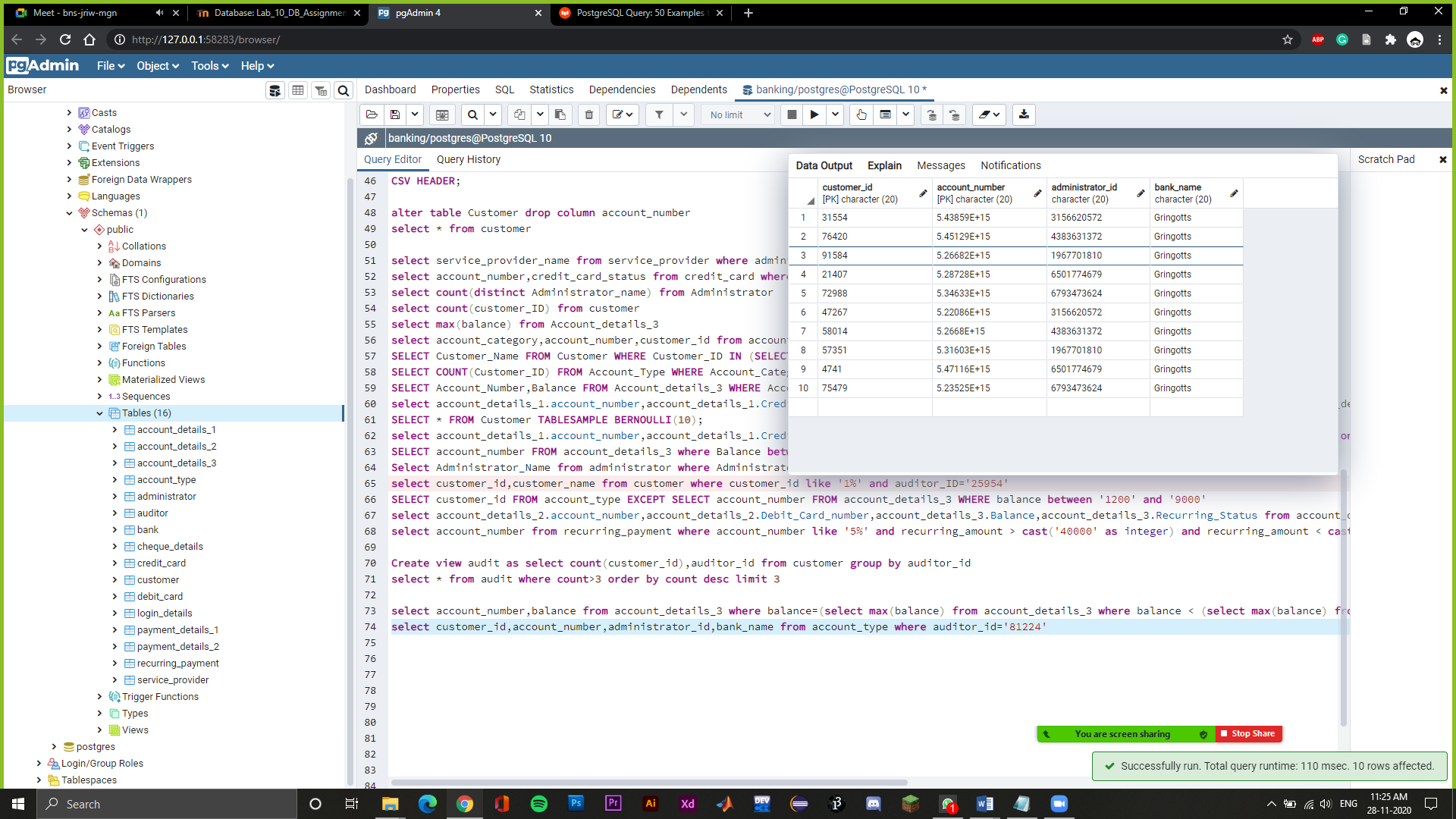
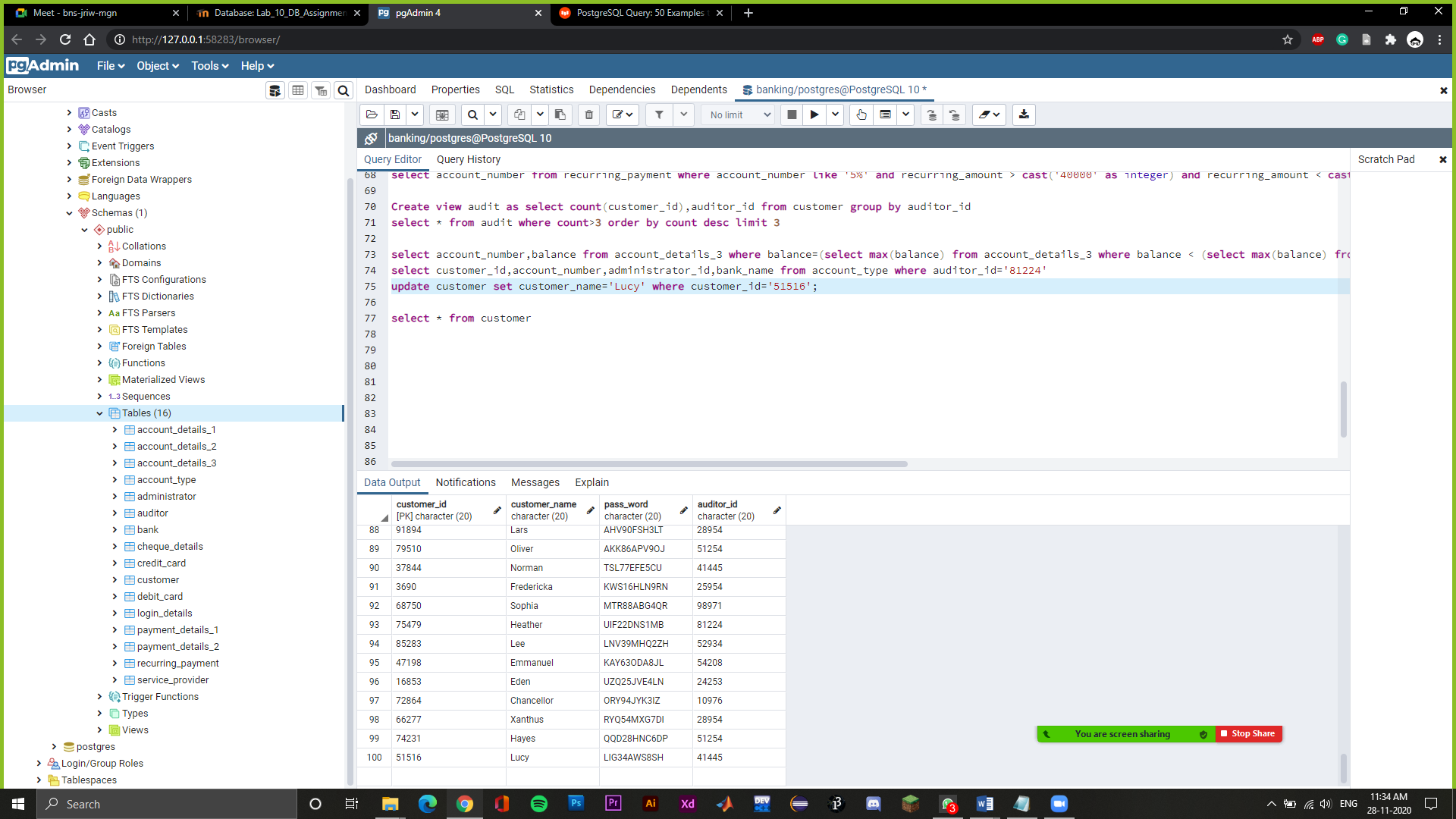
**select customer\_id,account\_number,administrator\_id,bank\_name from account\_type where auditor\_id='81224'**

->Showing customer id, account number, administrator id, bank name when auditor id is 81224.



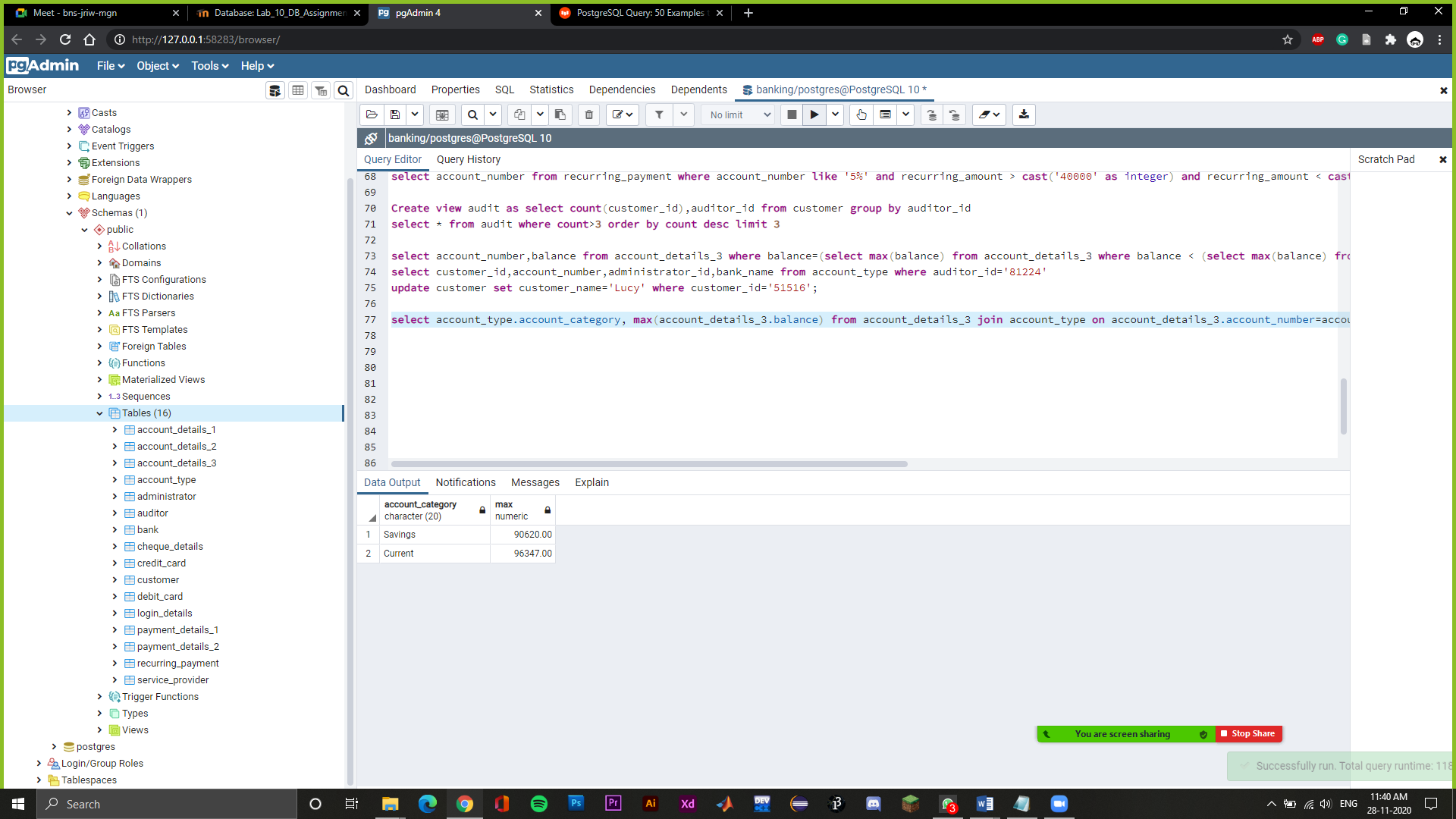
**update customer set customer\_name='Lucy' where customer\_id='51516';**

