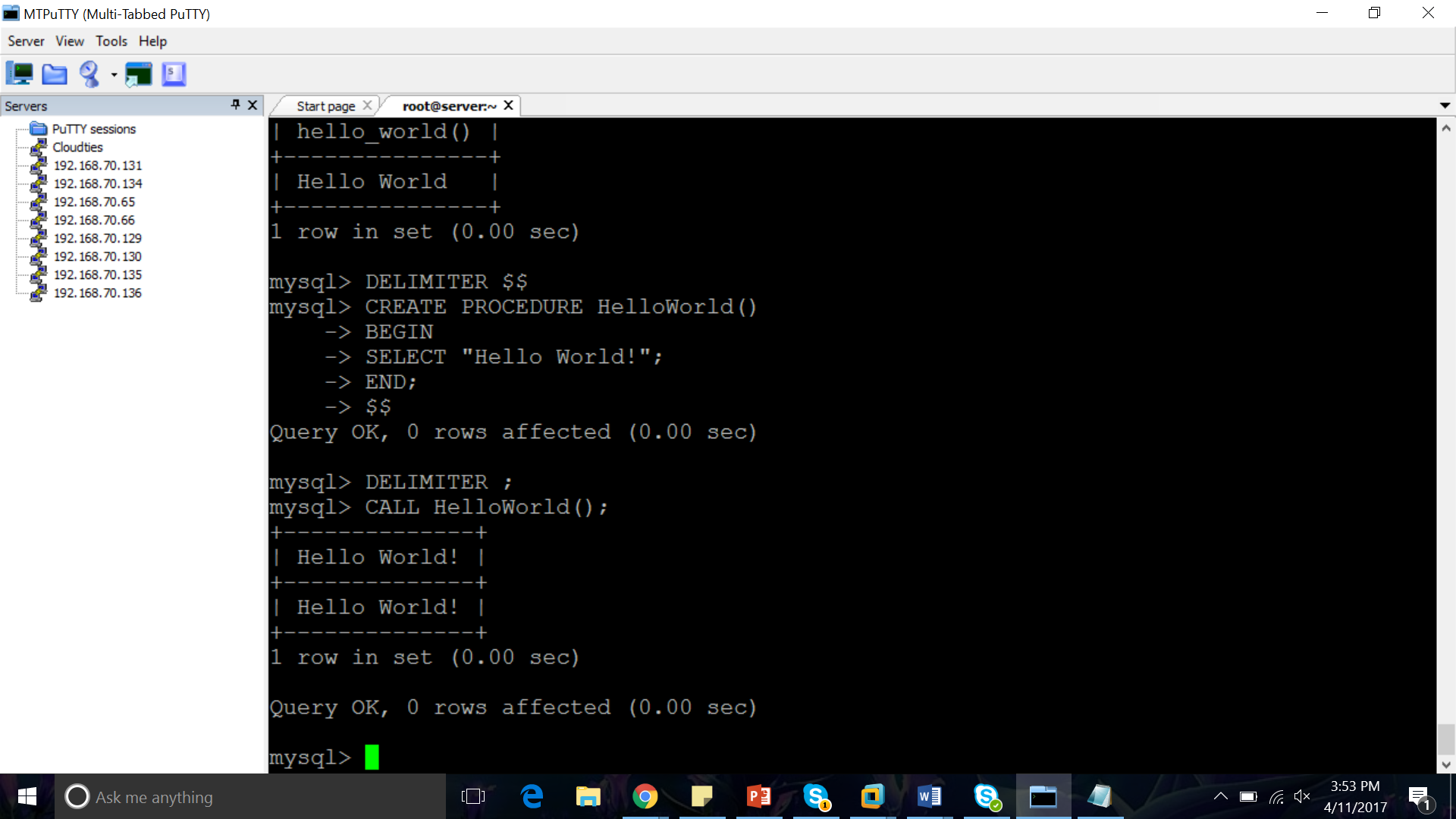
SELECT "Hello World!";

