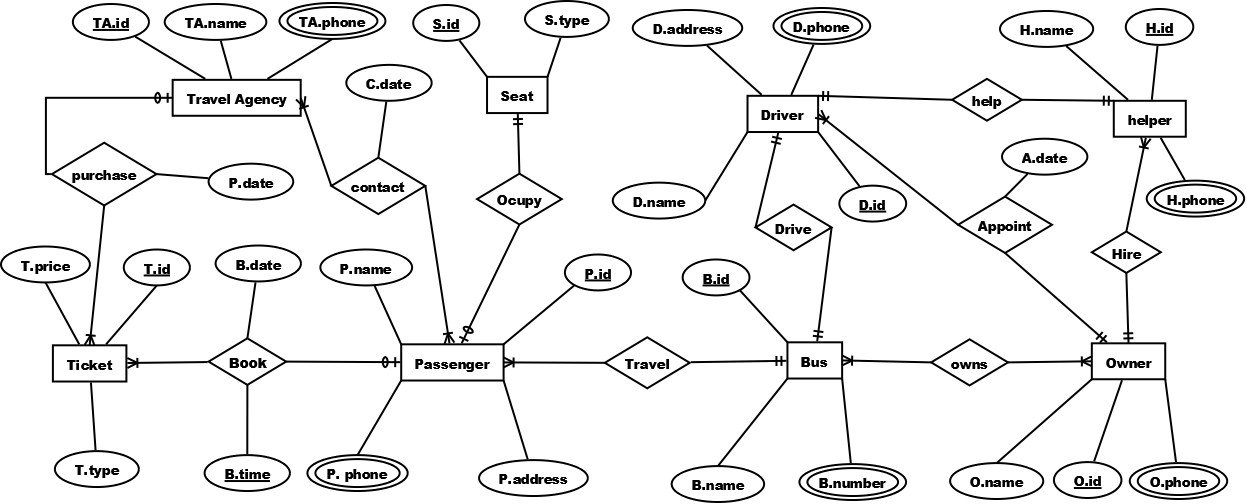
**INTRODUCTION**

This project Bus Ticket Counter Management System has been developed on SQL. The main aim to develop this project is to provide the details of bus and bookings by passengers. This system manages bus details, passenger details and route details. It is a simple project which is feasible and helps the work to be more efficient and accurate with complete security.

**CASE STUDY**

In a bus ticket counter management system a passenger may buy many tickets. One ticket may be booked by exactly one passenger. A passenger is defined by a passenger id. The system also stores passenger’s name, phone number, email, and address. A ticket is defined by ticket id, ticket type, ticket price. Booking date, booking id and the passenger who booked the ticket, his passenger id is also stored. A passenger can contact a travel agency to purchase his/her ticket. The system stores TA.id, TA.name, TA.phone of the travel agency and P.date, C.date of purchase and contact. There are many buses under the ticket counter management system. A bus has bus name, bus id, bus number, bus type. The bus is owned by an owner and he is defined by O.name, O.id and O.phone. An owner can appoint many drivers but only one driver can appoint one owner. The a.date is stored in the system The owner hires a helper who is defined by H.name, H.id and H.phone. A bus may have only one driver and only one helper may help the driver. A driver may drive only one bus. A bus is also driven by only one driver. Driver’s name, id, phone number, address is also stored. A bus has many seats. Seats have specific seat id and seat type. A passenger may occupy only one seat and a seat can also be occupied by only one passenger. A passenger may travel in one bus, but a bus can travel with many passengers.

****

**NORMALIZATION**

Purchase (T.id, T.price, T.type, TA.id, TA.name, TA.phone)

1NF: TA.phone multivalued attribute.

2NF: T.id, T.price, T.type,TA.id

TA.id, TA.name, TA.phone

3NF: No transitive dependency.

T.id, T.price, T.type,TA.id

TA.id, TA.name, TA.phone

Table: 1. T.id, T.price, T.type,TA.id

2. TA.id, TA.name, TA.phone

Occupy (P.id, P.name, P.phone, P.address, S.id, S.type)

1NF: P.phone is a multivalued attribute.

