**Online Store**

1. The main purpose of my project is to make an online store for clothes (like asos, forever 21 etc.) Customer makes an order online. Order might contain of a lot of clothes. Company will ship clothes to customer’s address. Manager will work with customer and get feedback on their work and online store in general from users. Therefore, I want my database to store information about clothes, companies, orders, customers, feedback managers, shipping. Their connection between each other is represented in ERD diagram.
2. create table orders (

id\_order int,

o\_sum int,

amount int,

primary key(id\_order)

);

create table clothes (

id\_clothes int,

c\_type varchar,

model varchar,

c\_size varchar,

amount int,

material varchar,

id\_order int,

primary key(id\_clothes),

foreign key (id\_order) references orders(id\_order)

);

create table shipping (

postalcode int,

country varchar,

city varchar,

primary key(postalcode)

);

create table customer (

id\_customer int,

c\_name varchar,

card int,

telnum varchar,

address varchar,

id\_order int,

postalcode int,

primary key(id\_customer),

foreign key (id\_order) references orders(id\_order),

foreign key (postalcode) references shipping(postalcode)

);

create table feedback (

id\_feed int,

feedback varchar,

primary key(id\_feed)

);

create table company (

id\_comp int,

comp\_name varchar,

address varchar,

postalcode int,

primary key(id\_comp),

foreign key (postalcode) references shipping(postalcode)

);

create table manager (

id\_manager int,

m\_name varchar,

salary int,

working\_days int,

id\_comp int,

id\_customer int,

id\_feed int,

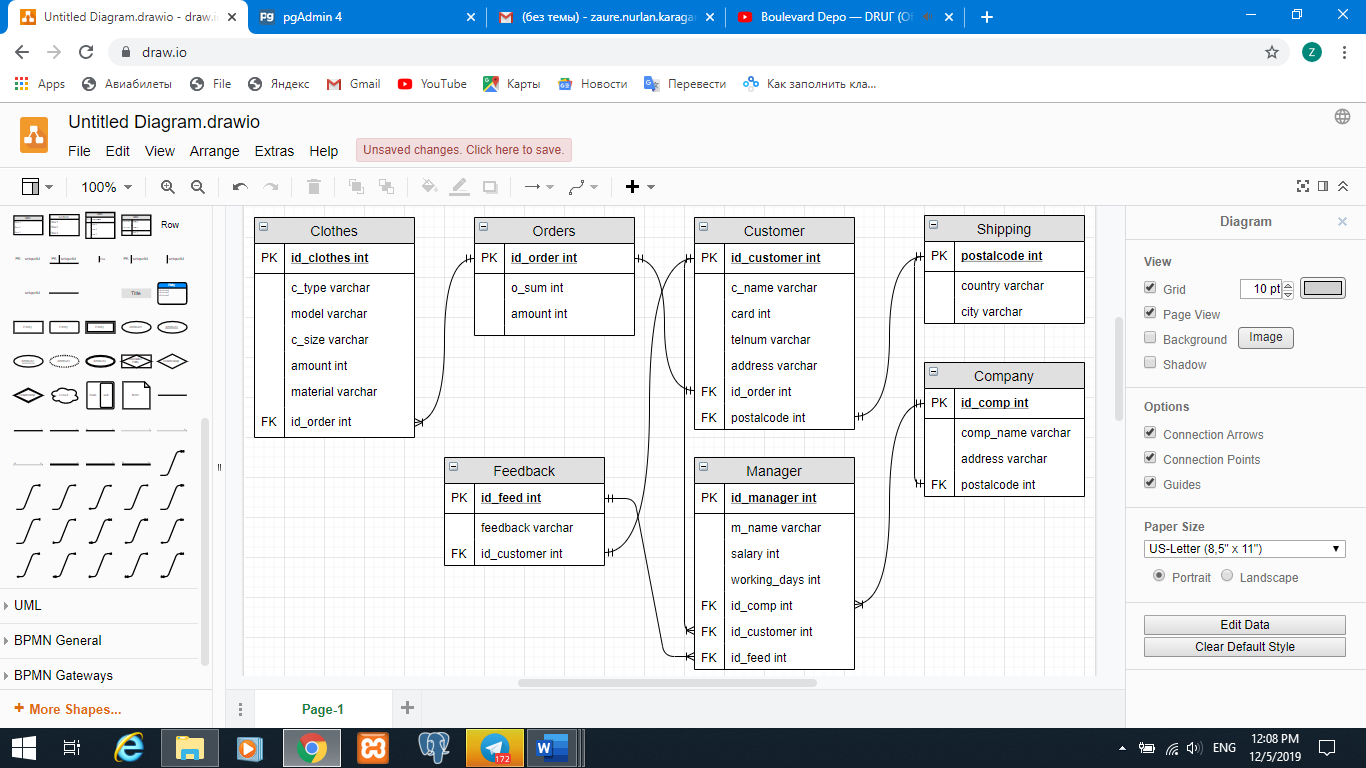
primary key(id\_manager),

foreign key (id\_comp) references company(id\_comp),

foreign key (id\_customer) references customer(id\_customer),

foreign key (id\_feed) references feedback(id\_feed)

);

1. 

hires

receives

sends

asks

makes

gives

analyzes

Contain

1. create table store\_owner (

owner\_id int,

owner\_name varchar

);

alter table store\_owner

add primary key (owner\_id);

alter table store\_owner

alter column owner\_name set not null;

drop column owner\_name;

drop table store\_owner;

alter table feedback

add column id\_customer int;

alter table feedback

add foreign key (id\_customer) references customer(id\_customer);

1. insert into shipping (postalcode, country, city) values

(1000, 'Kazakhstan', 'Astana'),

(5000, 'Kazakhstan', 'Almaty'),

(8000, 'Kazakhstan', 'Taraz'),

(2000, 'Kazakhstan', 'Karaganda'),

(101000, 'Russia', 'Moscow'),

(15000, 'Kazakhstan', 'Karaganda);

Update shipping set country=’Russia’, city=’Piter’ where postalcode=11550;

Delete from shipping where postalcode=11550;

insert into company (id\_comp, comp\_name, address, postalcode) values

(1, 'Assel', 'Glushchenko', 1000),

(2, 'Mechanical Piano', 'Zapadnaya', 5000),

(3, 'AIKEN', 'Ul.shevchenko', 8000),

(4, 'Alima', 'Ploshchad Pobedy', 2000),

(5, 'Zherebtsov', 'Akademika Bekturova', 101000),

(6, 'Aliya', 'Abay', 15000);

Update company set comp\_name= ‘Aliya Aliya’ where id\_comp=6;

Delete from company where address= ‘Abay’;

insert into orders (id\_order, o\_sum, amount) values

(1, 2000, 10),

(2, 1000, 5),

(3, 500, 3),

(4, 3000, 12),

(5, 1500, 7),

(6, 6600, 11);

Update orders set o\_sum=o\_sum+400 where id\_order=6;

Delete from orders where id\_order=6;

insert into clothes (id\_clothes, c\_type, model, c\_size, amount, material, id\_order) values

(1, 'hoodie', 'slimfit', 'M', 25, 'cotton', 1),

(2, 'leggins', 'slimfit', 'S', 10 , 'polyester', 2),

(3, 'shirt', 'regularfit', 'L', 10, 'cotton', 3),

(4, 'coat', 'regularfit', 'M', 5, 'nylon', 4),

(5, 'dress', 'slimfit', 'S', 15, 'silk', 5),

(6, 'coat', 'regularfit', 'M', 6, 'nylon', 6);

Update clothes set c\_type= ‘dress’ and c\_size=S where id\_order=6;

Delete from clothes where id\_clothes=6;

insert into customer (id\_customer,c\_name,card,telnum,address,id\_order,postalcode) values

(1, 'Zhanel', 101, '+7778745785', 'Ul.pospelova', 1, 1000),

(2, 'Aitugan', 102, '+77012366584', 'Serova 4', 2, 5000),

(3, 'Lyazzat', 103, '+77025879486', 'Ul.esenberlina', 3, 8000),

(4, 'Alisher', 104, '+77478925674', '2 Ambulatornyy Zaezd', 4, 2000),

(5, 'Marina', 105, '+77894567485', 'Mkr. Saltanat', 5, 101000),

(6, 'Alina', 106, '+77479529778', '2 Ambulatornyy Zaezd', 6, 2000);

Update customer set c\_name= ‘Aliya’ and telnum=’+77019529778’ where id\_order=6;

Delete from customer where id\_customer=6;

insert into feedback (id\_feed, feedback, id\_customer) values

(1, 'Good service. Keep it up!', 1),

(2, 'I had got the wrong size', 2),

(3, 'The maretial feels so good on my skin!', 3),

(4, 'I am obsessed with my purchase!', 4),

(5, 'I did not get my order in 2 weeks. I think you should work on your shipping system', 5),

(6, 'It was amazing!', 6);

Update set feedback = ‘Amazing’ where id\_feed =6;

Delete from where id\_feed=6;

insert into manager (id\_manager,m\_name, salary,working\_days,id\_comp,id\_customer,id\_feed) values