END;

$$

DELIMITER ;

CALL HelloWorld();



DELIMITER $$

CREATE PROCEDURE HelloWorld()

BEGIN

SELECT "Hello World!";

END;

$$

DELIMITER ;

CALL HelloWorld();

DELIMITER $$

CREATE FUNCTION hello\_world()

RETURNS TEXT

LANGUAGE SQL

BEGIN

RETURN 'Hello World';

END;

$$

DELIMITER ;

SELECT hello\_world();

create function abc() returns char(50) deterministic return "Hello World";

select abc();

create function abc(s CHAR(20)) returns char(50) deterministic return "Hello World";

select abc(‘Hi’);

DELIMITER //

CREATE FUNCTION ADDSUM()

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE A INT;

DECLARE b INT;

SET A = 20;

SET B = 10;

RETURN A+B;

END;

//

Select ADDSUM();

CREATE FUNCTION Table\_TEST()

RETURNS VARCHAR(32)

DETERMINISTIC

BEGIN

DECLARE NAME\_FOUND VARCHAR(32);

SELECT username INTO NAME\_FOUND FROM expenses WHERE groupID="001";

RETURN NAME\_FOUND;

END;

//

SELECT Table\_Test();

CREATE PROCEDURE HelloWorld()

DETERMINISTIC  
BEGIN  
ECHO "Hello World!";  
END; //

delimiter;  
CALL HelloWorld();

DELIMITER //

CREATE PROCEDURE id(IN username VARCHAR(32))

BEGIN

SELECT \*

FROM user

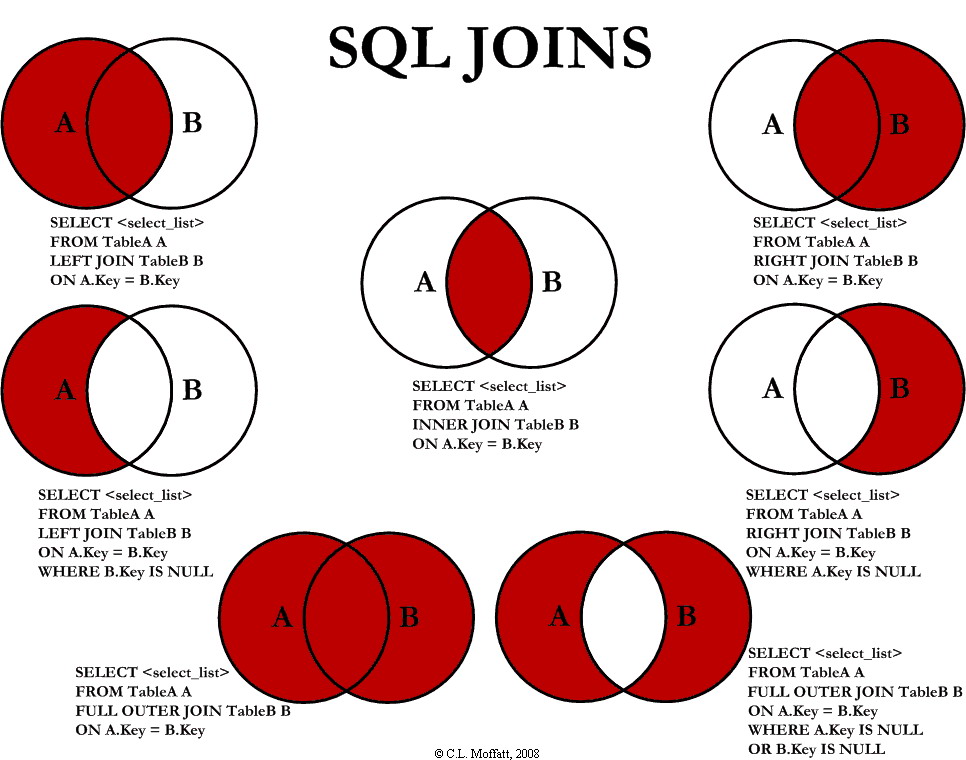
WHERE user = username;