2NF: P.id, P.name, P.phone, P.address

S.id, S.type

3NF: No transitive dependency.

P.id, P.name, P.phone, P.address

S.id, S.type

Table: 1. P.id, P.name, P.phone, P.address

2. S.id, S.type

Travel (P.id, P.name, P.phone, P.address, B.id, B.name, B.number)

1NF: P.phone, B.number are multivalued attributes.

2NF: P.phone, P.name, P.phone, P.address,B.id

B.id, B.name, B.number.

3NF: No transitive dependency.

P.phone, P.name, P.phone, P.address,B.id

B.id, B.name, B.number.

Table: 1. P.phone, P.name, P.phone, P.address, B.id

2. B.id, B.name, B.number.

Owns (B.id, B.name, B.number, O.id, O.phone, O.name)

1NF: B.number, O.phone are multivalued attributes.

2NF: B.id, B.name, B.number

O.id, O.phone, O.name

BO.id, B.id, O.id

3NF: No transitive dependency.

B.id, B.name, B.number

O.id, O.phone, O.name

BO.id, B.id, O.id

Table: 1. B.id, B.name, B.number

2. O.id, O.phone, O.name

3. BO.id, B.id, O.id

Help (H.id, H.name, H.number, D.id, D.phone, D,address, D.name)

1NF: H.phone, D.phone are multivalued attributes.

2NF: H.id, H.name, H.number, D.id

D.id, D.phone, D.address, D.name

3NF: No transitive dependency.

H.id, H.name, H.number, D.id

D.id, D.phone, D.address, D.name

Table: 1. H.id, H.name, H.number, D.id

2. D.id, D.phone, D.address, D.name

Contact (TA.id, TA.name, TA.phone, P.id, P.name, P.address, P.phone, C.date)

1NF: TA.phone, P.phone are multivalued attributes.

2NF: TA.id, TA.name, TA.phone

P.id, P.name, P.address, P.phone

TA\_P.id, TA.id, P.id

C.TA\_P.id, C.date, TA.id, P.id, TA\_P.id

3NF: No transitive dependency.

TA.id, TA.name, TA.phone

P.id, P.name, P.address, P.phone

TA\_P.id, TA.id, P.id

C.TA\_P.id, C.date, TA.id, P.id, TA\_P.id

Table: 1. TA.id, TA.name, TA.phone

2. P.id, P.name, P.address, P.phone

3. TA\_P.id, TA.id, P.id

4.C.TA\_P.id, C.date, TA.id, P.id, TA\_P.id

Book ( P.id, P.name, P.phone, P.address, T.id, T.price, T.type, B.date, B.time)

1NF: P.phone is a multivalued attribute.

2NF: P.id, P.name, P.phone, P.address

T.id, T.price, T.type, P.id

PT.id, B.date, B.time, P.id, T.id

3NF: No transitive dependency.

P.id, P.name, P.phone, P.address

T.id, T.price, T.type, P.id

PT.id, B.date, B.time, P.id, T.id

Table: 1.P.id, P.name, P.phone, P.address

2.T.id, T.price, T.type, P.id

3.PT.id, B.date, B.time, P.id, T.id

Drive (D.id, D.name, D.phone, D.address, B.id, B.name, B.number)

1NF: D.phone, B.number are multivalued attributes.

2NF: D.id, D.name, D.phone, D.address

B.id, B.name, B.number, D.id

3NF: No transitive dependency.

D.id, D.name, D.phone, D.address

B.id, B.name, B.number, D.id

Table: 1. D.id, D.name, D.phone, D.address

2. B.id, B.name, B.number, D.id

Appoint ( D.id, D.name, D.phone, D.address, O.id, O.name, O.phone, A.date)

1NF: D.phone, O.phone are multivalued attributes.

2NF: D.id, D.name, D.phone, D.address

O.id, O.name, O.phone

DO.id, A.date, D.id, O.id

3NF: No transitive dependency

D.id, D.name, D.phone, D.address

O.id, O.name, O.phone

DO.id, A.date, D.id, O.id

Table: 1. D.id, D.name, D.phone, D.address

2. O.id, O.name, O.phone

3. DO.id, A.date, D.id, O.id

Hire (H.id, H.name, H.phone, O.id, O.phone, O.name)

1NF: H.phone, O.phone are multivalued attributes.