(1, 'Kaisar', 150000, 3, 1, 1, 1),

(2, 'Otabek', 200000, 4, 2, 2, 2),

(3, 'Alikhan', 170000, 4, 3, 3, 3),

(4, 'Dimash', 300000, 6, 4, 4, 4),

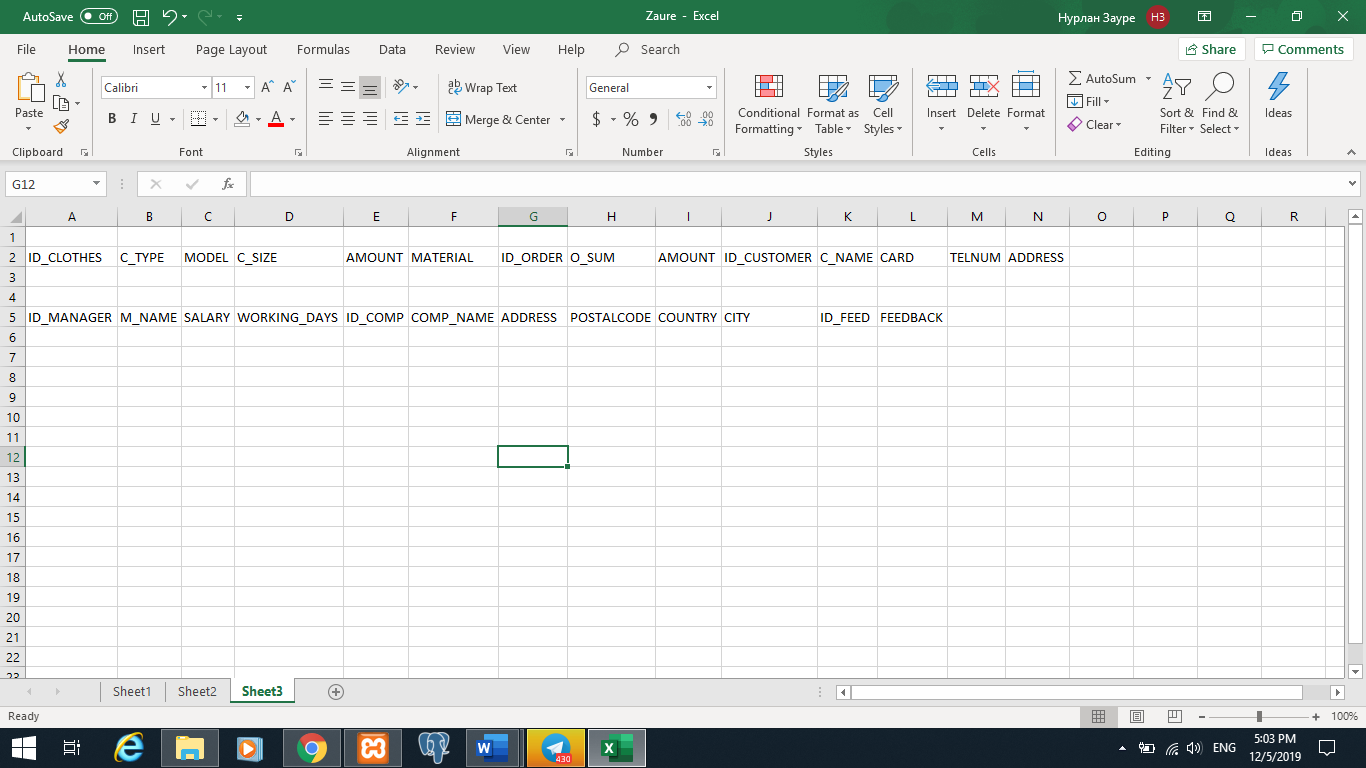
(5, 'Zaure', 250000, 5, 5, 5, 5),

(6, 'Dima', 450000, 6, 6, 6, 6);

Update set salary= salary-50000 where id\_manager=6;

Delete from where id\_manager=6;

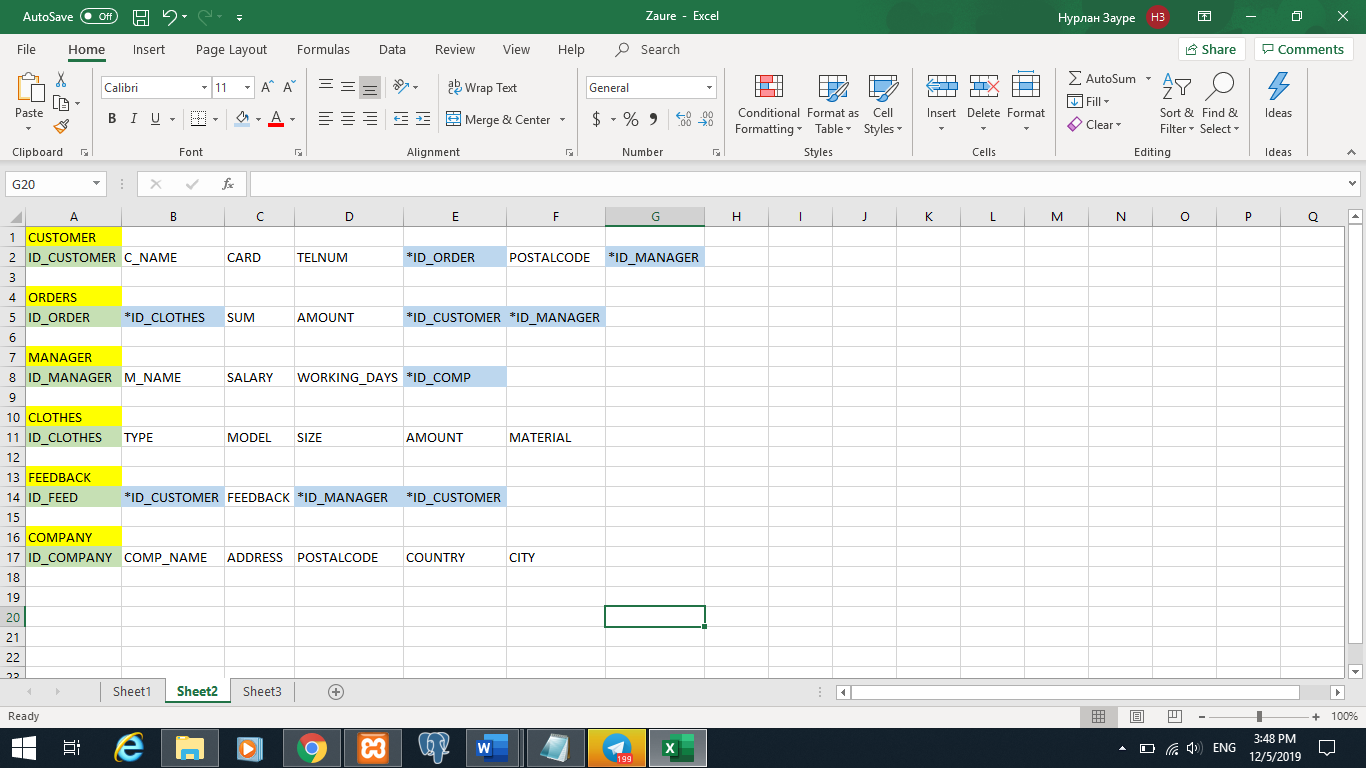
1. Unnormalized data



First Normalization

All data should be atomic.

Second Normalization



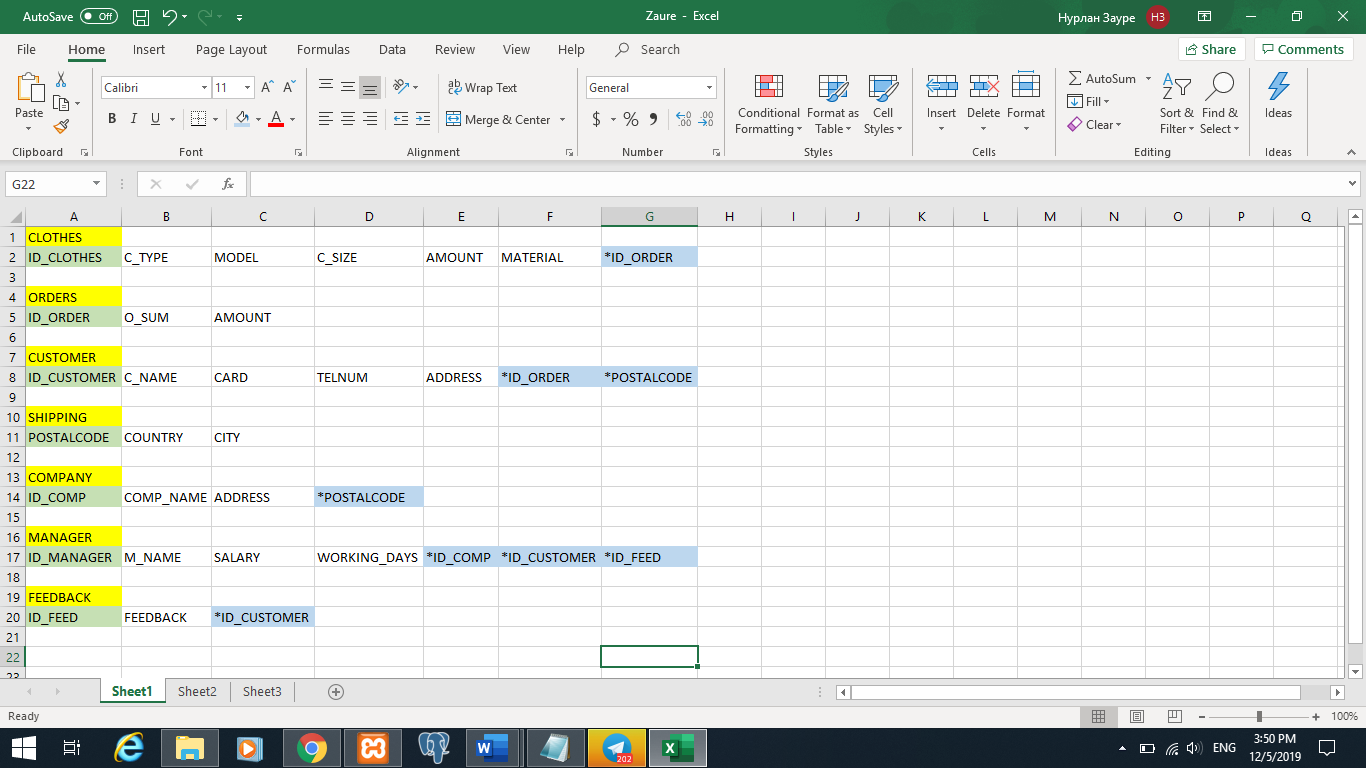
In 2 NF we got rid of partial functional dependence, but we still face some troubles.