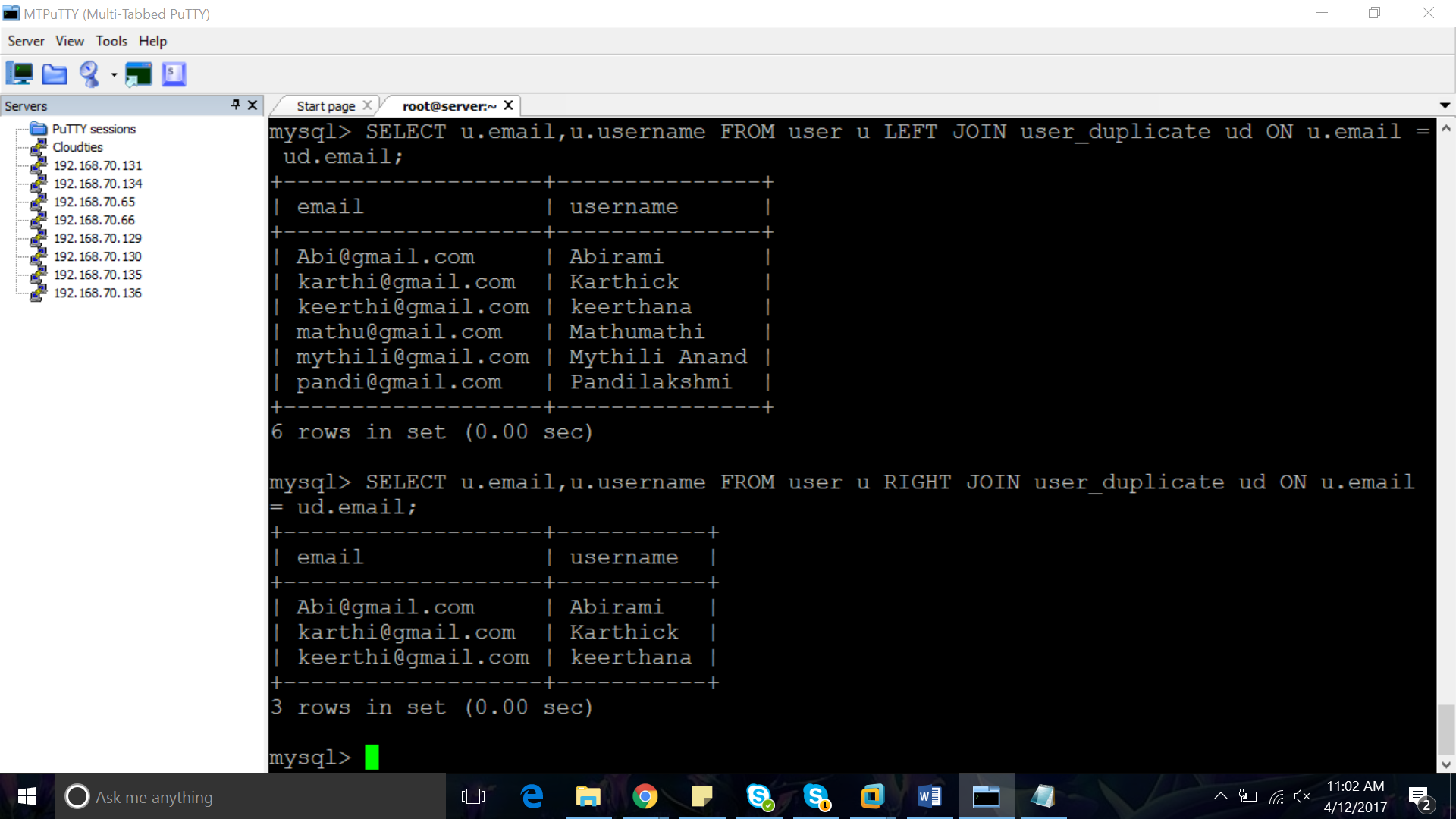
END //

DELIMITER ;