->Update customer name as lucy whose customer id is 51516



**select account\_type.account\_category, max(account\_details\_3.balance) from account\_details\_3 join account\_type on account\_details\_3.account\_number=account\_type.account\_number group by account\_type.account\_category**

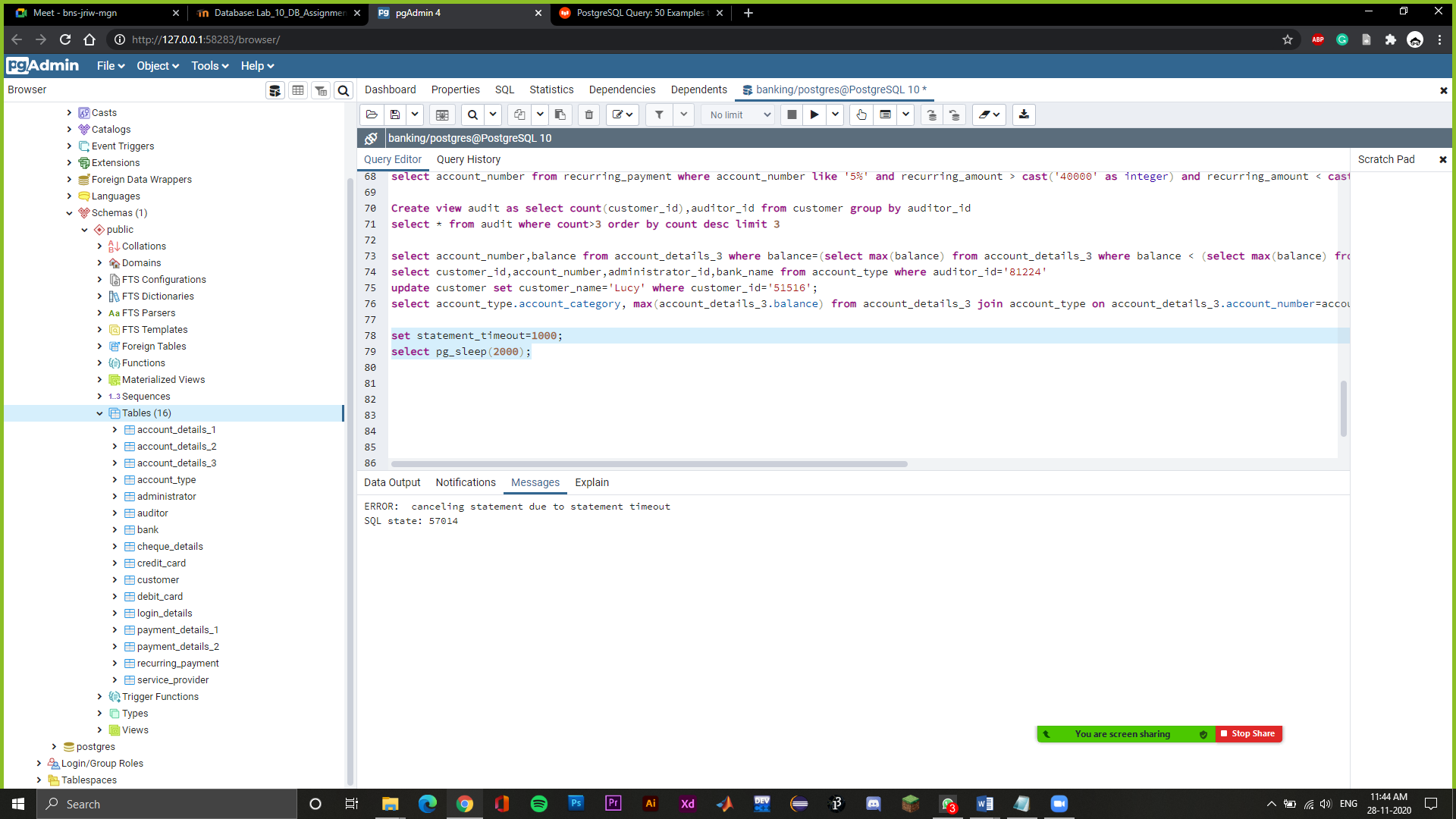
->Show highest balance of each account category



**set statement\_timeout=1000;**

**select pg\_sleep(2000);**

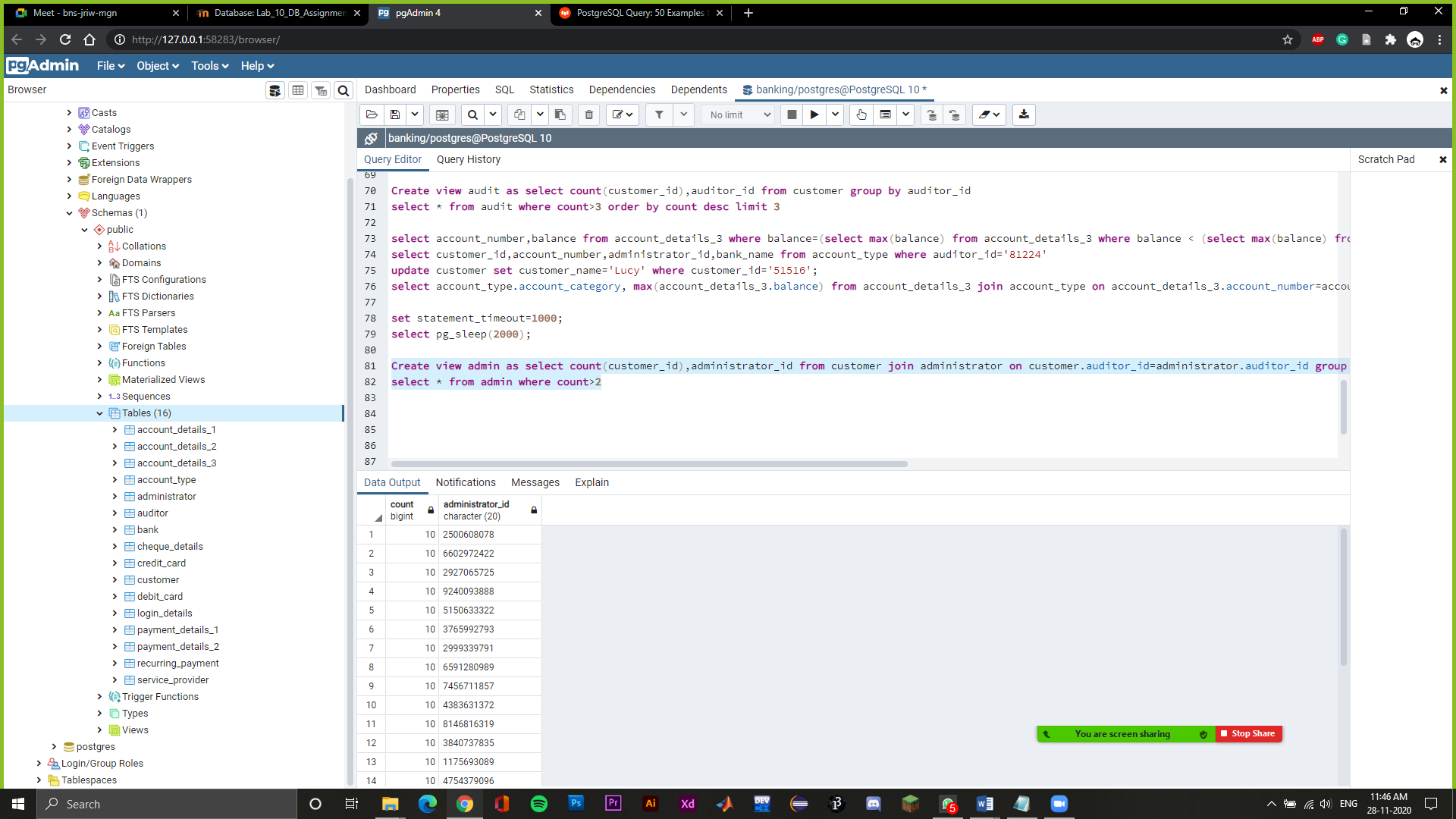
->Set max query time as 1000 ms



**Create view admin as select count(customer\_id),administrator\_id from customer join administrator on customer.auditor\_id=administrator.auditor\_id group by administrator\_id**