Update: we can’t change information in feedback because we will have to change it everywhere.

Delete: we will have to delete everything if we want to delete some information in feedback.

Insert: we will have to insert information to other tables in order to add information to feedback table.

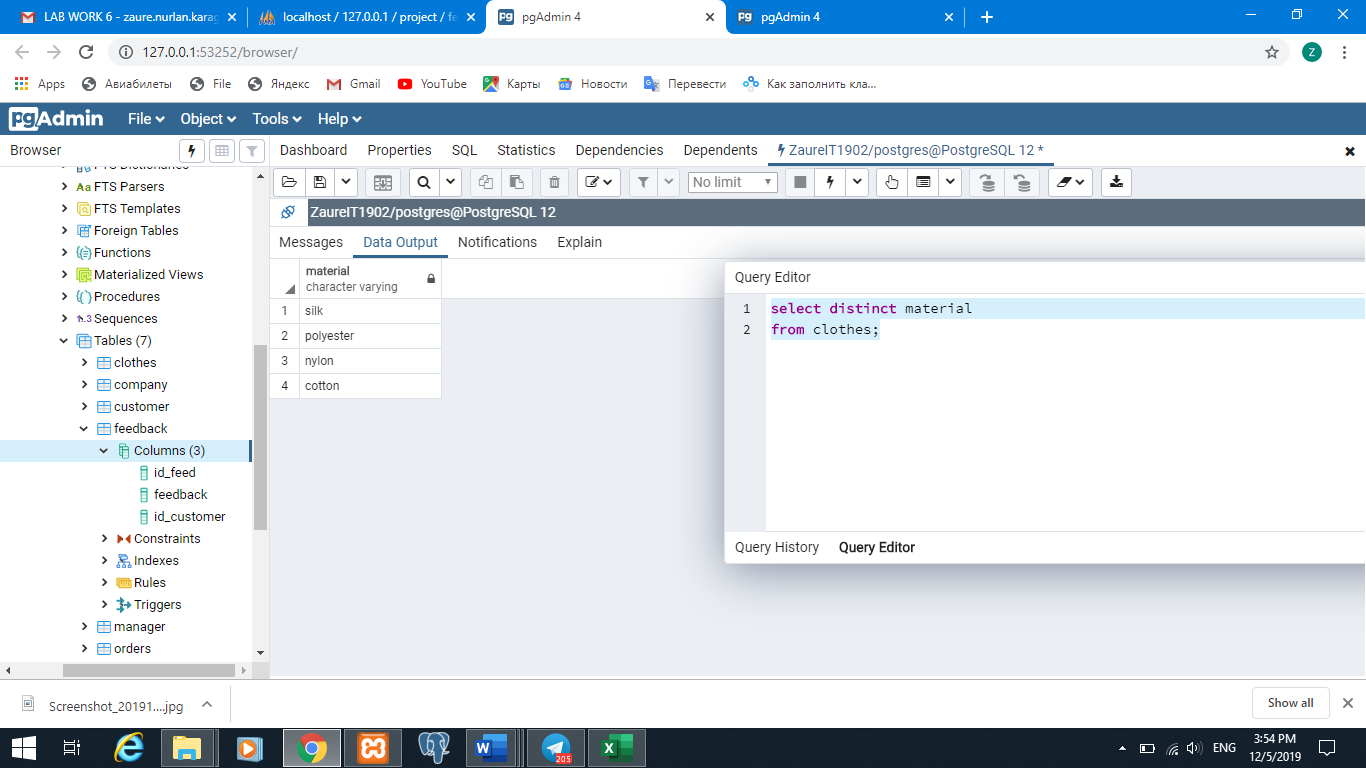
Third Normalization



Data is normalized because we got rid of transitive fuctional dependence. Therefore, there is no data redundancy.

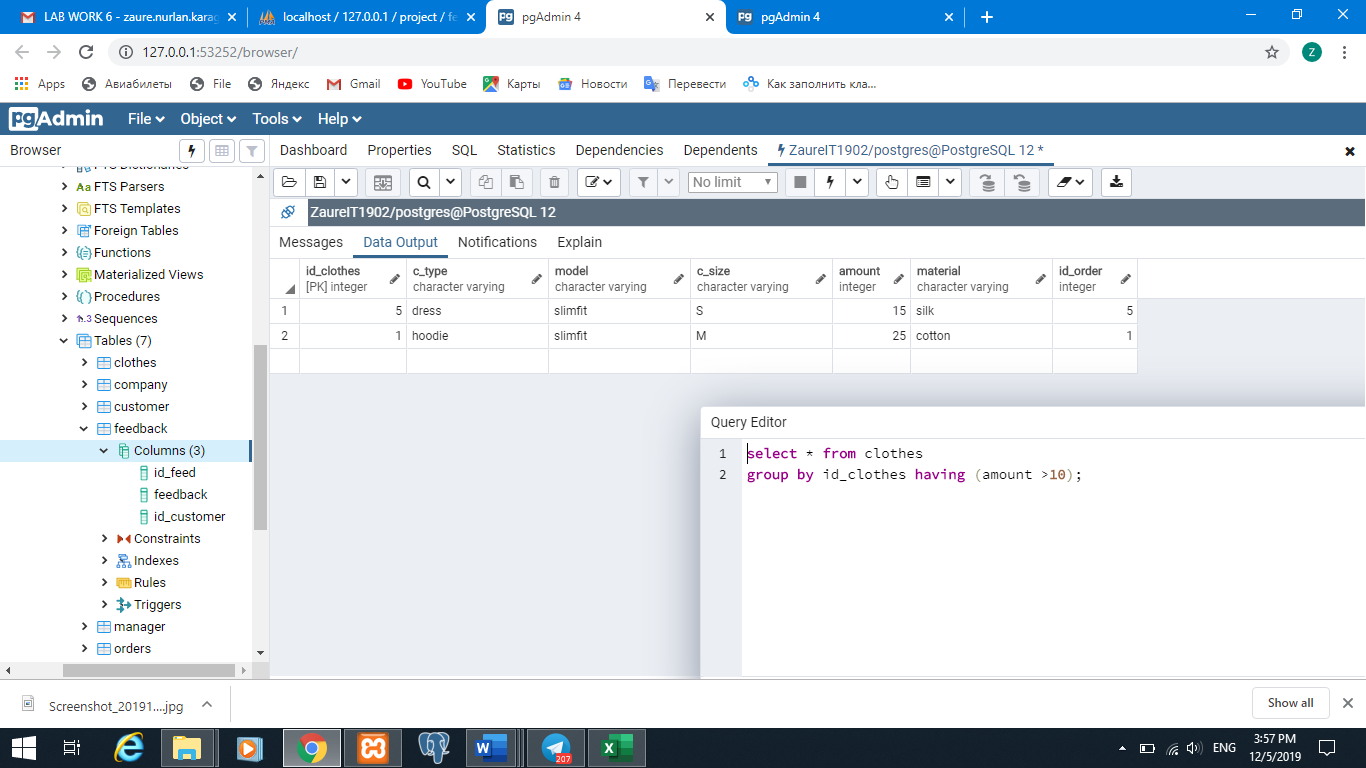
1. select distinct material

from clothes;