CALL id('keerthana');

**SQL UNION AND INTERSECTION:**





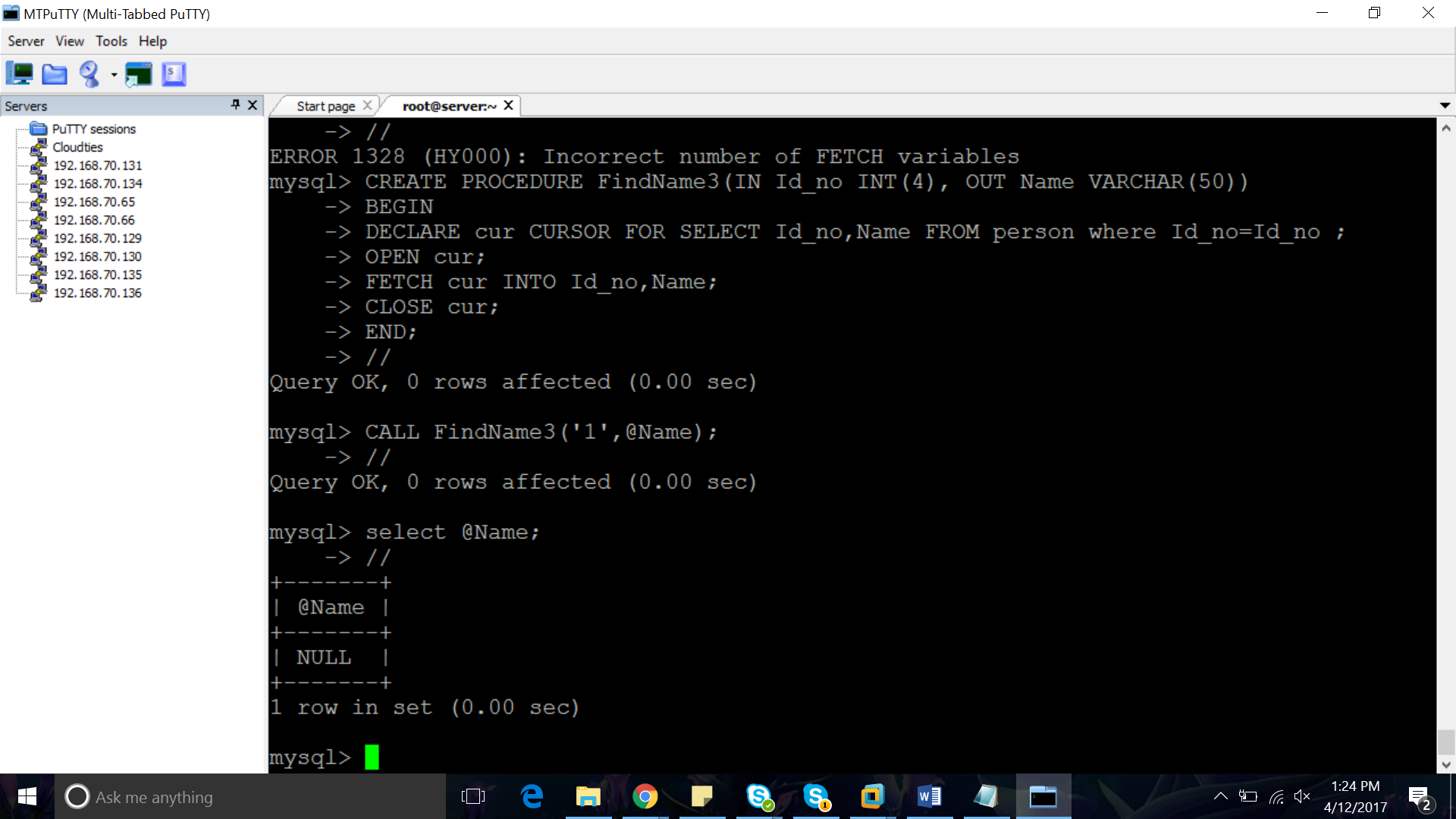
SELECT u.email,u.username FROM user u LEFT JOIN user\_duplicate ud ON u.email = ud.email;