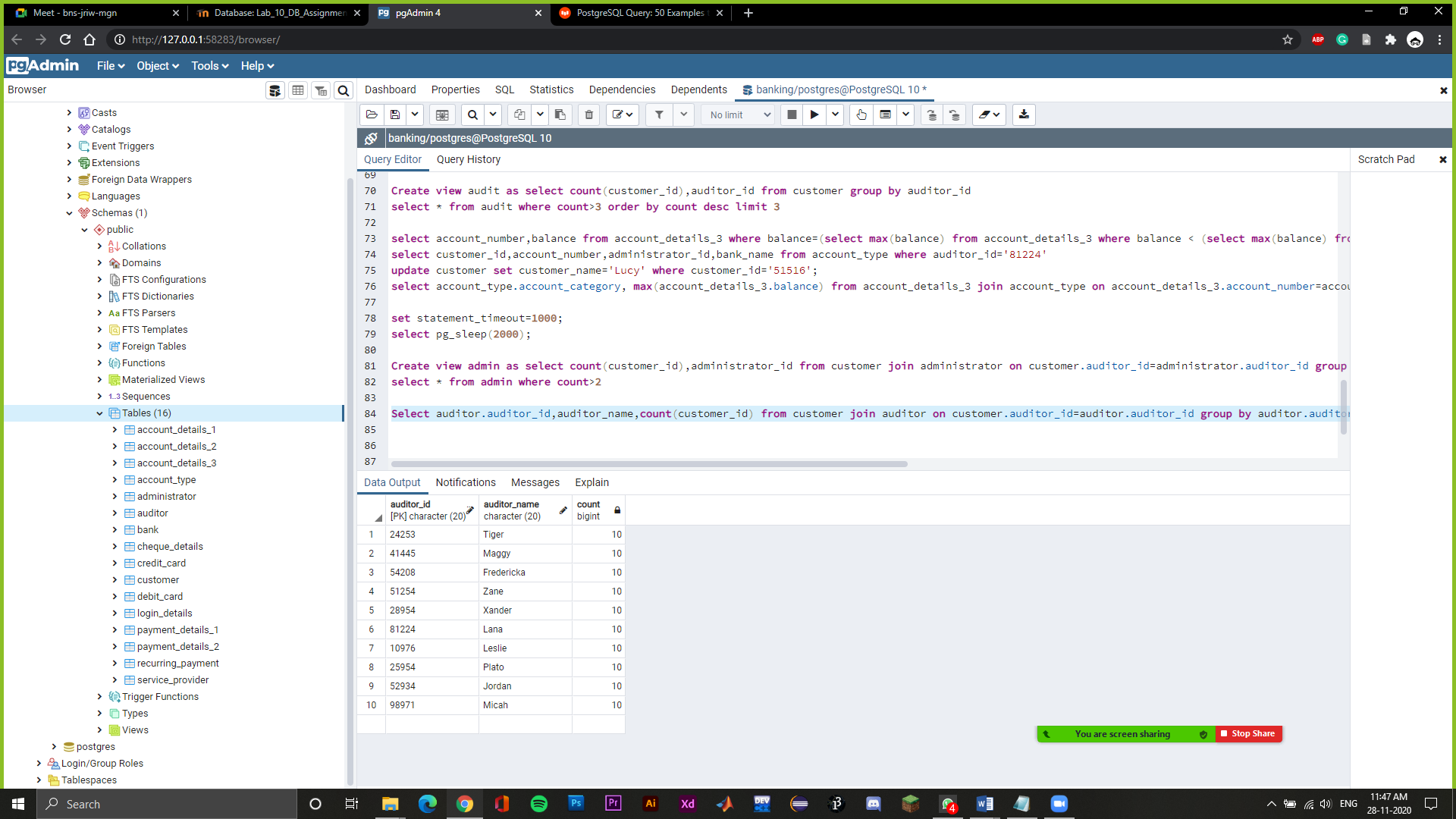
**select \* from admin where count>2**

->Created a view named admin which shows administrator id and count of customer id it serves and below that we have printed down where count is greater than 2.



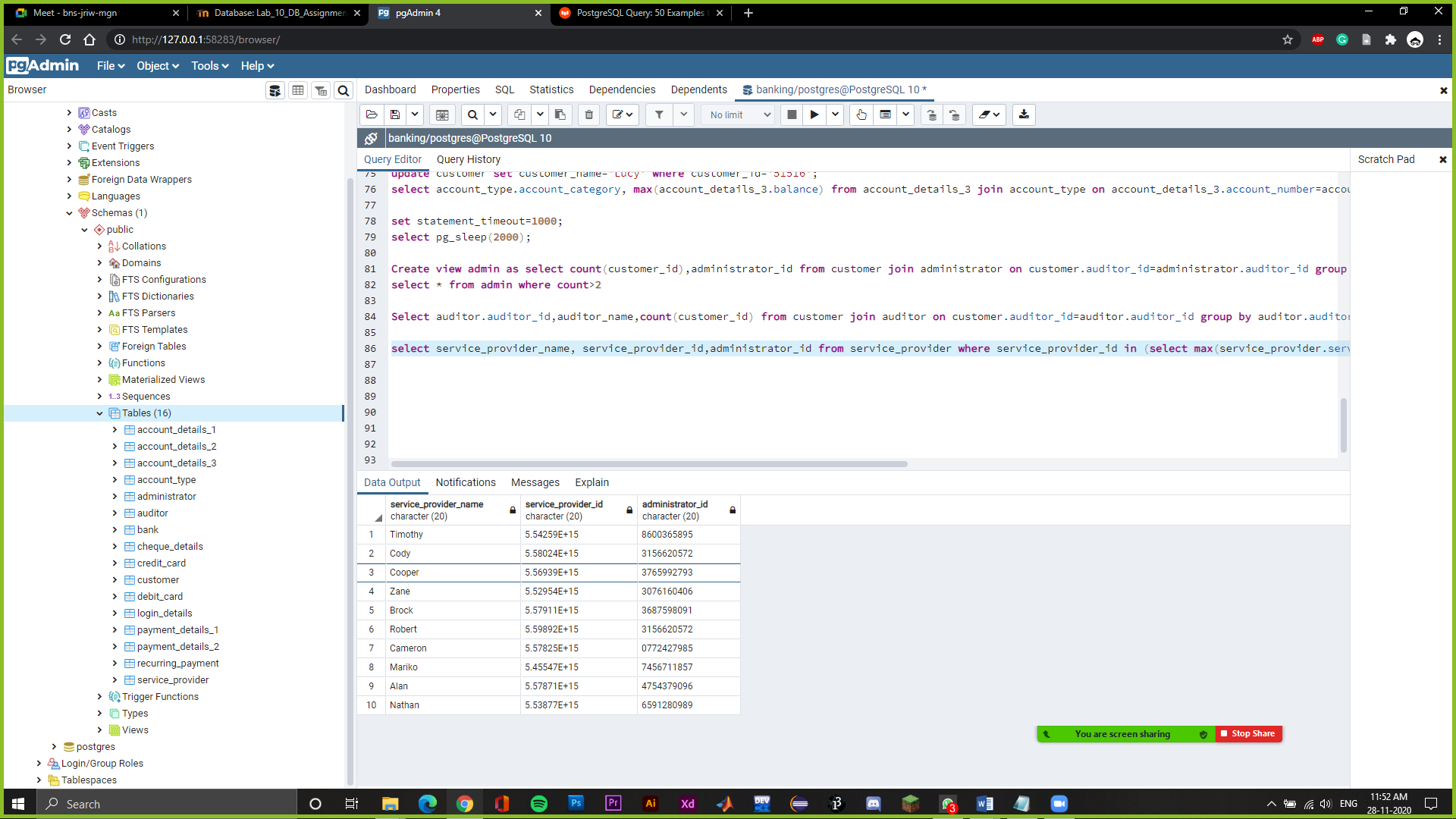
**Select auditor.auditor\_id,auditor\_name,count(customer\_id) from customer join auditor on customer.auditor\_id=auditor.auditor\_id group by auditor.auditor\_id**

->Showing auditor id, auditor name and count of the customer id by performing join operation.



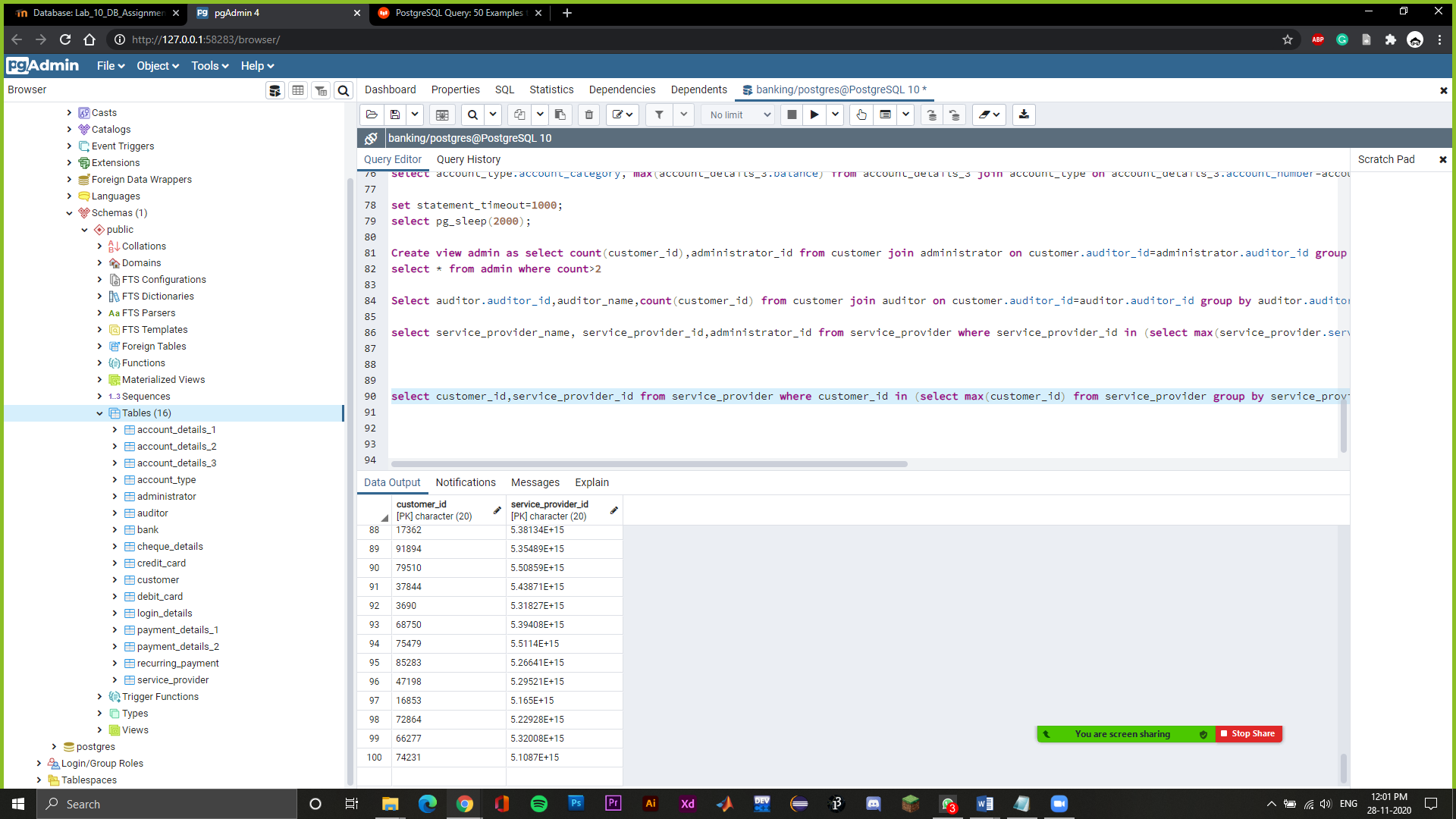
**select service\_provider\_name, service\_provider\_id,administrator\_id from service\_provider where service\_provider\_id in (select max(service\_provider.service\_provider\_id) from service\_provider group by auditor\_id)**

->Print service provider name, its ID, corresponding administrator ID where the service provider ID is maximum corresponding to its auditor.