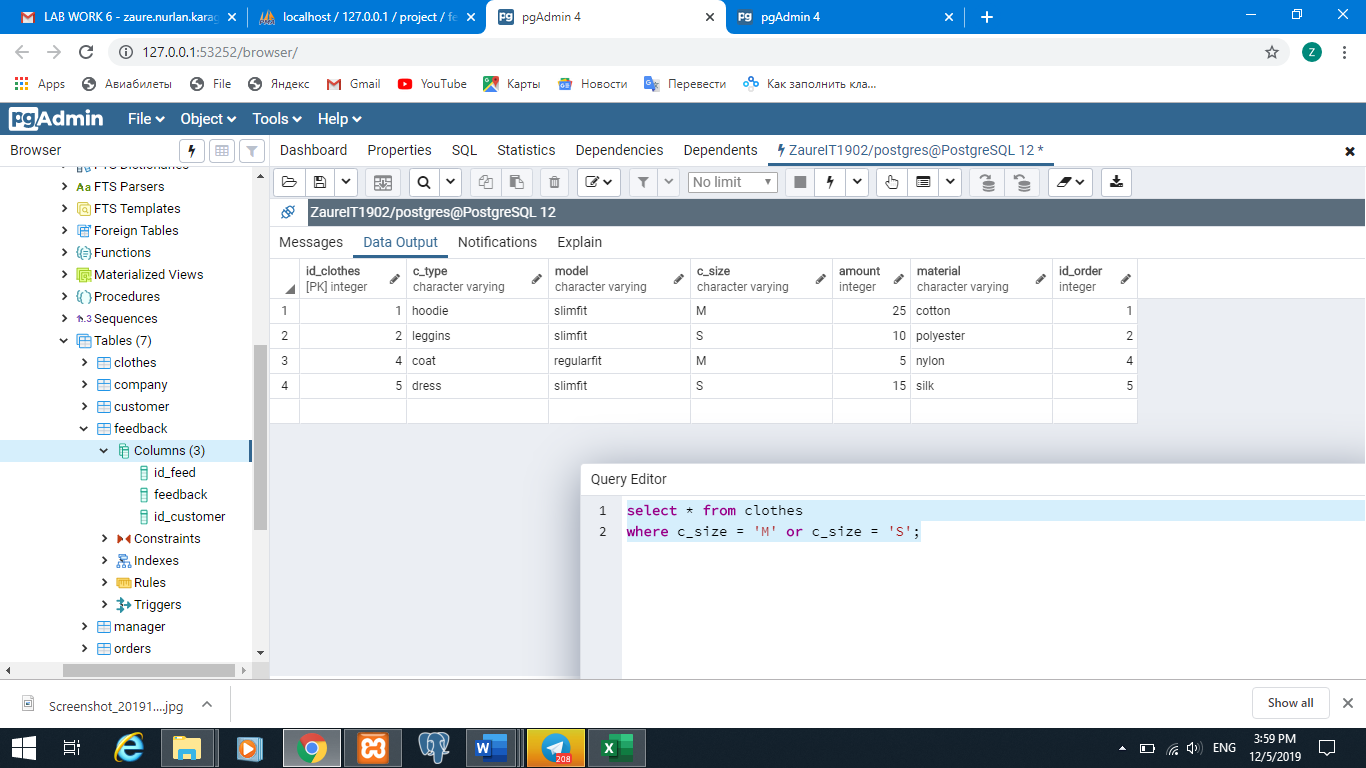
select \* from clothes

group by id\_clothes having (amount >10);



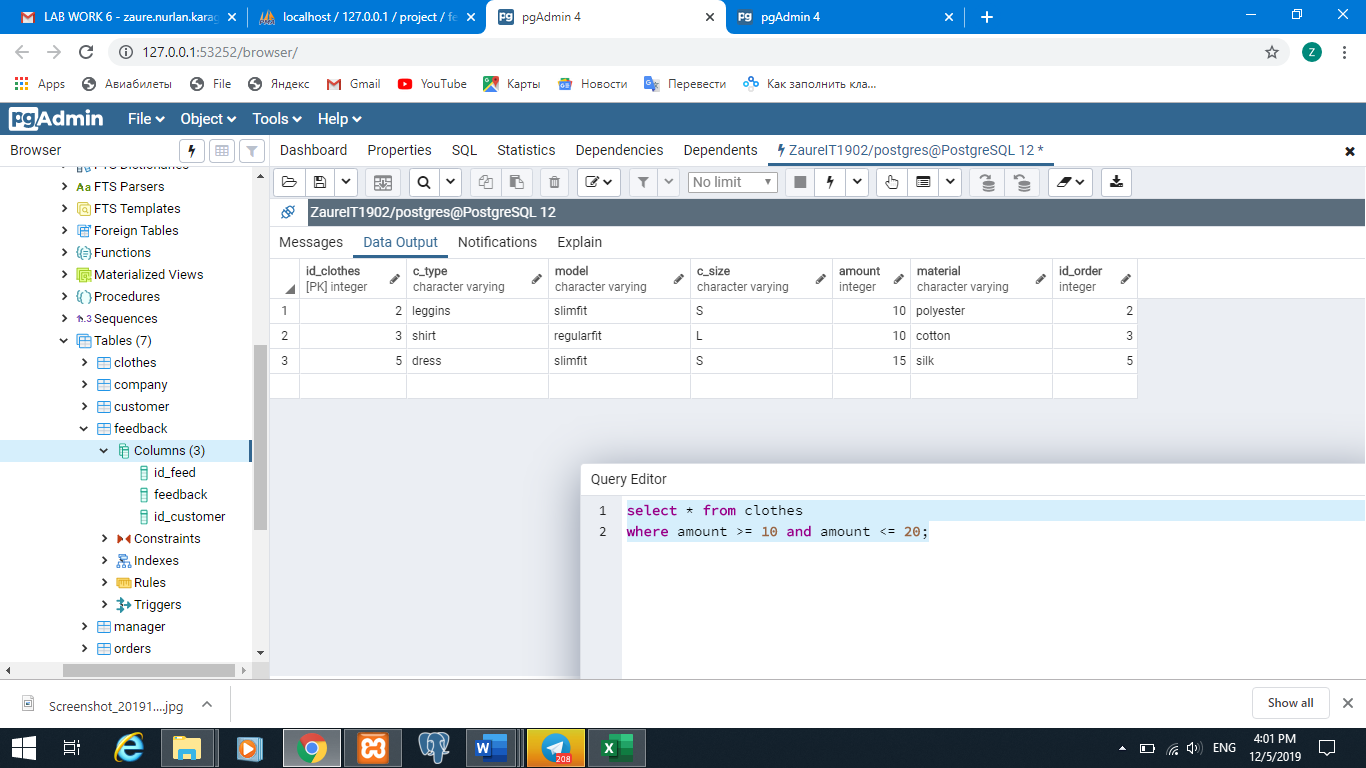
select \* from clothes

where c\_size = 'M' or c\_size = 'S';



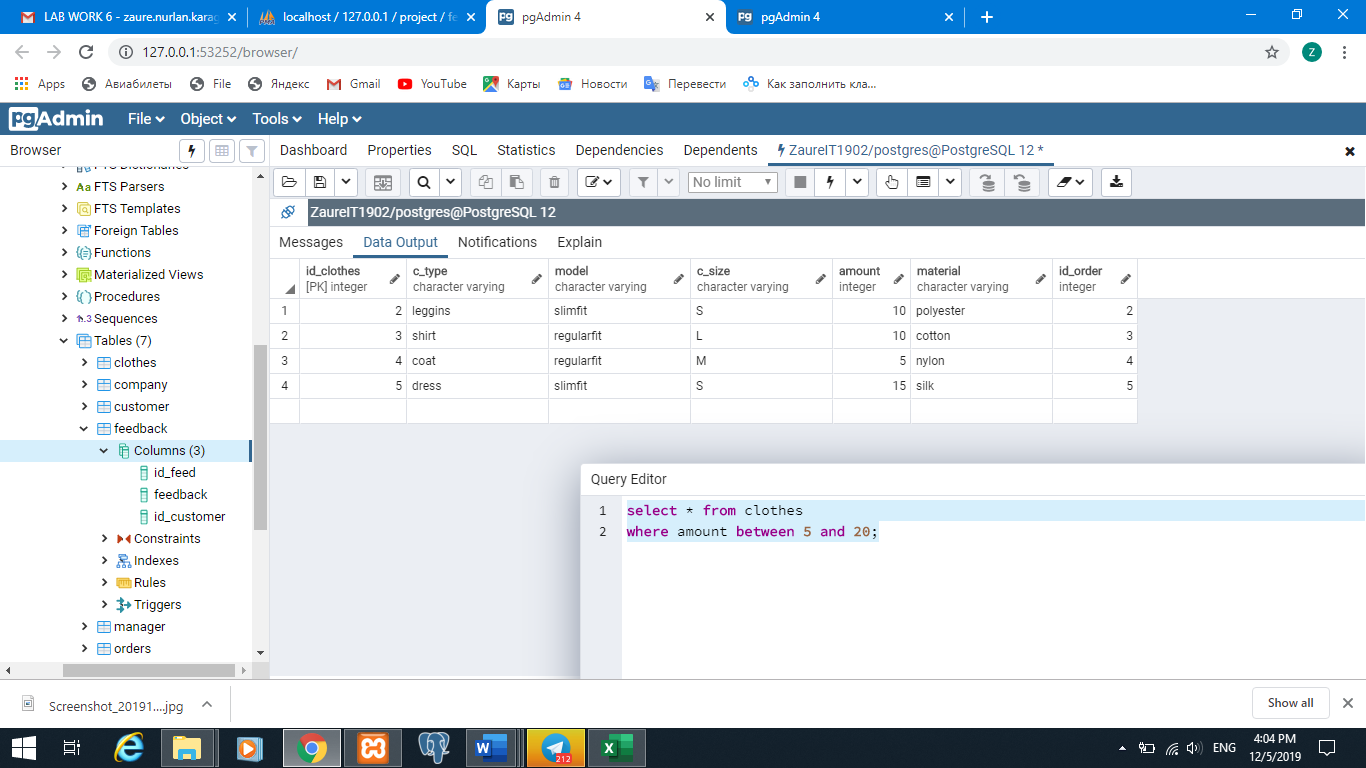
select \* from clothes

where amount >= 10 and amount <= 20;