SELECT u.email,u.username FROM user u RIGHT JOIN user\_duplicate ud ON u.email = ud.email;

SELECT u.email,u.username FROM user u LEFT JOIN user\_duplicate ud ON u.email = ud.email IS NULL;

SELECT u.email,u.username FROM user u RIGHT JOIN user\_duplicate ud ON u.email = ud.email;

**CURSOR:**



CREATE PROCEDURE FindName3(IN Id\_no INT(4), OUT Name VARCHAR(50))

BEGIN

DECLARE cur CURSOR FOR SELECT Id\_no,Name FROM person where Id\_no=Id\_no ;

OPEN cur;

FETCH cur INTO Id\_no,Name;

SELECT \* FROM person;

CLOSE cur;

END;

//

CALL FindName3('1',@Name);

CREATE PROCEDURE curdemo3()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE a varchar(32);

DECLARE b varchar(32);

DECLARE c varchar(32);

DECLARE d varchar(32);

DECLARE e varchar(32);

DECLARE f varchar(15);

DECLARE h varchar(15);

DECLARE i varchar(32);

DECLARE cur1 CURSOR FOR SELECT username,password,epassword,name,email,creationDate,modificationDate FROM user;

DECLARE cur2 CURSOR FOR SELECT email FROM user\_backup;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur1;

OPEN cur2;

read\_loop: LOOP

FETCH cur1 INTO a, b,c,d,e,f,h;

FETCH cur2 INTO i;

IF done THEN

LEAVE read\_loop;

END IF;

IF e!=i THEN

INSERT INTO user\_backup VALUES (a,b,c,d,e,f,h);

END IF;

END LOOP;

CLOSE cur1;

CLOSE cur2;