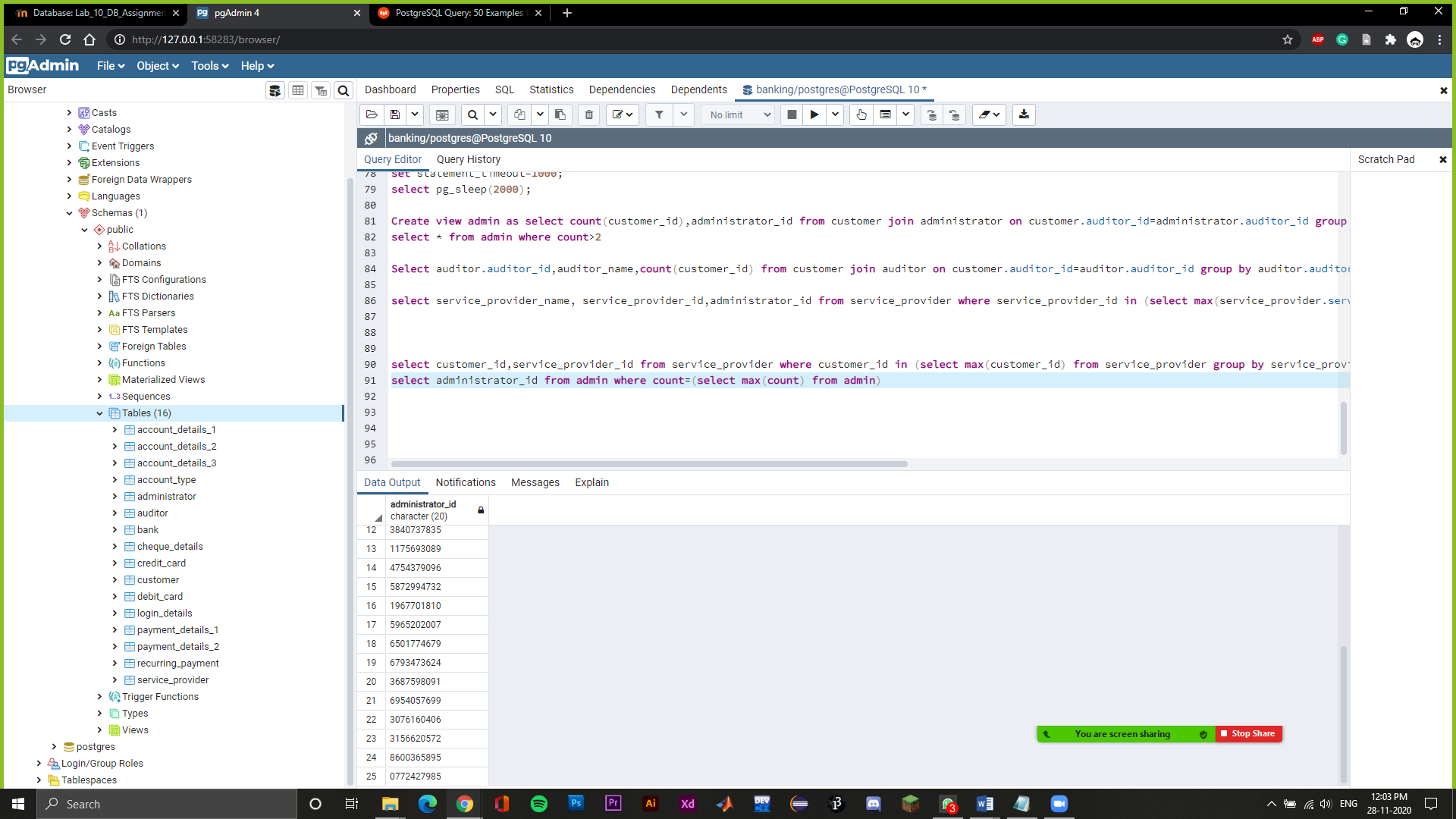
**select customer\_id,service\_provider\_id from service\_provider where customer\_id in (select max(customer\_id) from service\_provider group by service\_provider\_id)**

->Select Maximum Customer ID from the pool of customers who are being serviced by a certain service provider and show the corresponding service provider ID as well



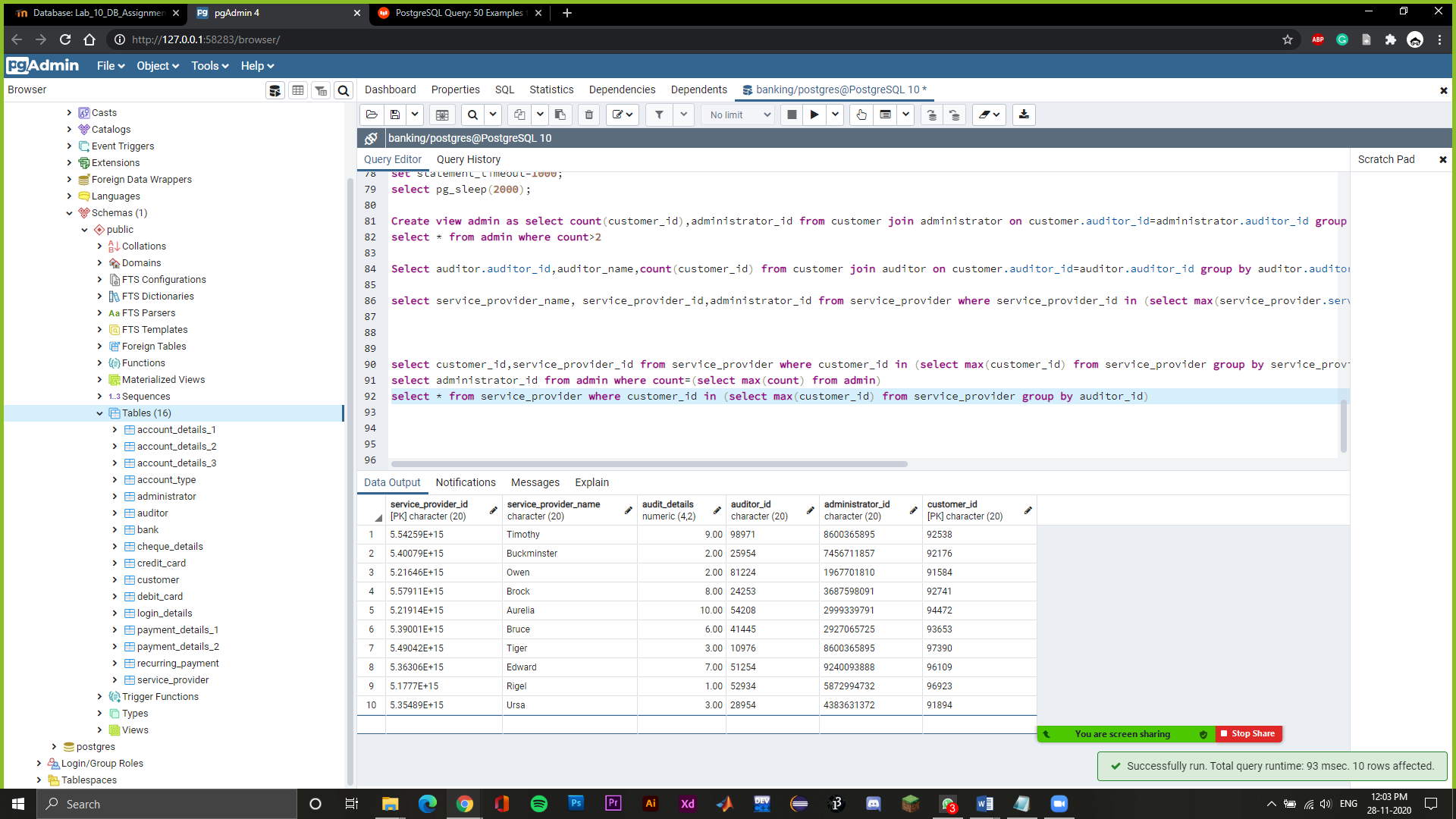
**select administrator\_id from admin where count=(select max(count) from admin)**

->Print ID’s of the administrator for those who serve the maximum number of customers.