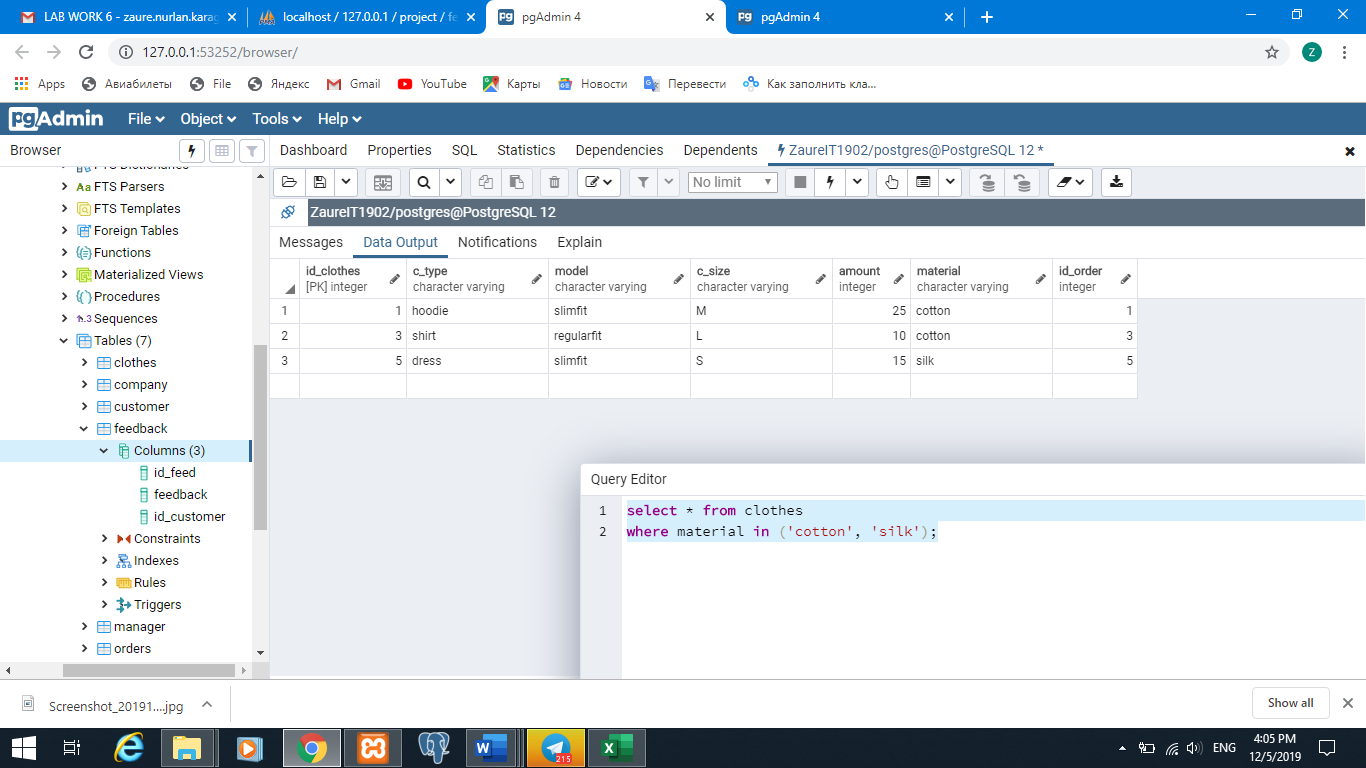
select \* from clothes

where amount between 5 and 20;



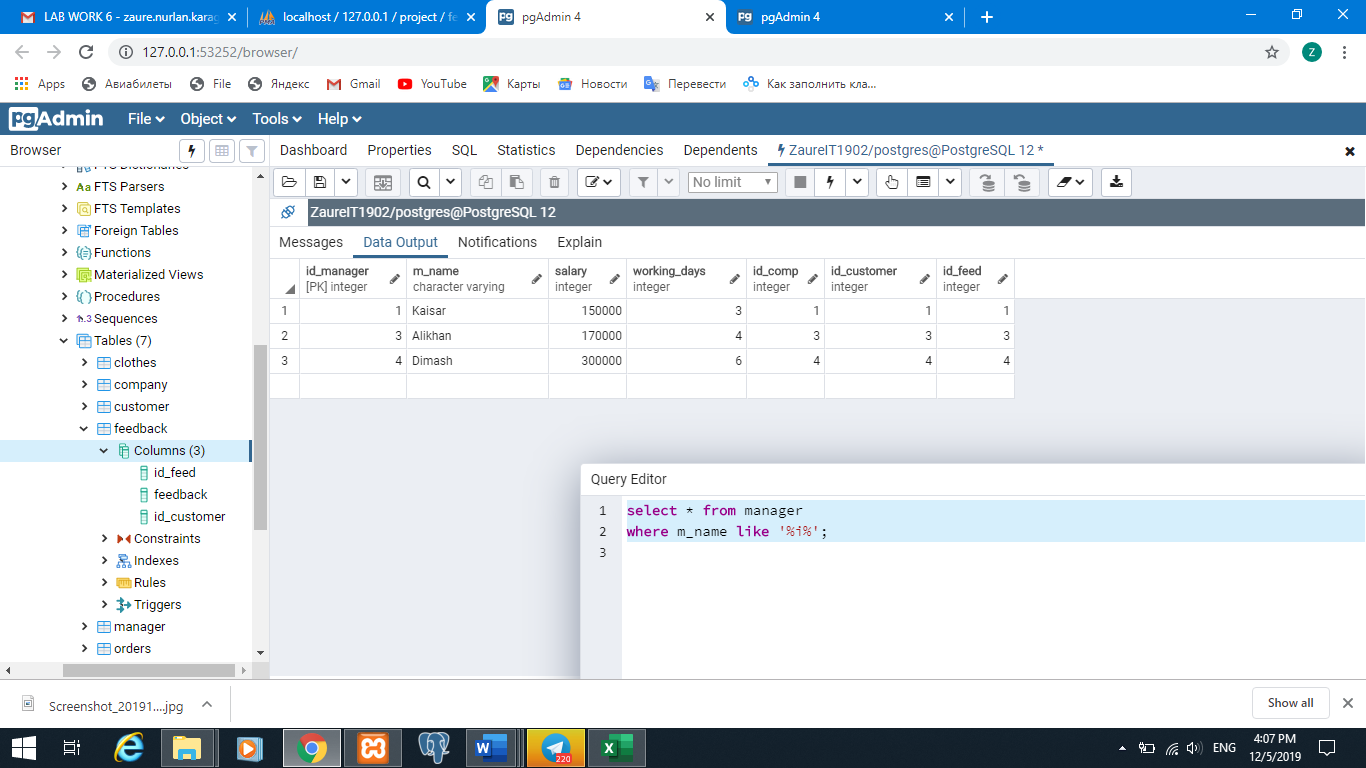
select \* from clothes

where material in ('cotton', 'silk');



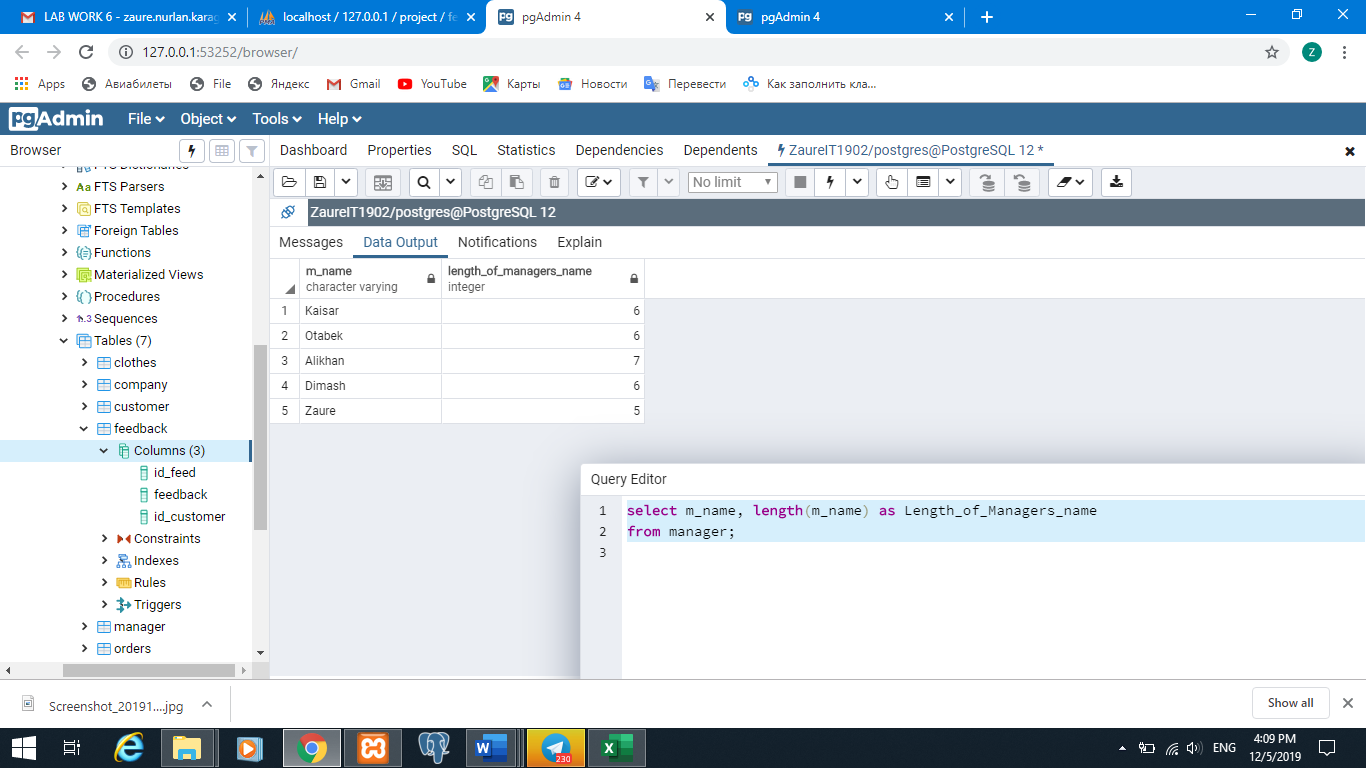
select \* from manager

where m\_name like '%i%';