2NF: H.id, H.name, H.phone, O.id

O.id, O.phone, O.name

3NF: No transitive dependency.

H.id, H.name, H.phone, O.id

O.id, O.phone, O.name

Table: 1. H.id, H.name, H.phone, O.id

2. O.id, O.phone, O.name

**FINAL TABLE**

1. T.id, T.price, T.type,TA.id

2. TA.id, TA.name, TA.phone

3. S.id, S.type

4. P.phone, P.name, P.phone, P.address, B.id

5. O.id, O.phone, O.name

6. BO.id, B.id, O.id

7. H.id, H.name, H.number, D.id, O.id

8. C.TA\_P.id, C.date, TA.id, P.id, TA\_P.id

9. T.id, T.price, T.type, P.id

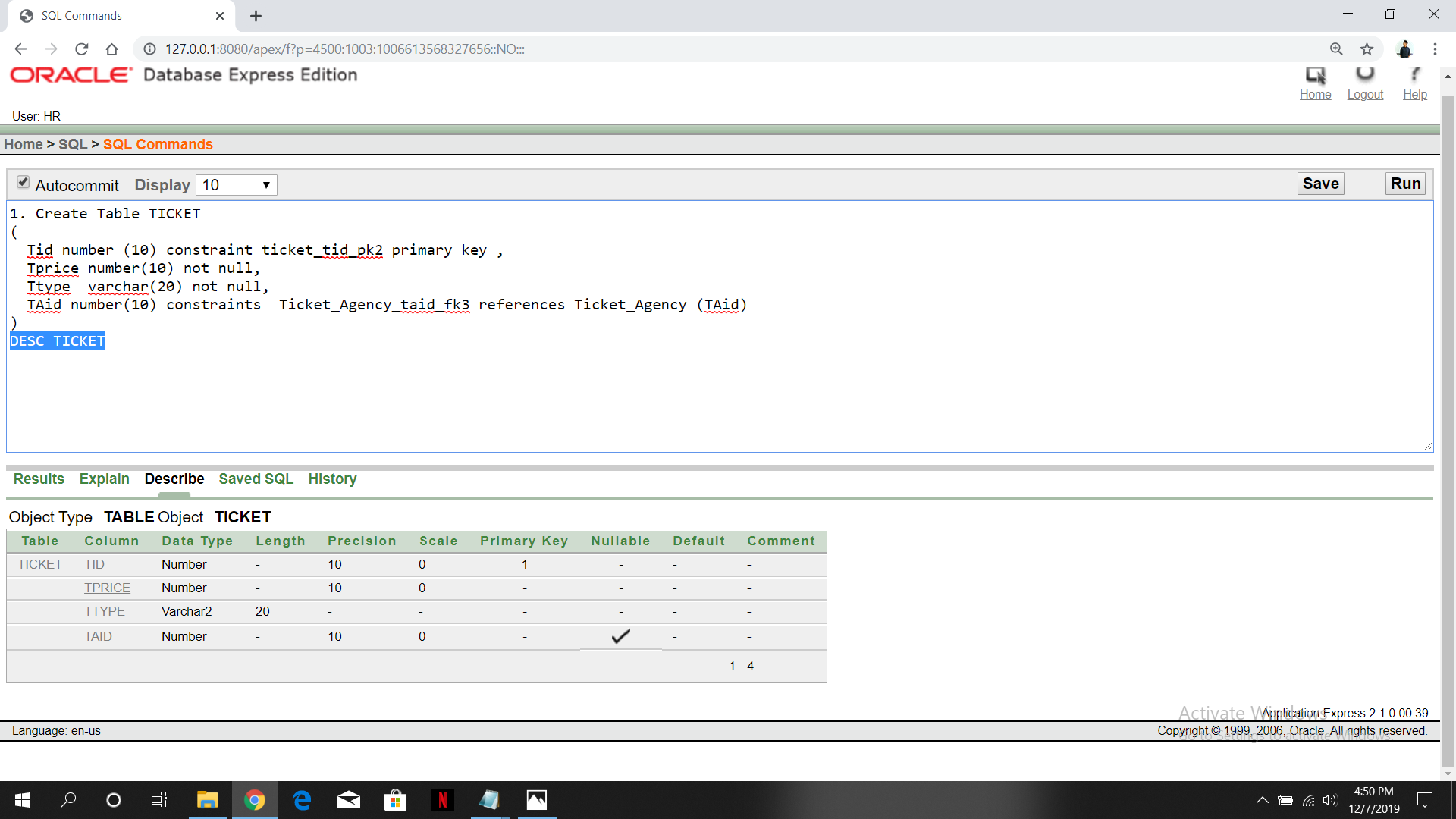
10. PT.id, B.date, B.time, P.id, T.id

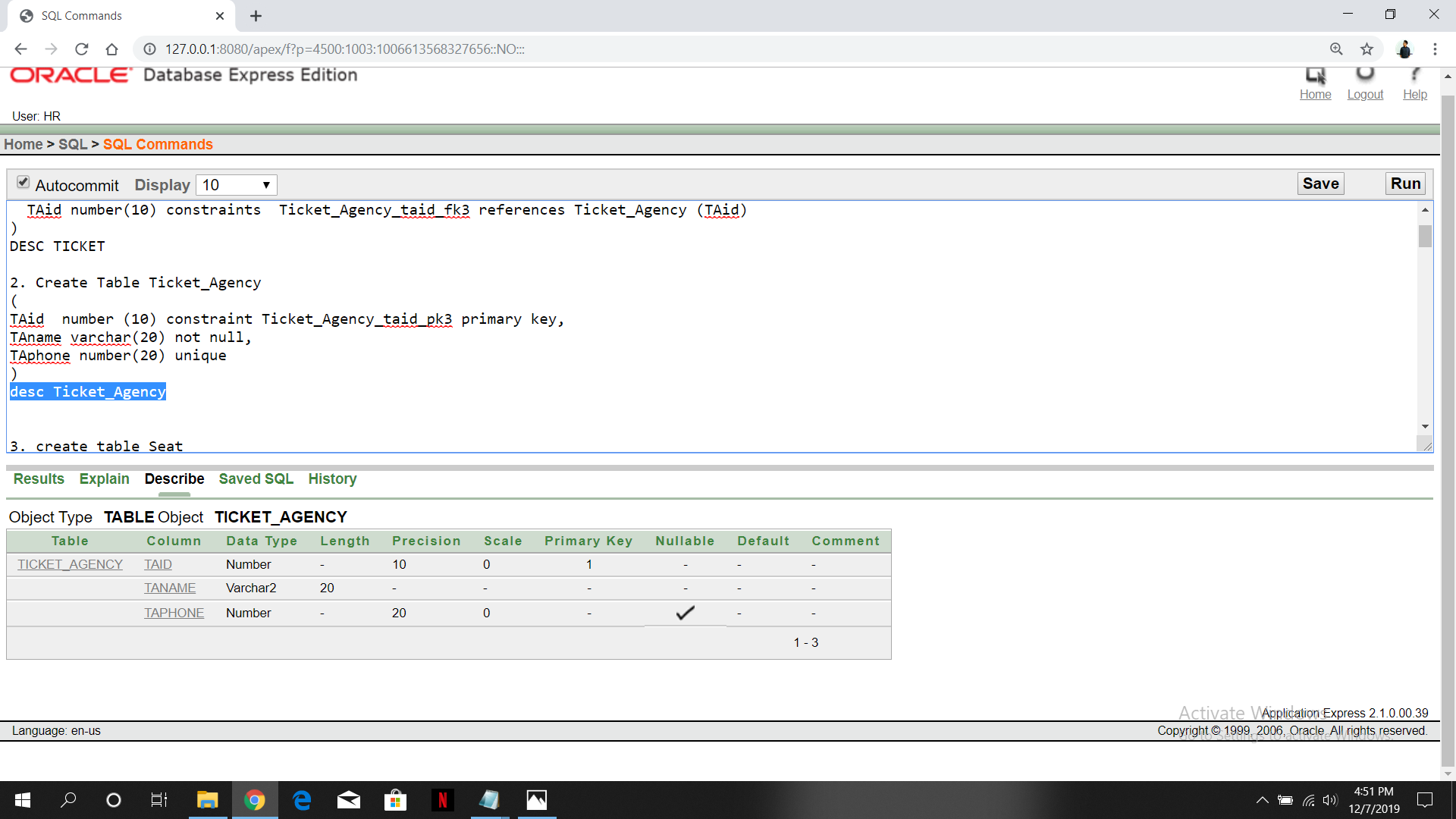
11. B.id, B.name, B.number, D.id

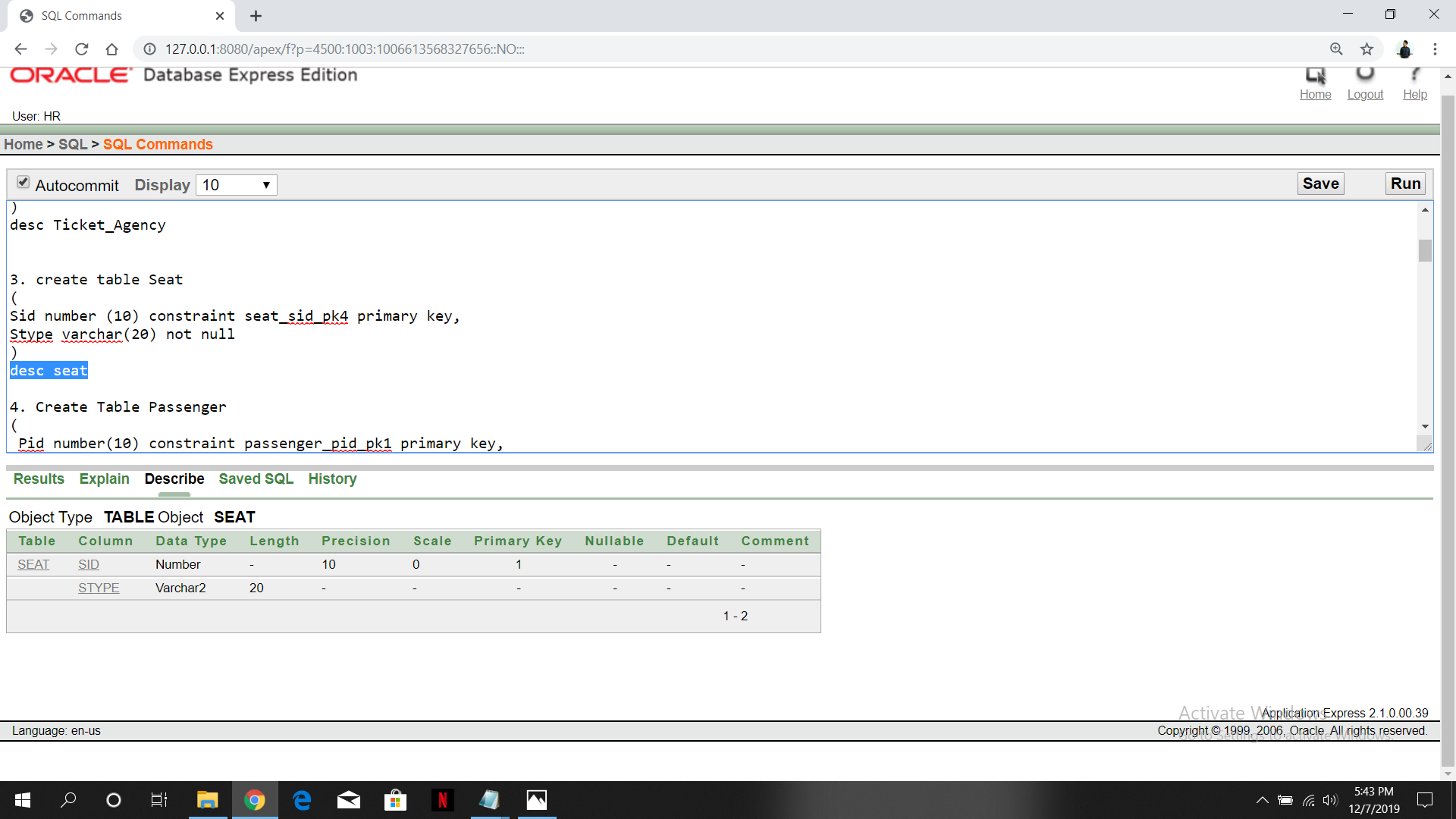
12. D.id, D.name, D.phone, D.address, O.id