**select \* from service\_provider where customer\_id in (select max(customer\_id) from service\_provider group by auditor\_id)**

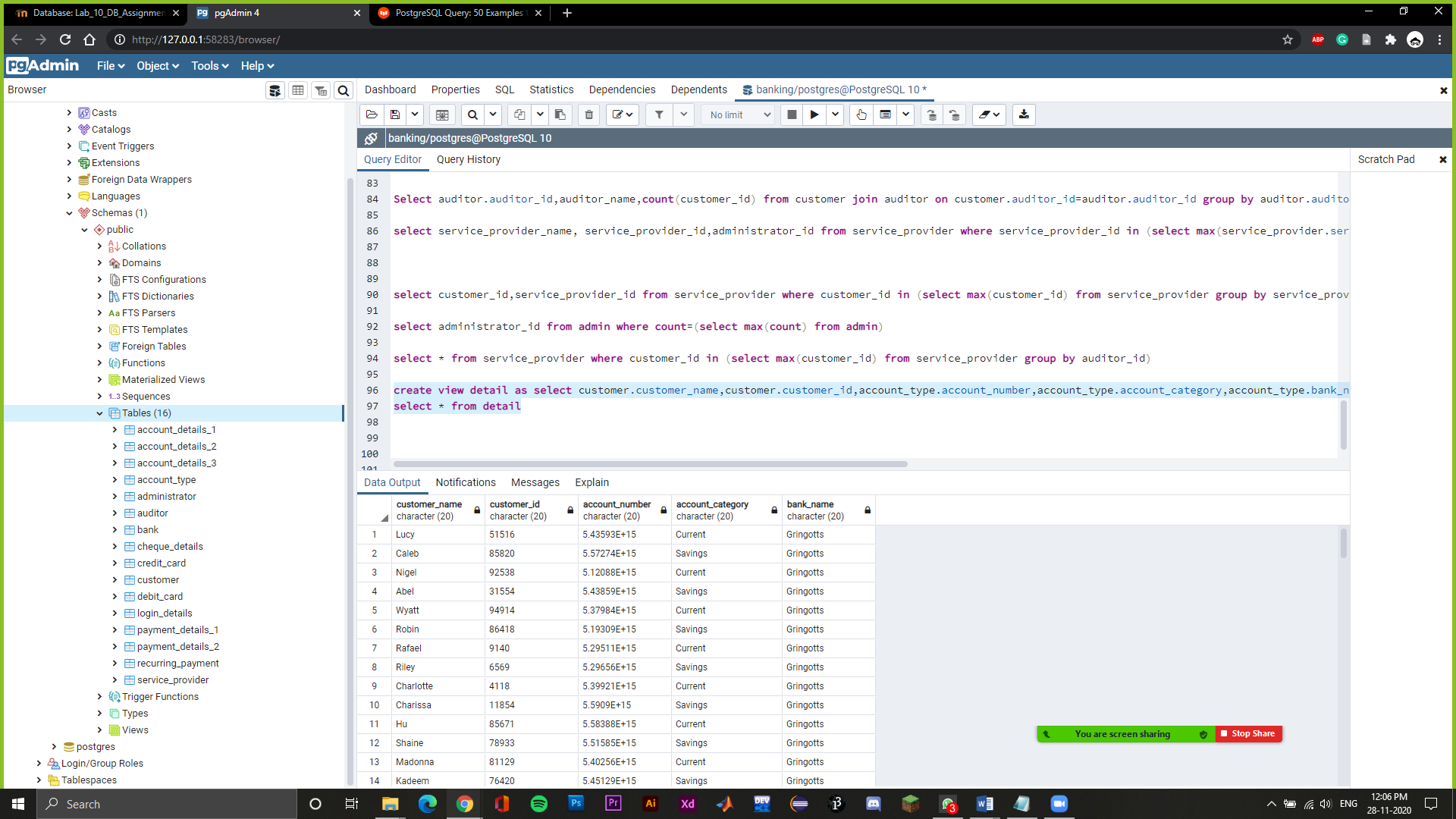
->Select maximum Customer ID along with its details corresponding to an Auditor



**create view detail as select customer.customer\_name,customer.customer\_id,account\_type.account\_number,account\_type.account\_category,account\_type.bank\_name from customer join account\_type on customer.customer\_id=account\_type.customer\_id**

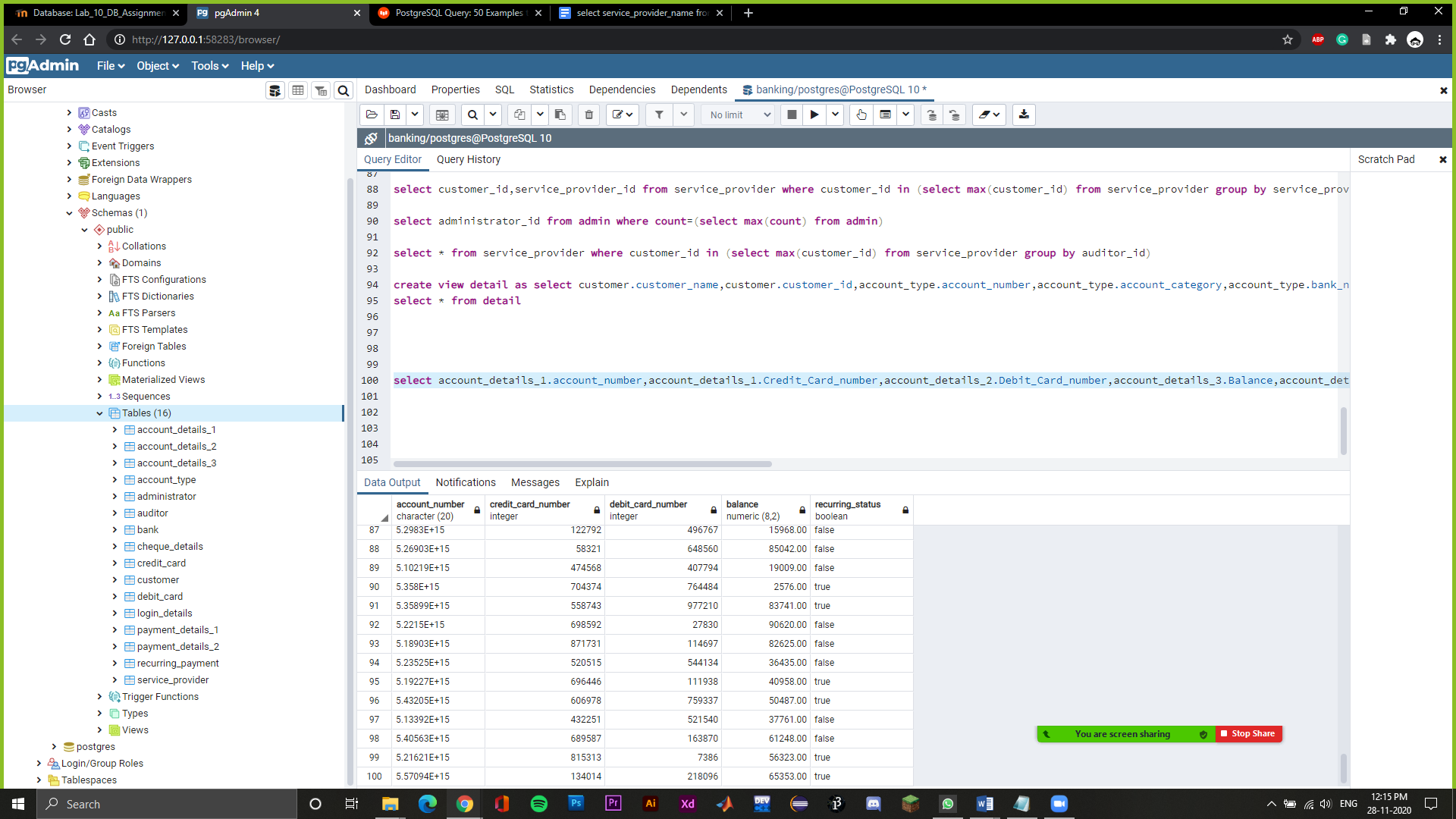
**select \* from detail**

->Create a view which shows all the information of a customer like customer name, customer id, account number, account category and bank name.



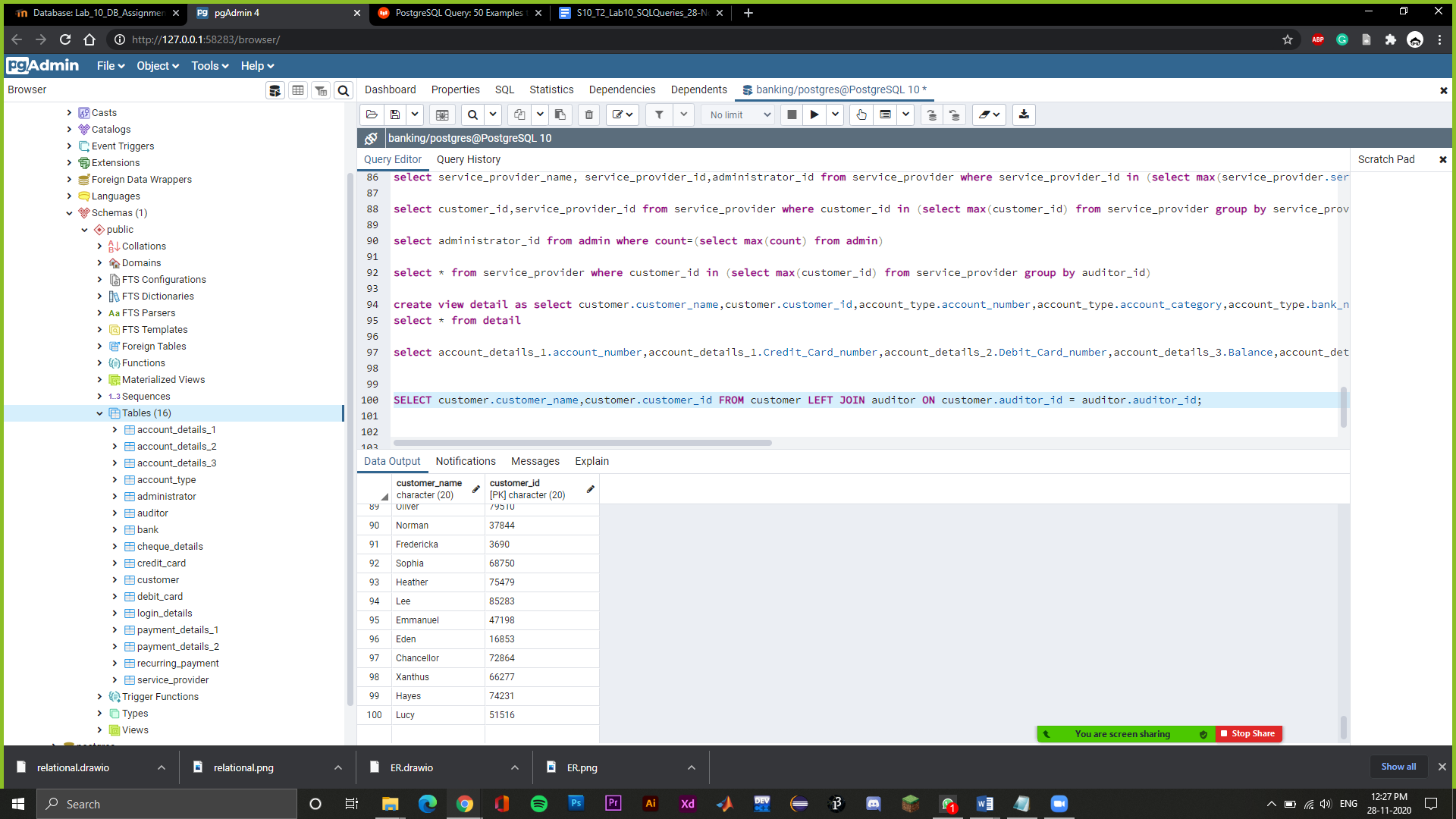
**Select account\_details\_1.account\_number, account\_details\_1.Credit\_Card\_number, account\_details\_2.Debit\_Card\_number, account\_details\_3.Balance, account\_details\_3.Recurring\_Status from account\_details\_1 join account\_details\_3 on account\_details\_1.account\_number=account\_details\_3.account\_number join account\_details\_2 on account\_details\_1.account\_number=account\_details\_2.account\_number**