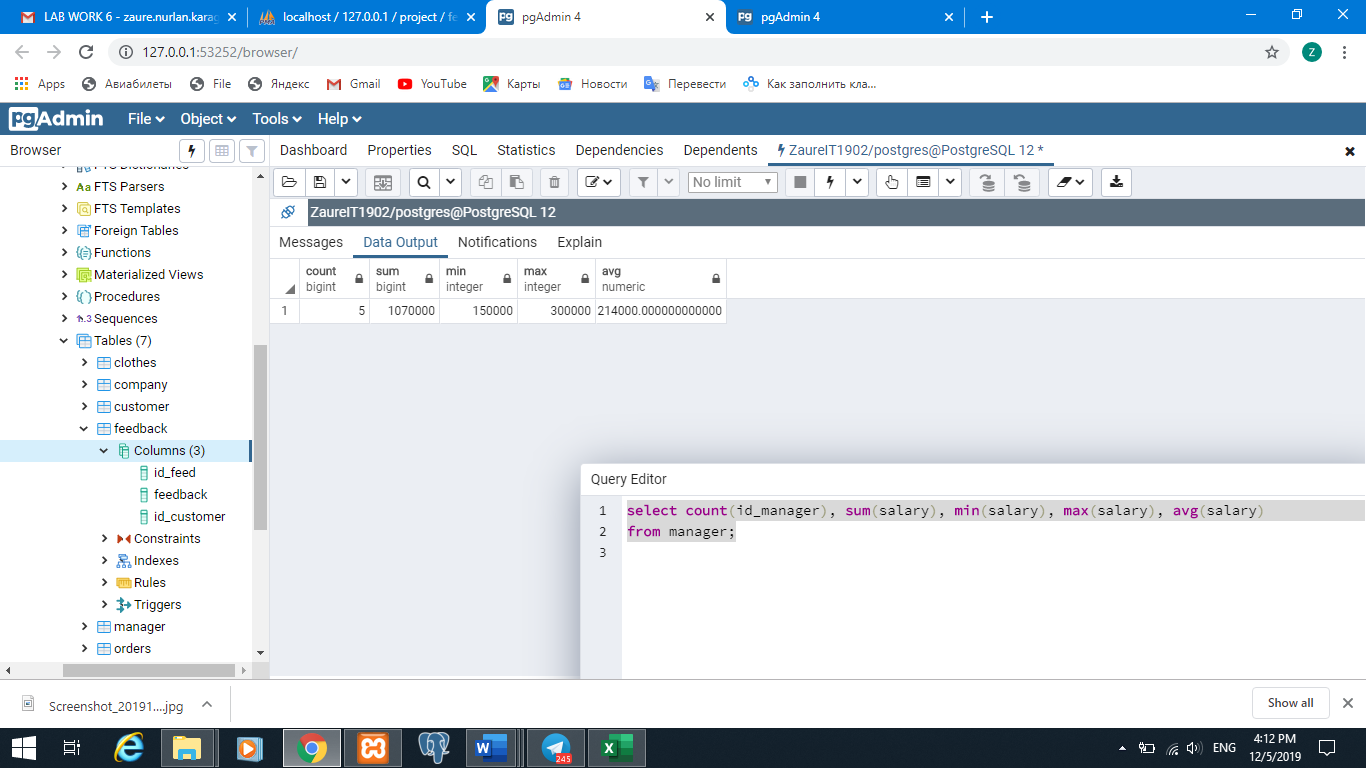
select m\_name, length(m\_name) as Length\_of\_Managers\_name

from manager;



select count(id\_manager), sum(salary), min(salary), max(salary), avg(salary)

from manager;



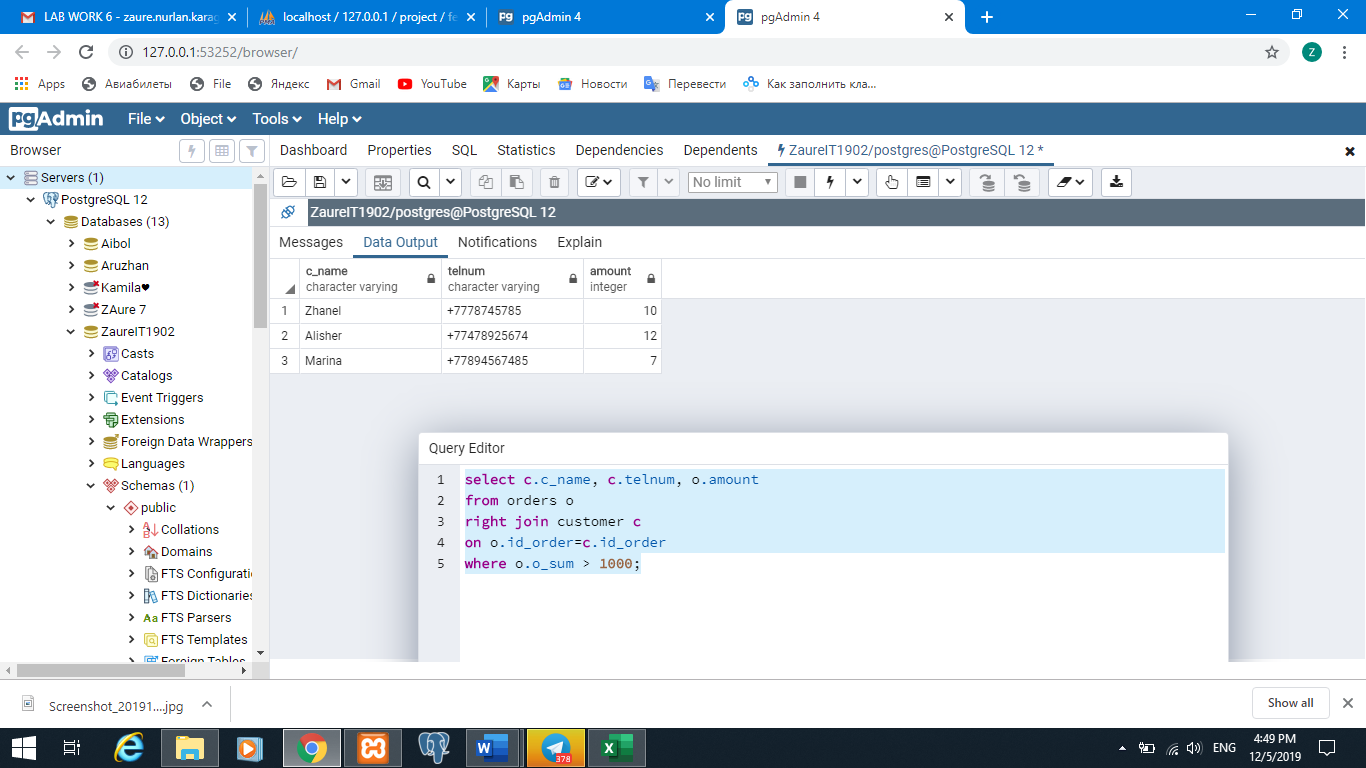
select c.c\_name, c.telnum, o.amount

from orders o

right join customer c

on o.id\_order=c.id\_order

where o.o\_sum > 1000;



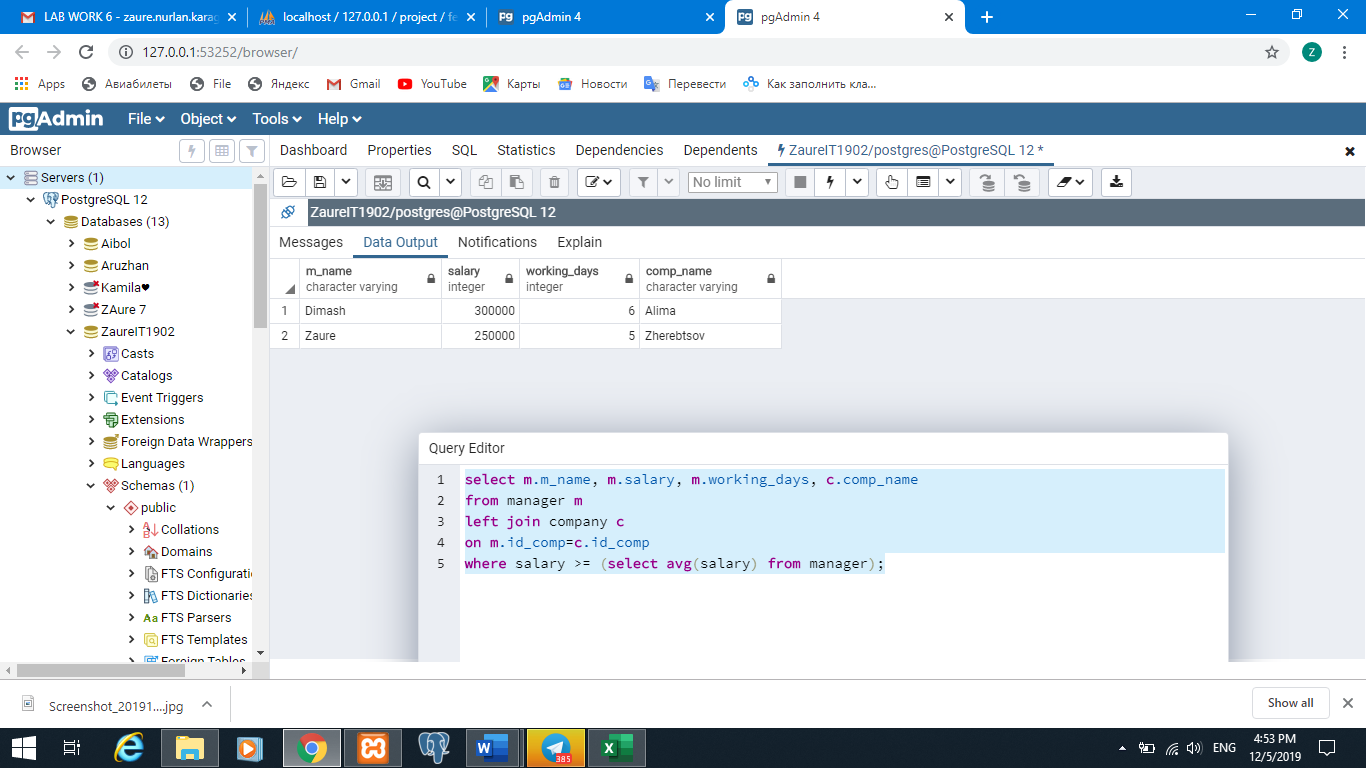
select m.m\_name, m.salary, m.working\_days, c.comp\_name