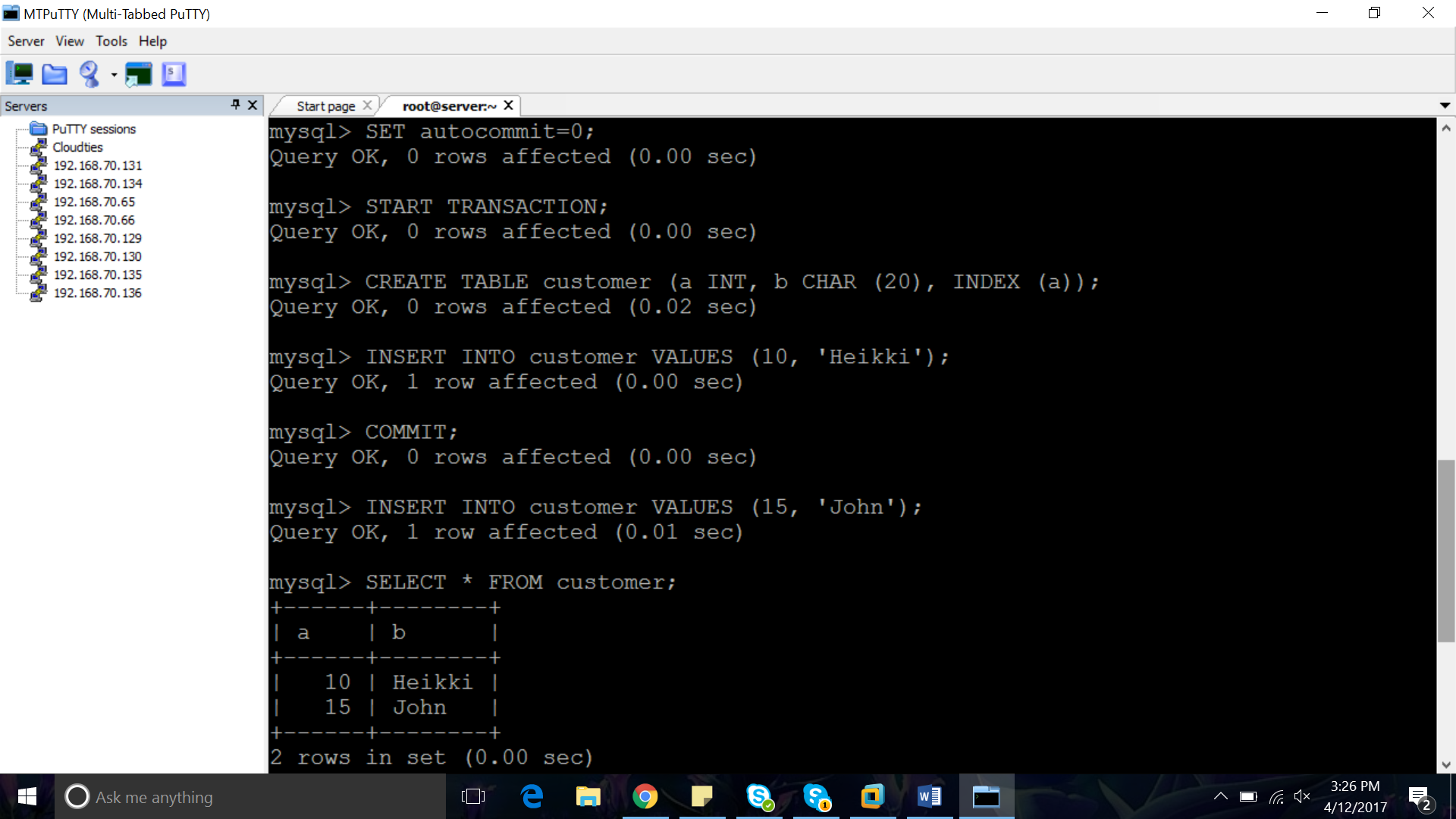