->Join account\_details\_1,account\_details\_2 and account\_details\_3 based on their account number and display their details



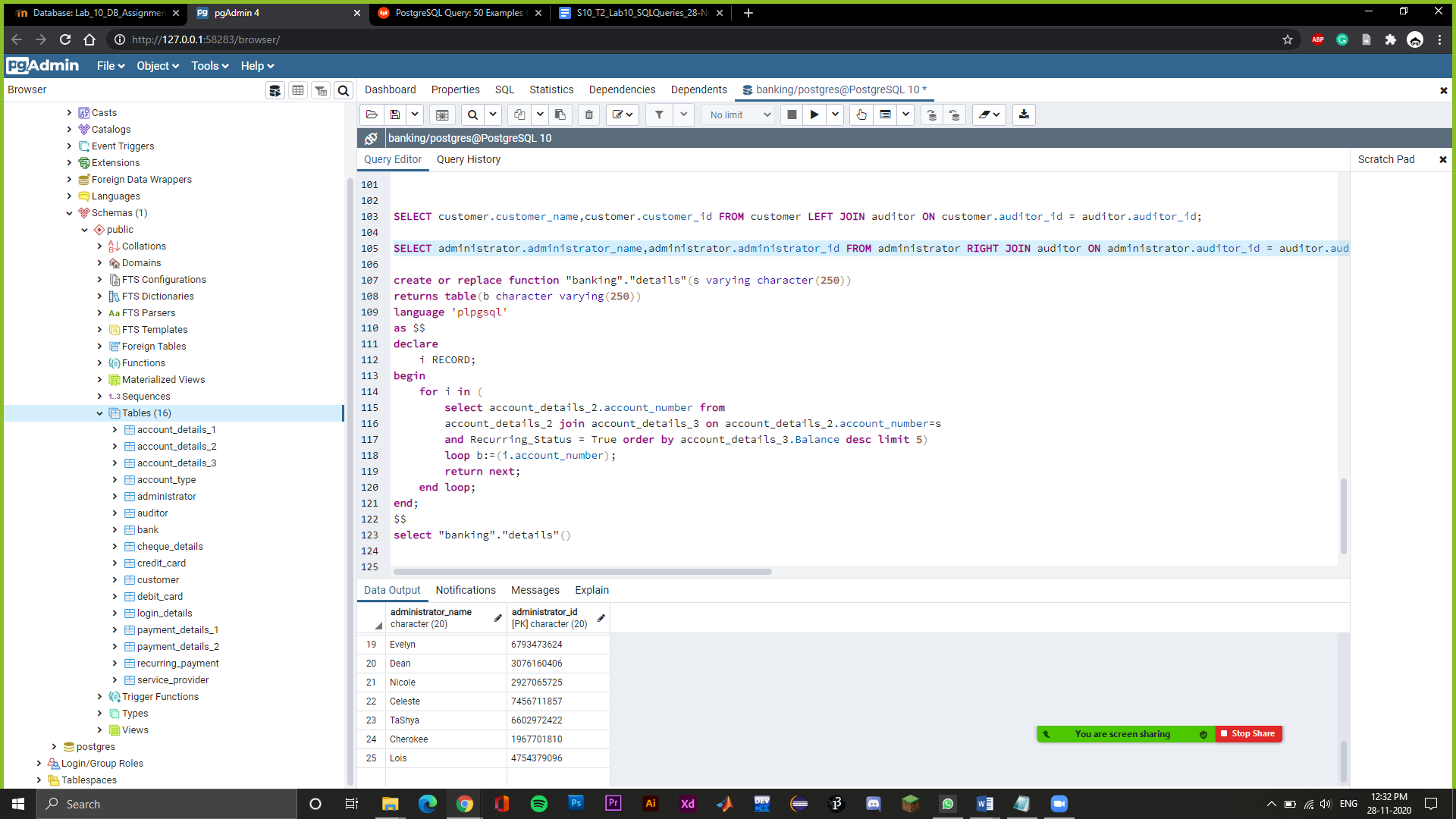
**SELECT customer.customer\_name,customer.customer\_id FROM customer LEFT JOIN auditor ON customer.auditor\_id = auditor.auditor\_id;**

->Shows left join of customer name on customer id.



**SELECT administrator.administrator\_name,administrator.administrator\_id FROM administrator RIGHT JOIN auditor ON administrator.auditor\_id = auditor.auditor\_id;**

->Shows the right join of administrator name on administrator id.



**create or replace function "details"()**

**returns table(b character varying(250))**

**language 'plpgsql'**

**as $$**

**declare**

**i RECORD;**

**begin**

**for i in (**

**select account\_details\_2.account\_number from**

**account\_details\_2 join account\_details\_3 on account\_details\_2.account\_number=account\_details\_3.account\_number**