from manager m

left join company c

on m.id\_comp=c.id\_comp

where salary >= (select avg(salary) from manager);



select f.feedback, m.m\_name, m.working\_days

from feedback f

left join manager m

on f.id\_feed=m.id\_feed

where salary >= (select avg(salary) from manager);

select c.c\_name, f.feedback, m.m\_name, m.working\_days

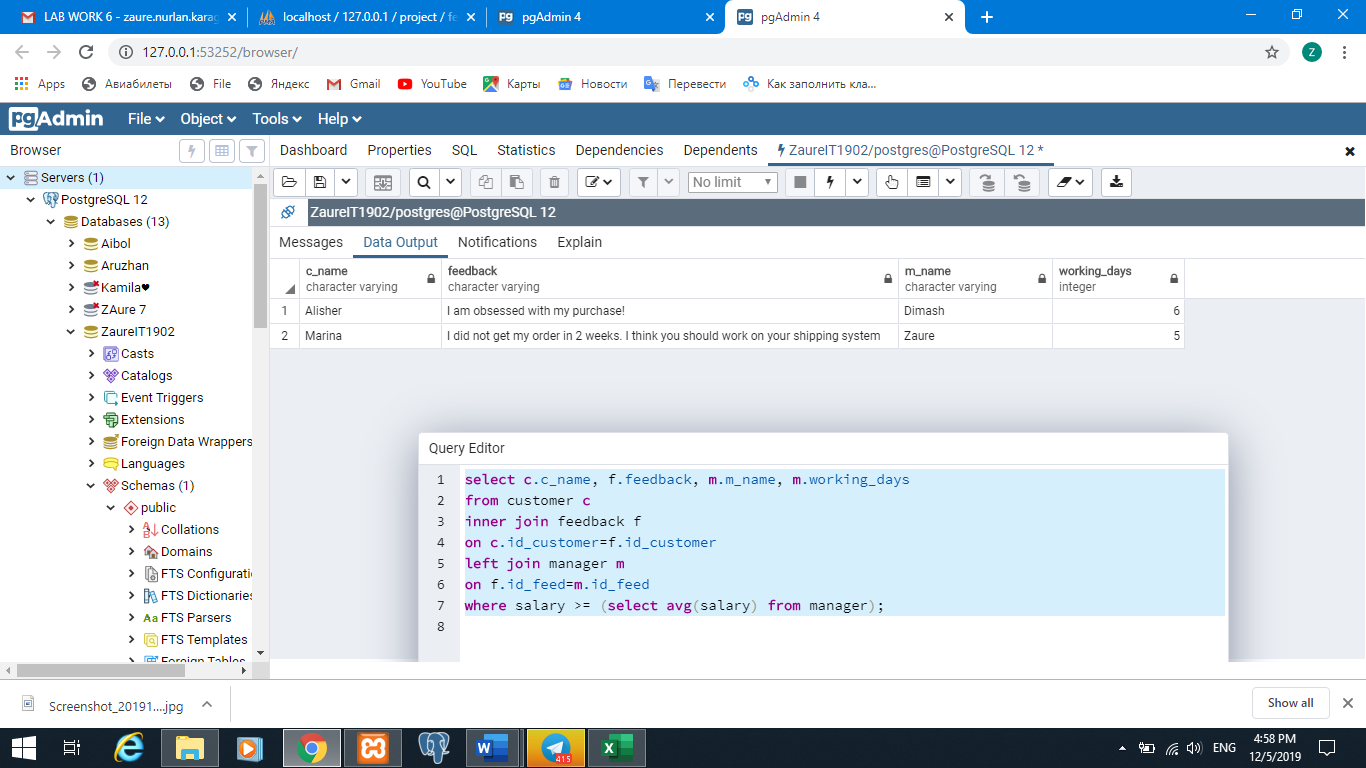
from customer c

inner join feedback f

on c.id\_customer=f.id\_customer

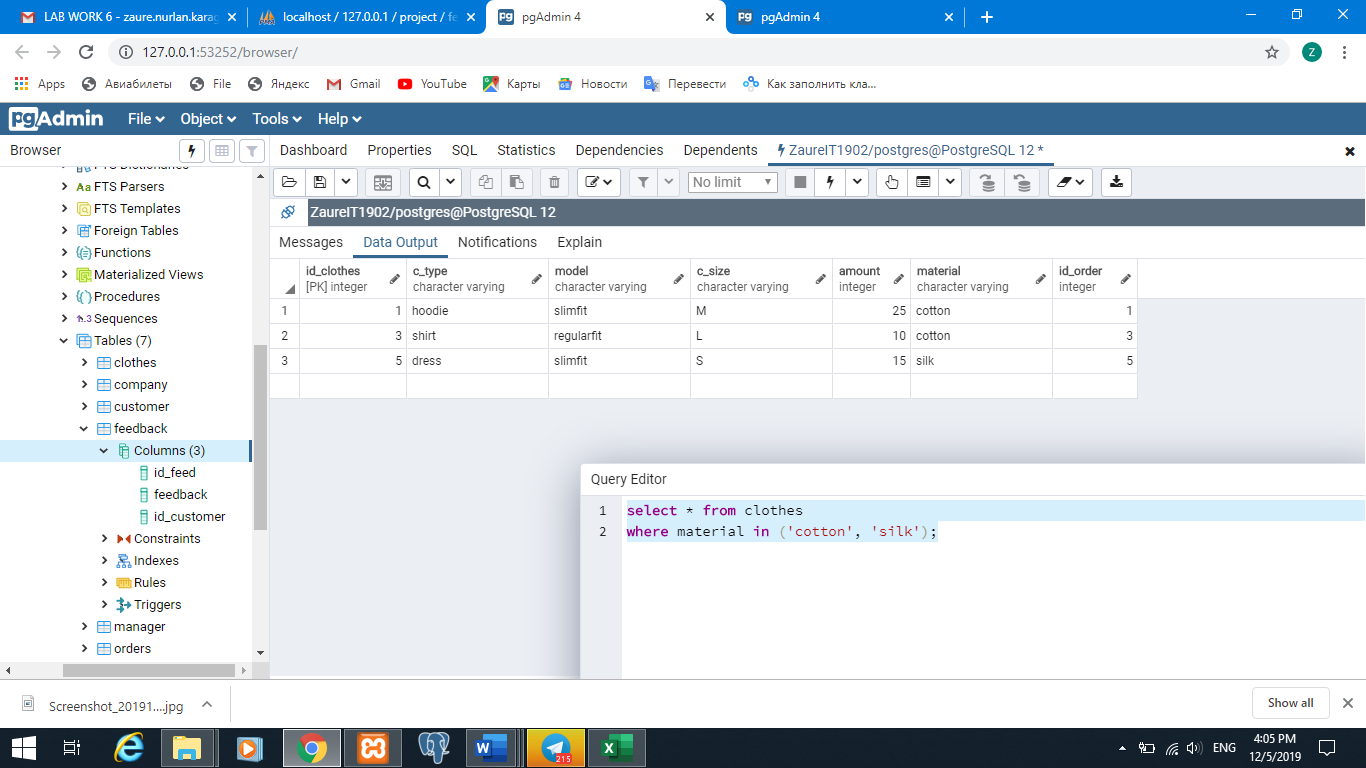
left join manager m

on f.id\_feed=m.id\_feed

where salary >= (select avg(salary) from manager);