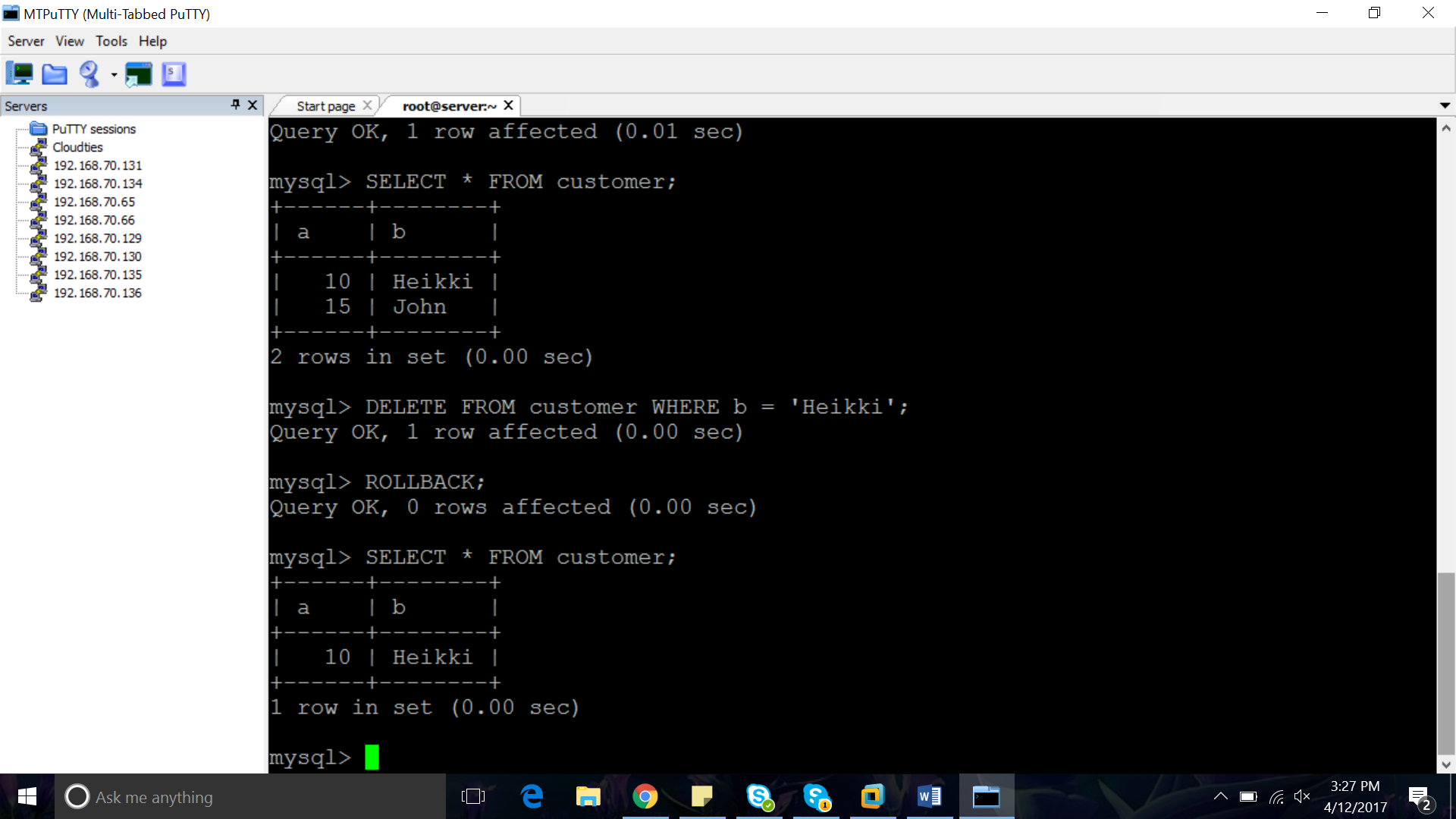
END;