**and Recurring\_Status = True order by account\_details\_3.Balance desc limit 5)**

**loop b:=(i.account\_number);**

**return next;**

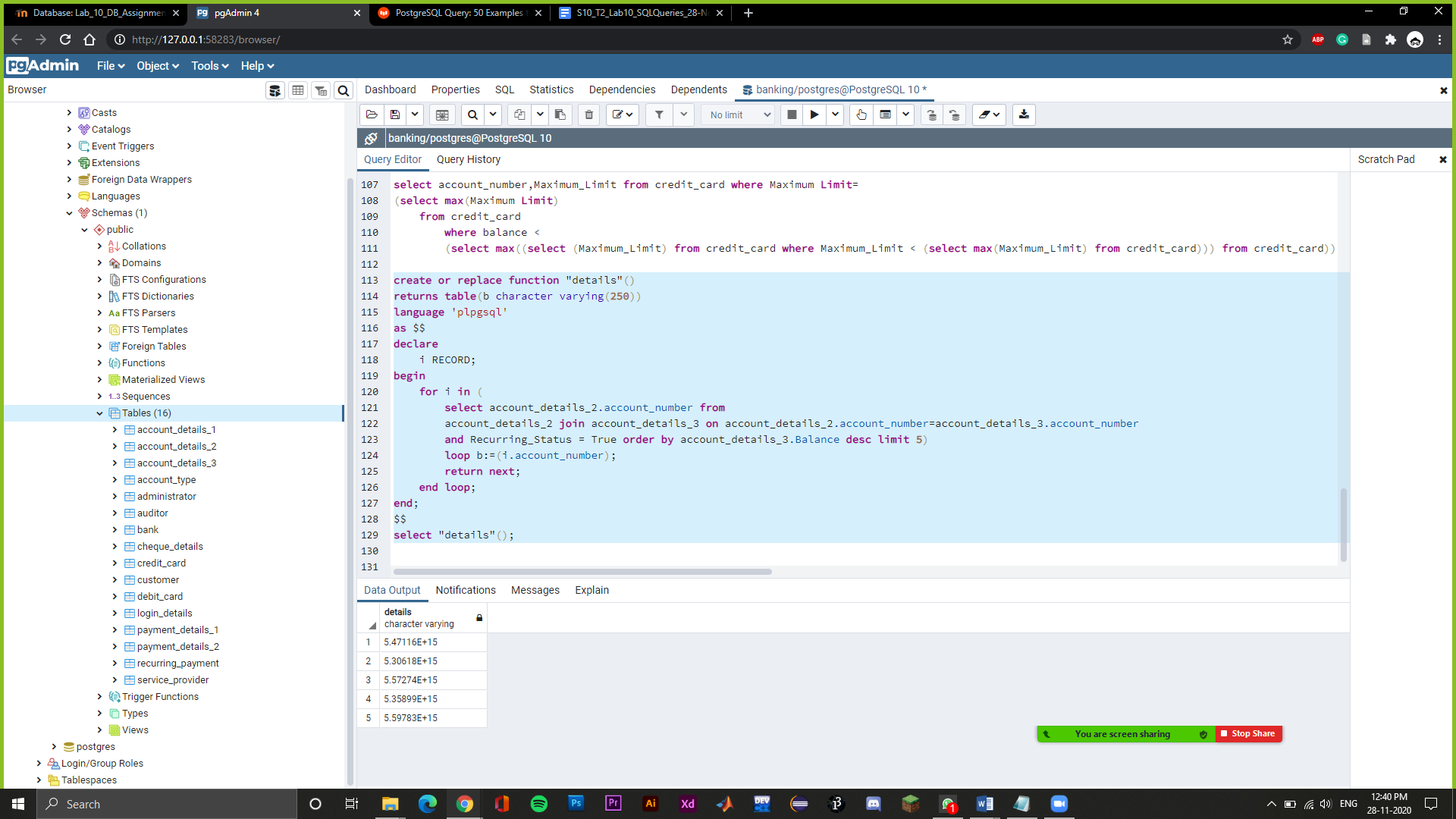
**end loop;**

**end;**

**$$**

**select "details"();**

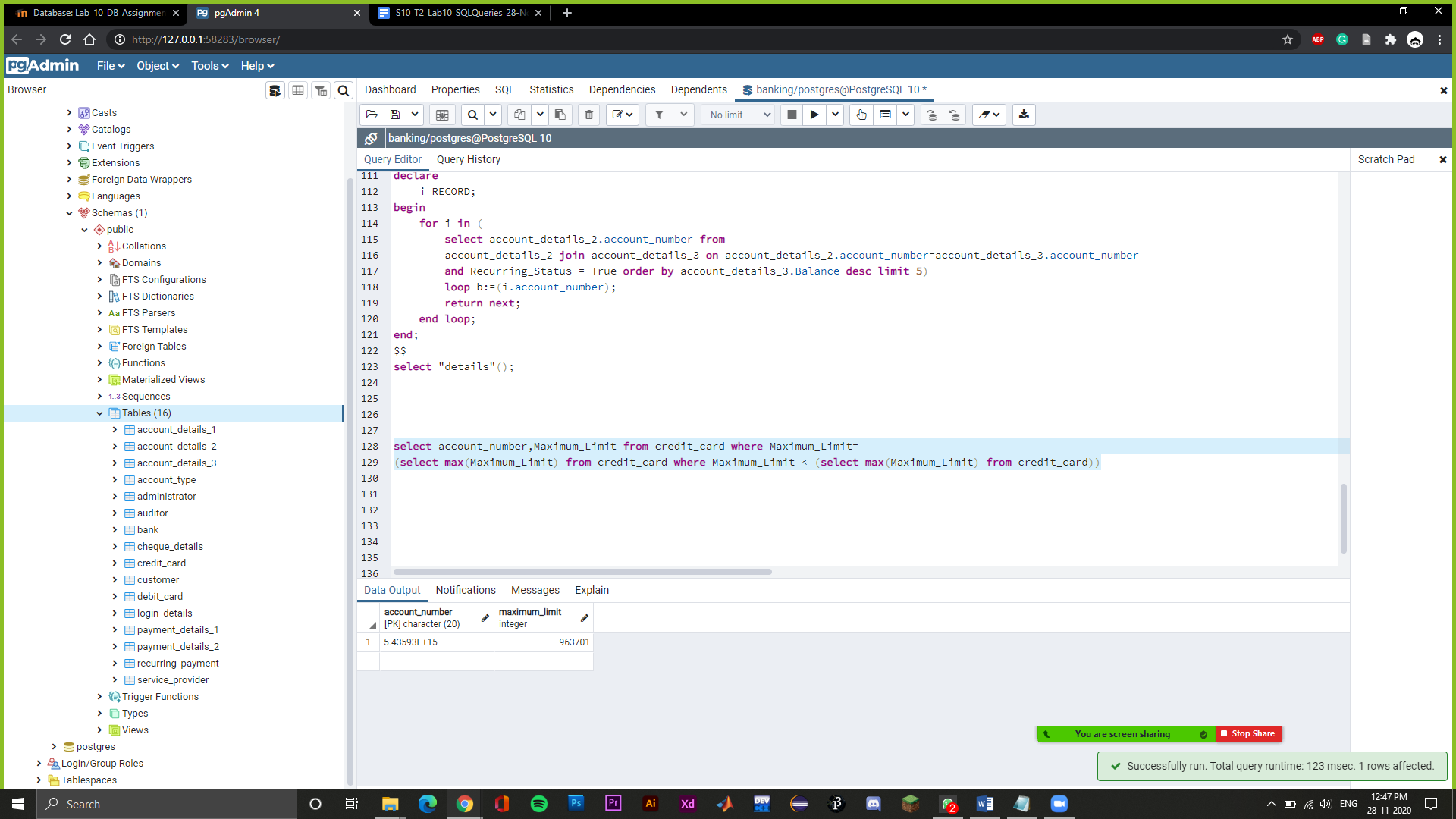
->Create a function to display the top 5 account numbers having the highest balance from the join of account\_details\_2 and account\_details\_3 where recurring status is true



**select account\_number,Maximum\_Limit from credit\_card where Maximum\_Limit=**

**(select max(Maximum\_Limit) from credit\_card where Maximum\_Limit < (select max(Maximum\_Limit) from credit\_card))**

->Select the account number with the second highest Credit Limit



c**reate or replace function "details1"()**

**returns table(b character varying(250),number character varying(250),number1 character varying(250),number3 character varying(250))**

**language 'plpgsql'**

**as $$**

**declare**

**i RECORD;**

**begin**

**for i in (select customer\_id,account\_number,administrator\_id,bank\_name from account\_type where auditor\_id='81224')**

**loop b:=(i.customer\_id);number:=(i.account\_number);number1:=(i.administrator\_id);number3:=(i.bank\_name);**

**return next;**

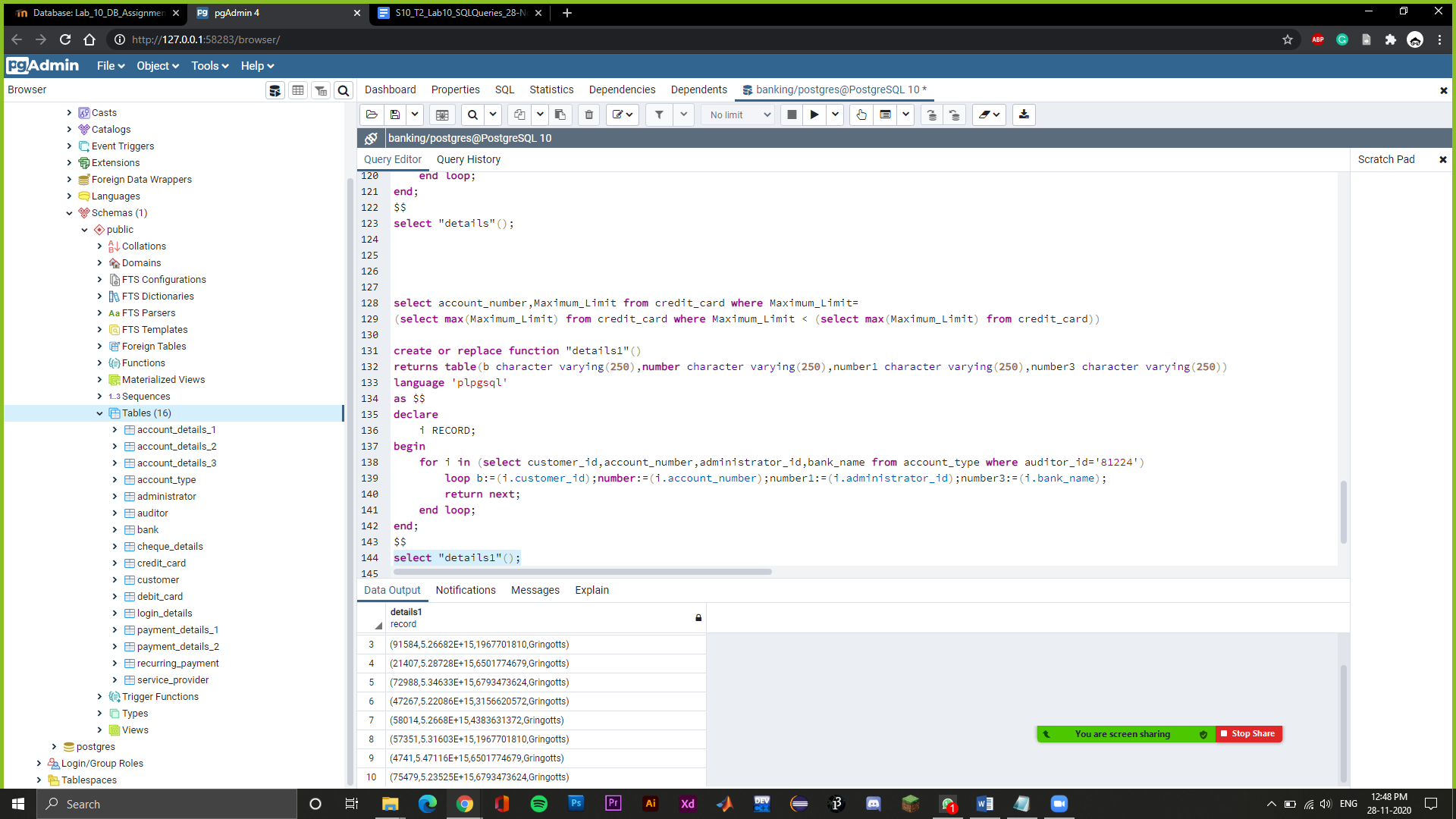
**end loop;**

**end;**

**$$**

**select "details1"();**

->Create a function to display customer ID, account number, administrator ID and bank Name from account\_type when the auditor ID is ‘81224’



**create or replace function "details4"()**

**returns table(b character varying(250),number character varying(250))**

**language 'plpgsql'**

**as $$**

**declare**

**i RECORD;**

**begin**

**for i in (SELECT customer\_id FROM account\_type EXCEPT SELECT account\_number FROM account\_details\_3 WHERE balance between '1200' and '9000'**

**)**

**loop b:=(i.customer\_id);**

**return next;**

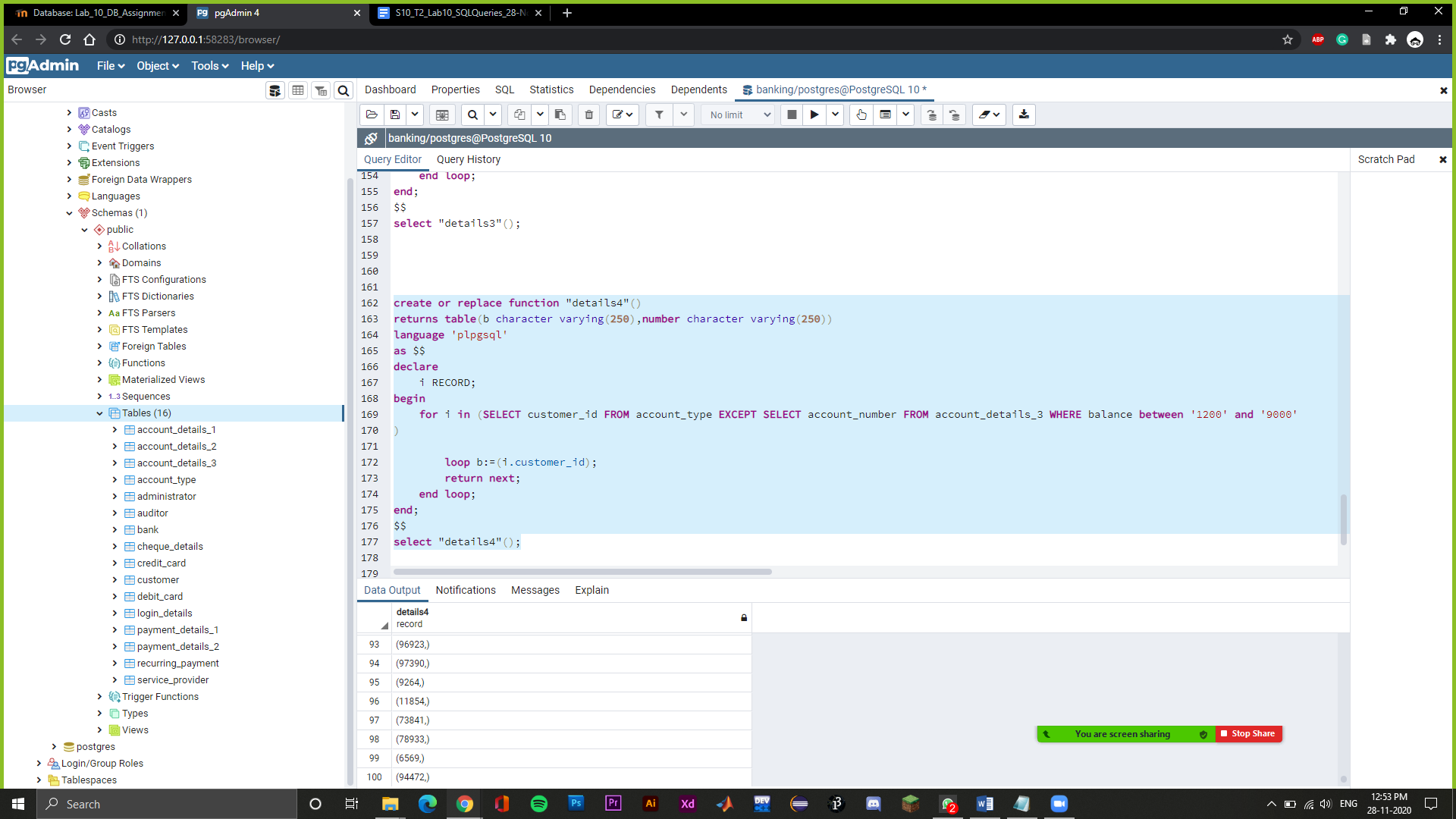
**end loop;**

**end;**

**$$**

**select "details4"();**

->Create a stored procedure of customer IDs except those who have their account balance between 1200 and 9000.