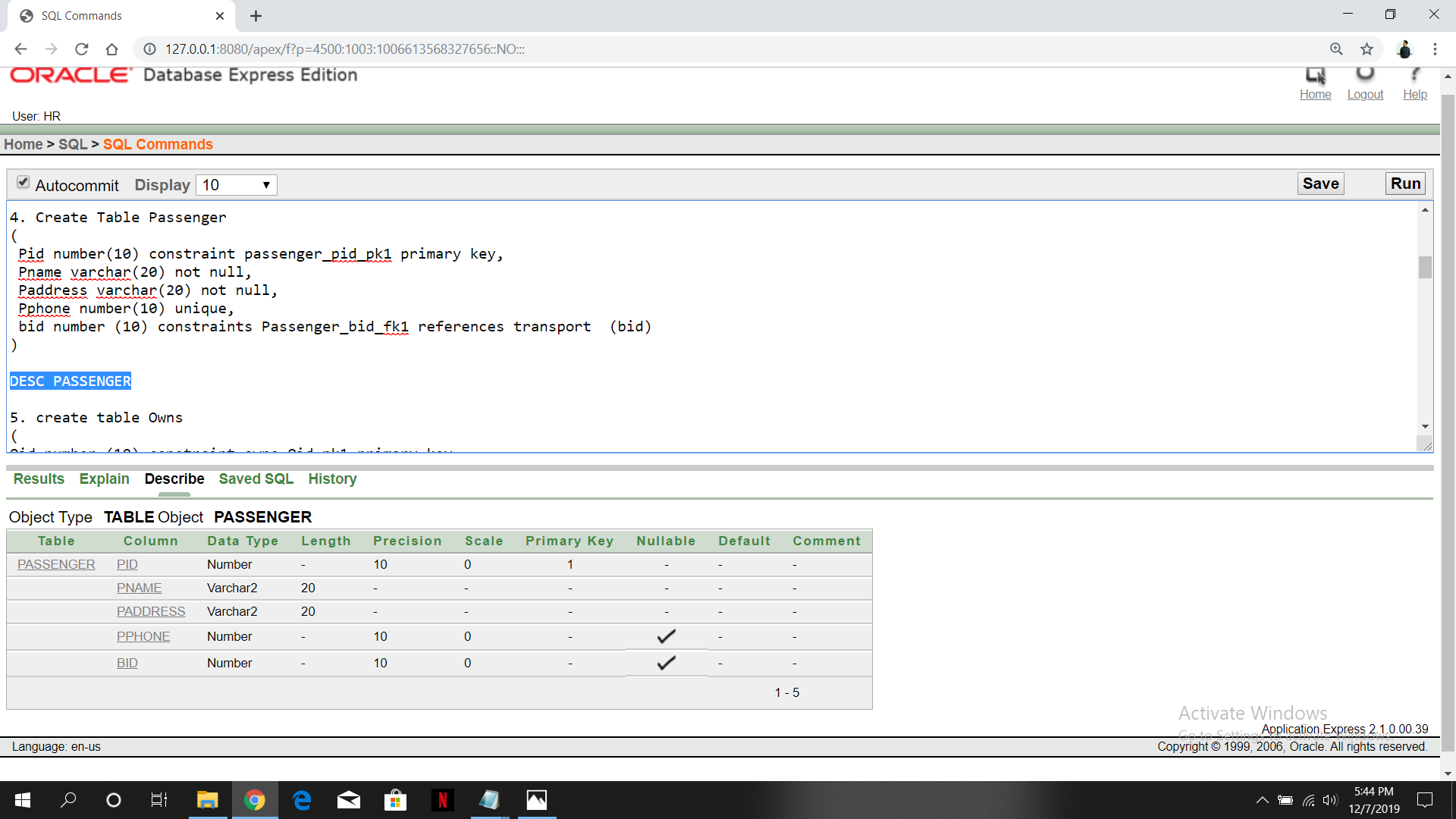
13. DO.id, A.date, D.id, O.id

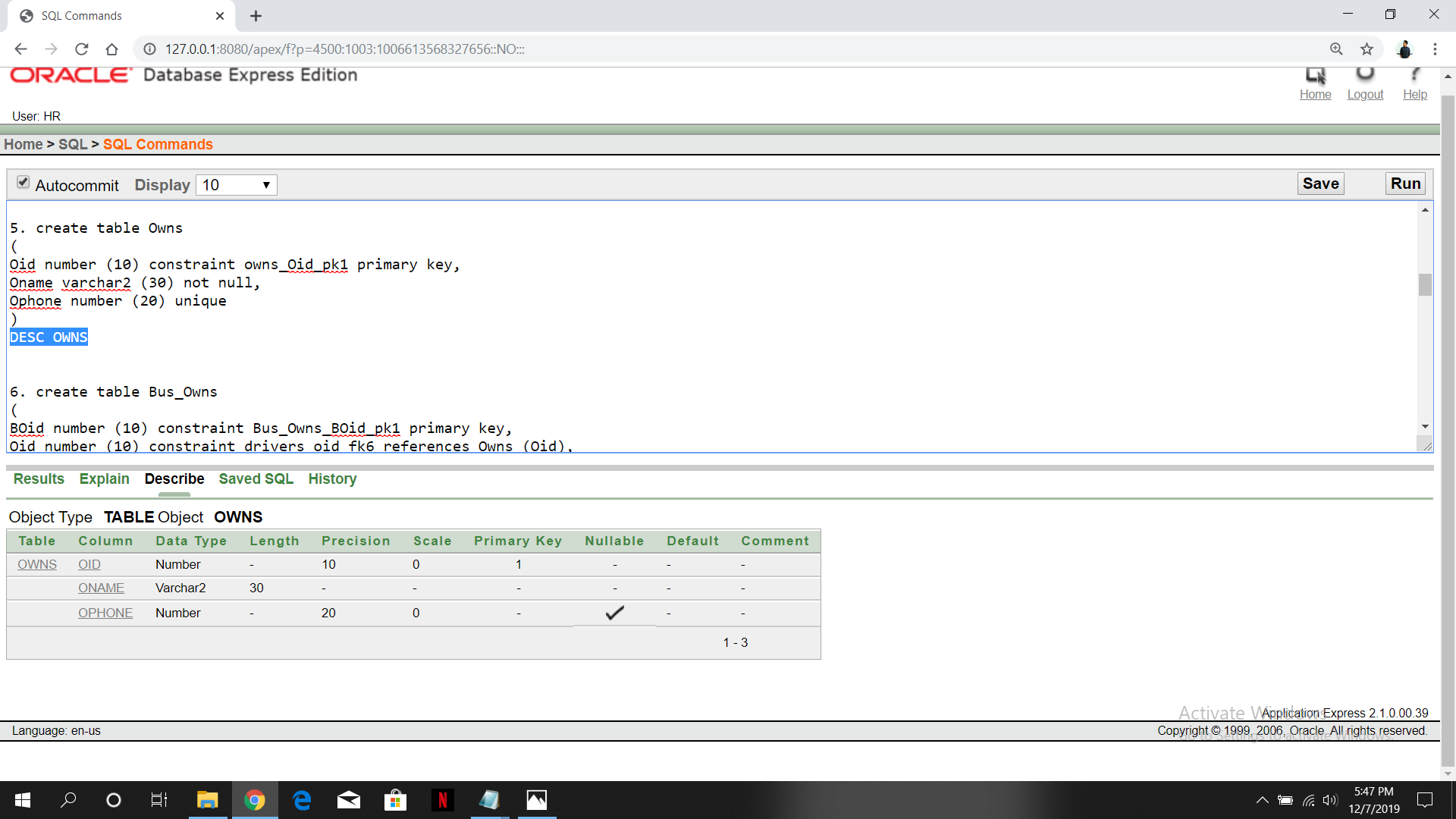