**TRANSACTIONS:**

Read

<http://www.dotnetspeak.com/data/transaction-isolation-levels-explained-in-details/>

<https://elliot.land/post/sql-transaction-isolation-levels-explained>

SET autocommit=0;

START TRANSACTION;

CREATE TABLE customer (a INT, b CHAR (20), INDEX (a));

INSERT INTO customer VALUES (10, 'Heikki');

COMMIT;

INSERT INTO customer VALUES (15, 'John');

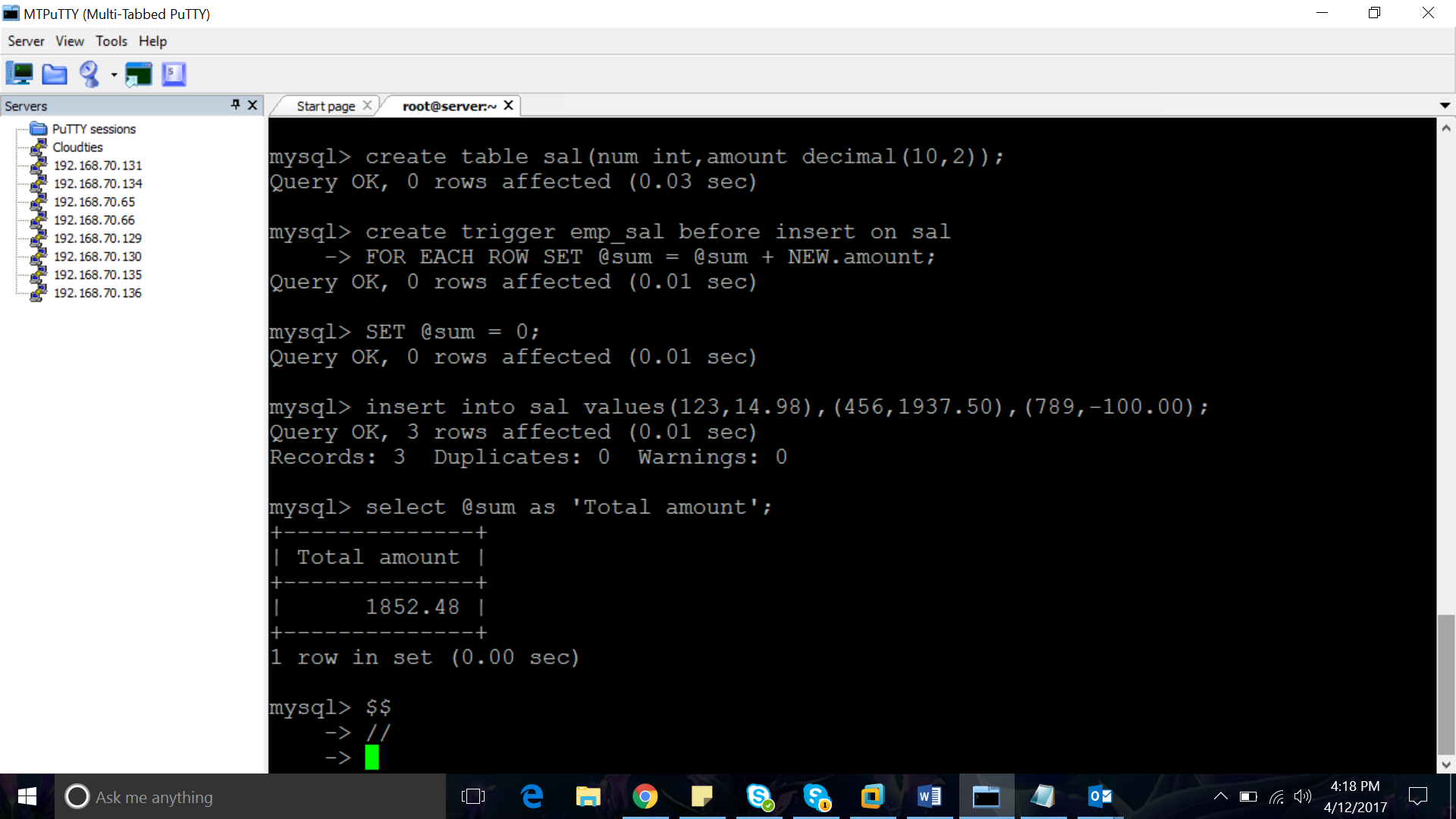
SELECT \* FROM customer;

DELETE FROM customer WHERE b = 'Heikki';

ROLLBACK;

SELECT \* FROM customer;

**TRIGGERS:**



create table sal(num int,amount decimal(10,2));

create trigger emp\_sal before insert on sal

FOR EACH ROW SET @sum = @sum + NEW.amount;

SET @sum = 0;

insert into sal values(123,14.98),(456,1937.50),(789,-100.00);

select @sum as 'Total amount';

$$

DELIMITER //

CREATE TRIGGER contacts\_after\_insert AFTER INSERT ON person FOR EACH ROW

BEGIN

DECLARE Name varchar(50);

SELECT FindName3('1',@Name) INTO Name;

INSERT INTO person\_dup(Id\_no,Name,Age,Salutation)VALUES('10','Mithu','7','Ms');

END; //

DELIMITER;

**SUB QUERIES:**

SELECT username,oweto,amount FROM settlements WHERE amount=(SELECT MAX(amount) FROM settlements);