**create or replace function func\_1()**

**returns trigger**

**language 'plpgsql'**

**as $$**

**begin**

**if (new.bank\_name is NULL)**

**then UPDATE administrator SET bank\_name = 'Gringotts' WHERE administrator\_id = new.administrator\_id;**

**raise notice 'Updated to default bank!';**

**end if;**

**return new;**

**end**

**$$;**

**create trigger "def\_bank"**

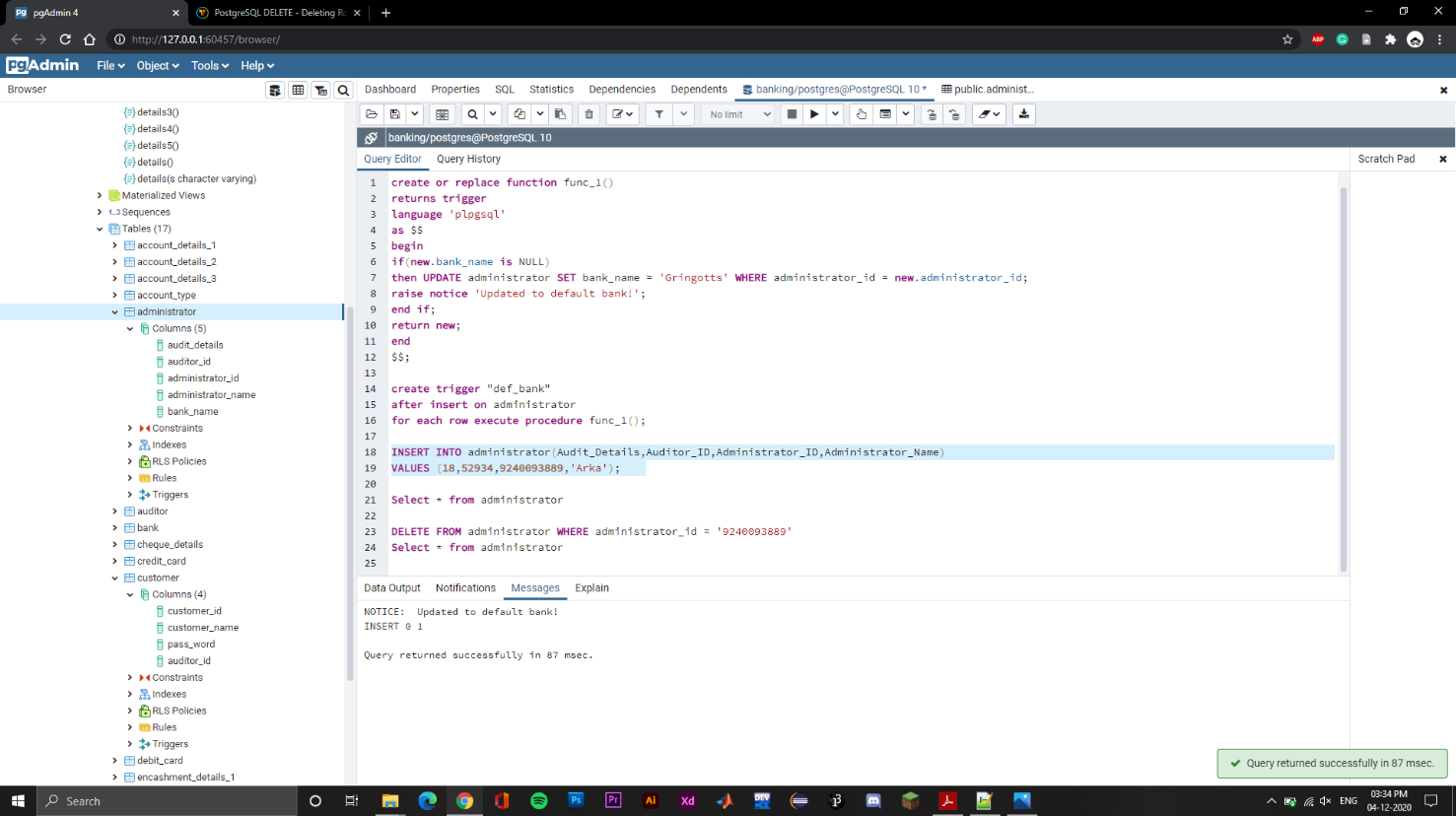
**after insert on administrator**

**for each row execute procedure func\_1();**

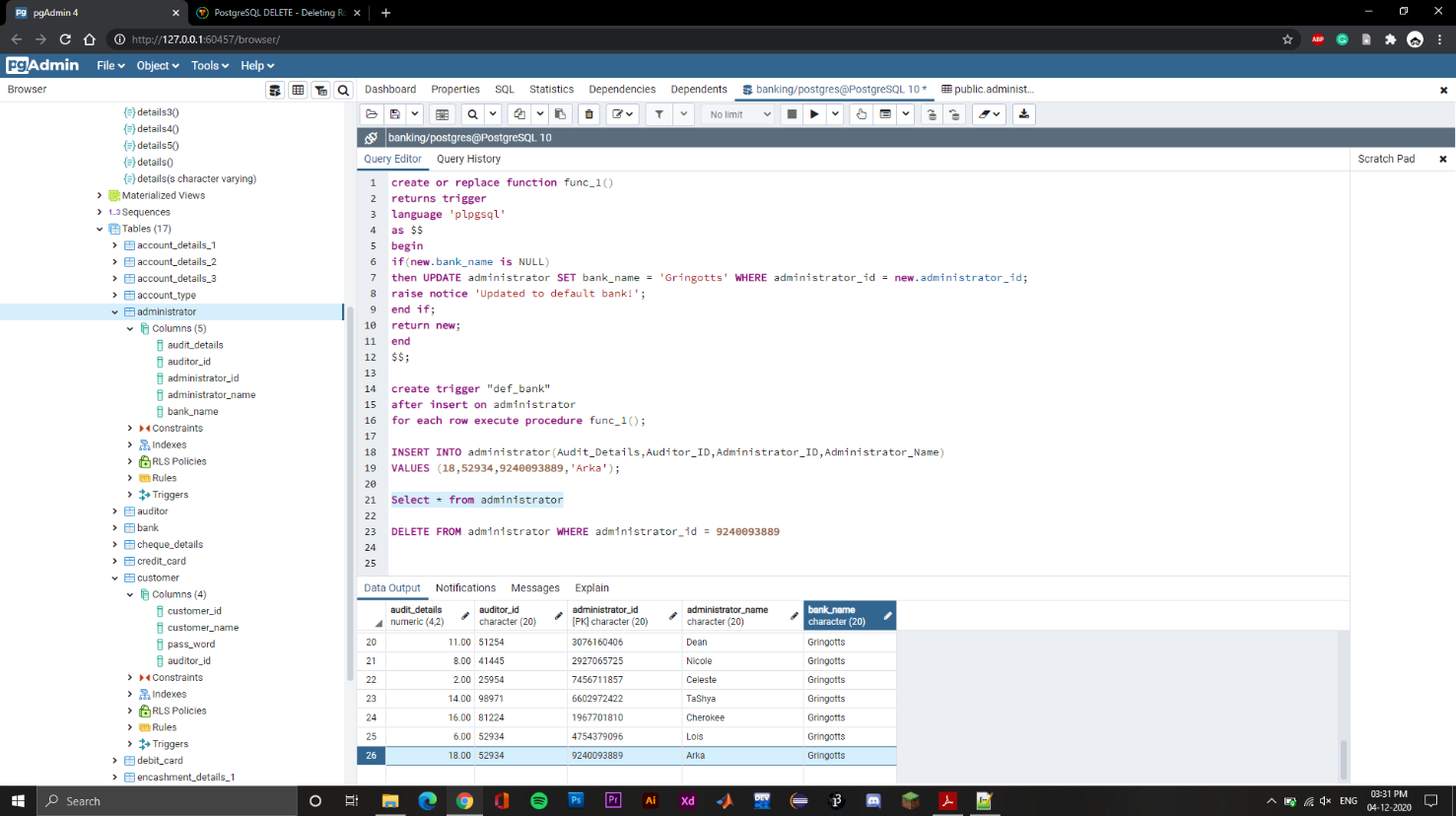
**INSERT INTO administrator(Audit\_Details,Auditor\_ID,Administrator\_ID,Administrator\_Name)**

**VALUES (18,52934,9240093889,'Arka');**

-> Insert the default name via after insert trigger if we don’t add the bank name for a new administrator



**Select \* from administrator**



**create or replace function func\_3()**

**returns trigger**

**language 'plpgsql'**

**as $$**

**begin**

**if(new.balance < 100)**

**then raise notice 'Balance must be in 3 digits';**

**return old;**

**end if;**

**return new;**

**end**

**$$;**

**create trigger "bal"**

**before insert on account\_details\_3**

**for each row execute procedure func\_3();**

**INSERT INTO account\_details\_3(Account\_Number,Balance,Recurring\_Status)**

**VALUES ('5.43999E+15', 74, false);**

->If balance before insert is in 2 digits then raise minimum balance notice