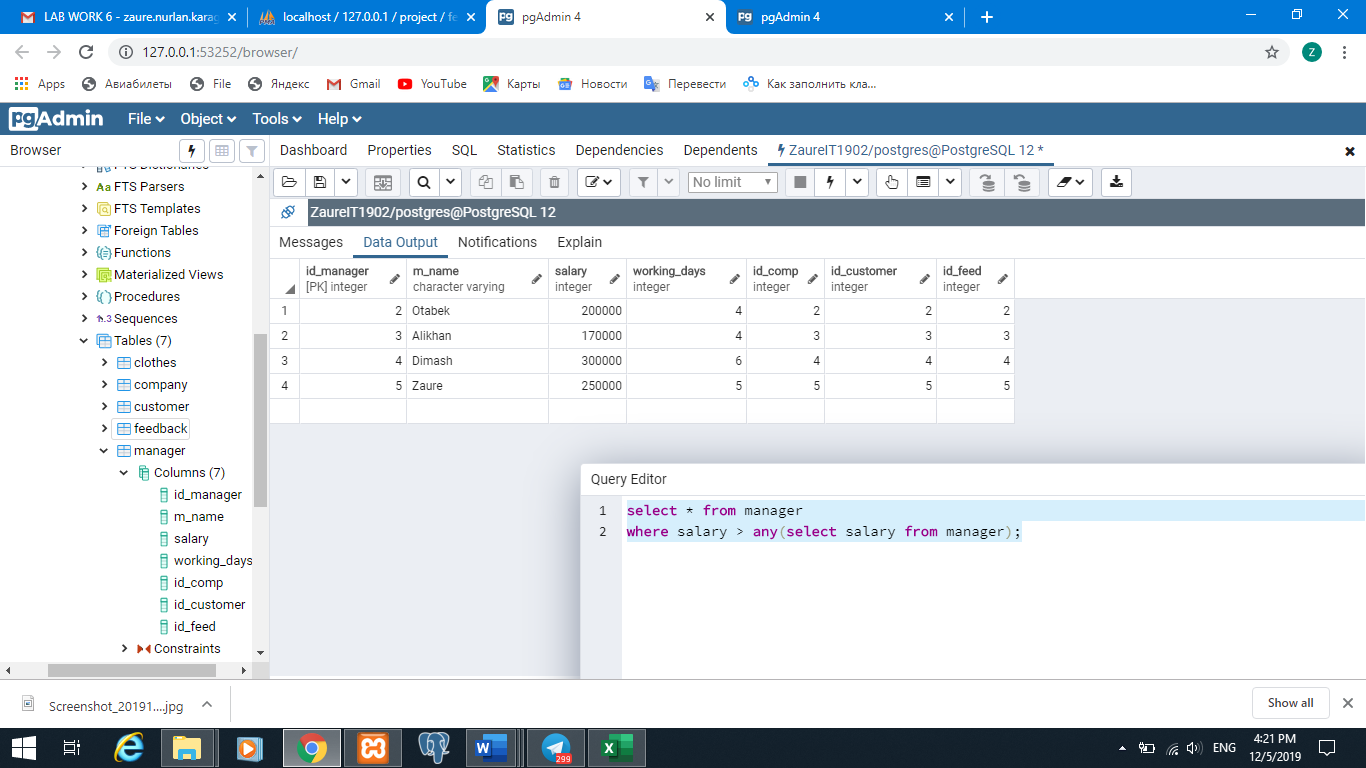
1. select \* from clothes

where material in ('cotton', 'silk');



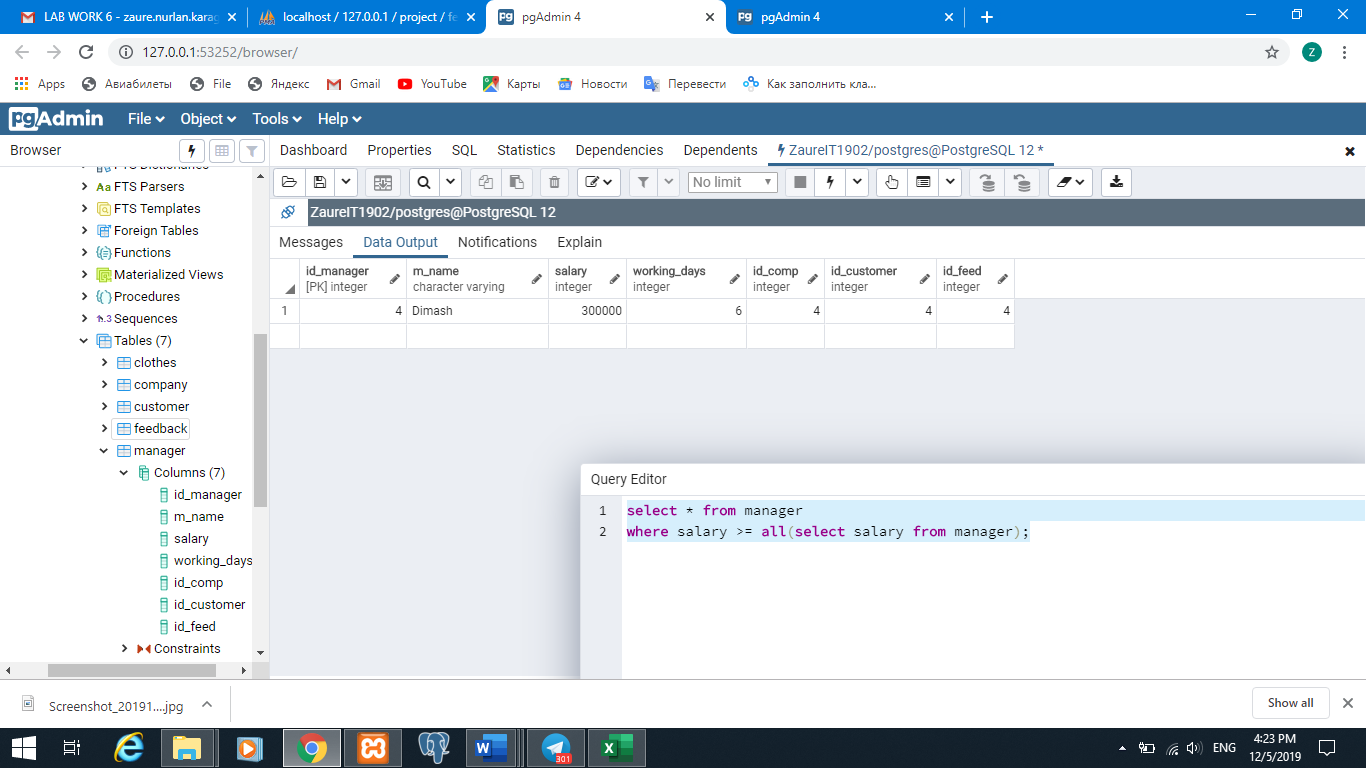
select \* from manager

where salary > any(select salary from manager);



select \* from manager

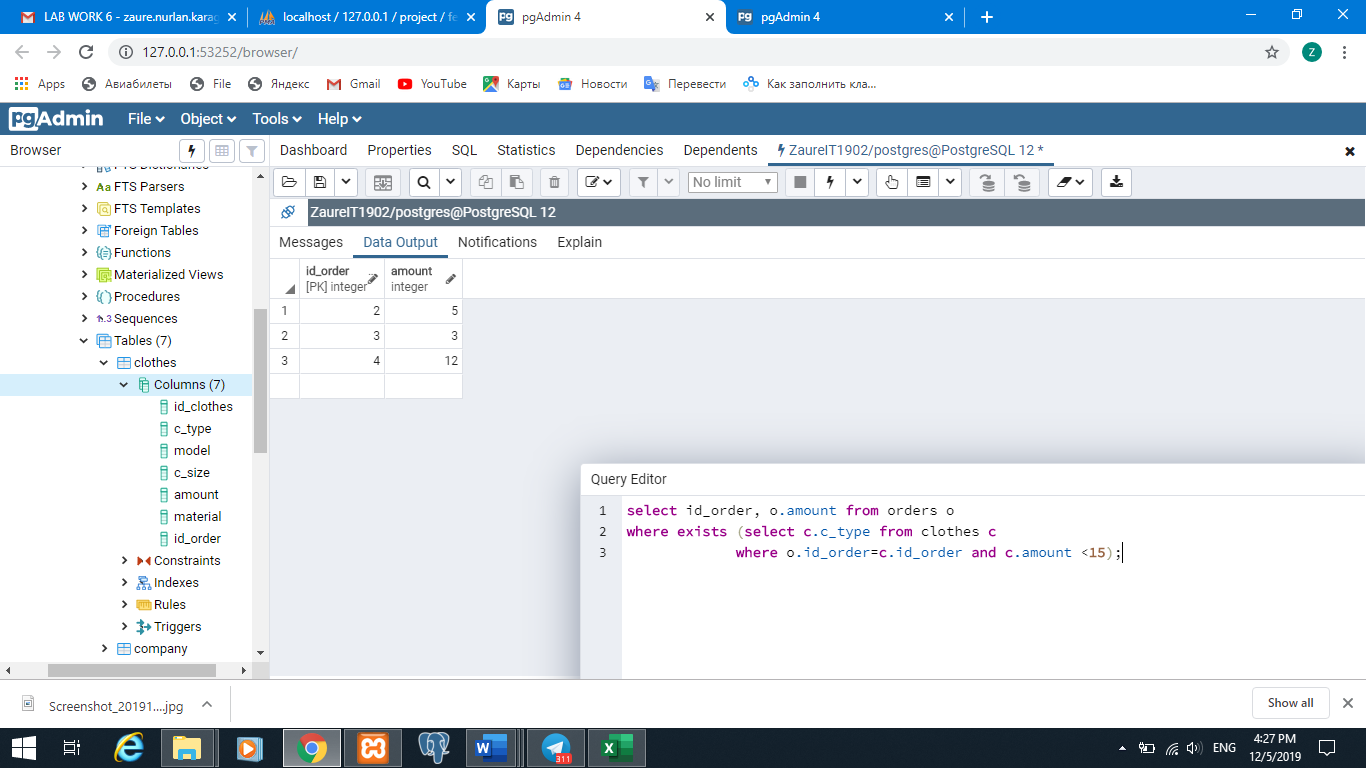
where salary >= all(select salary from manager);



select id\_order, o.amount from orders o

where exists (select c.c\_type from clothes c

where o.id\_order=c.id\_order and c.amount <15);



select id\_order, o.amount from orders o

where not exists (select c.c\_type from clothes c

where o.id\_order=c.id\_order and c.amount <15);