**TABLE CREATION**

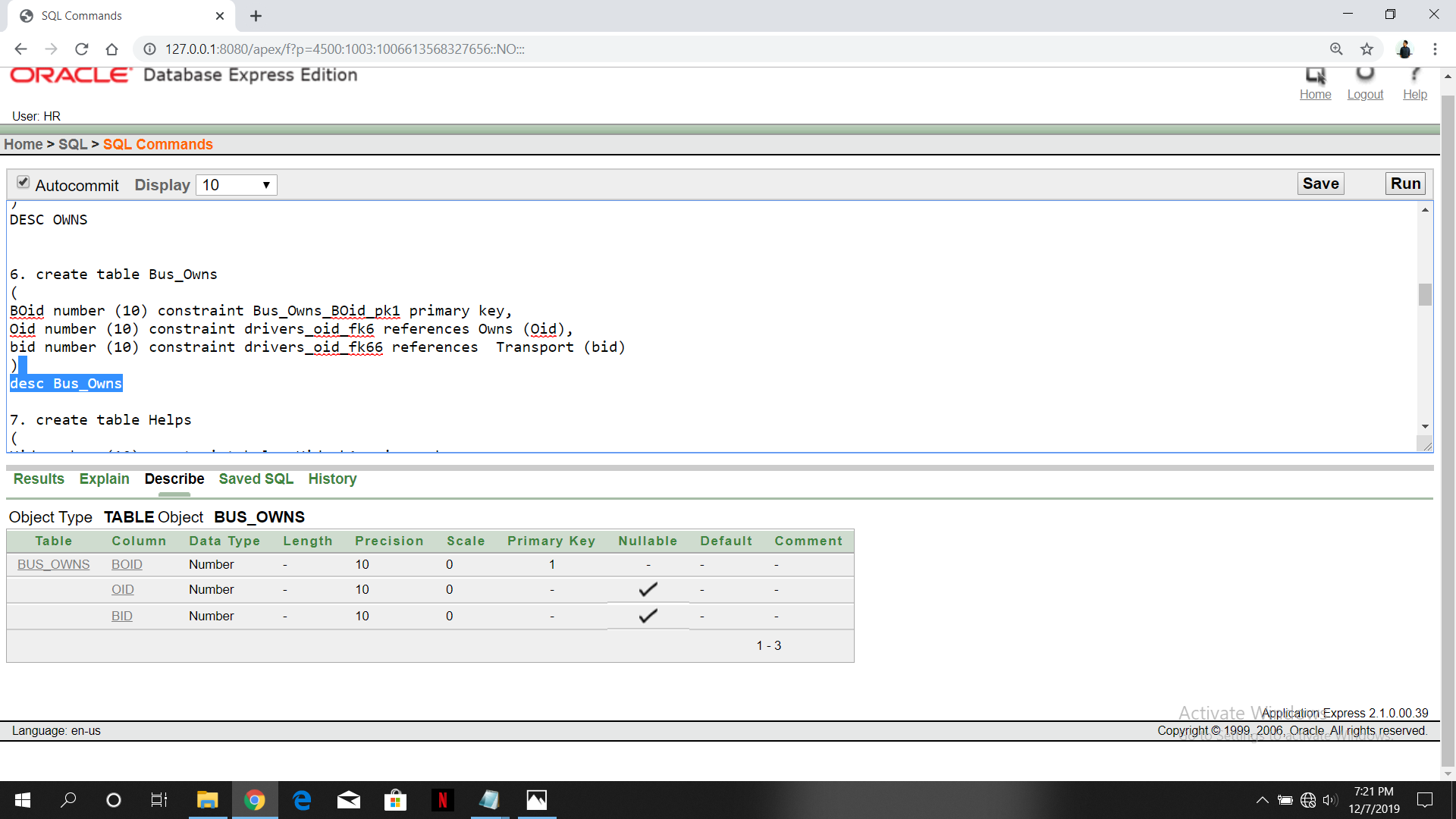


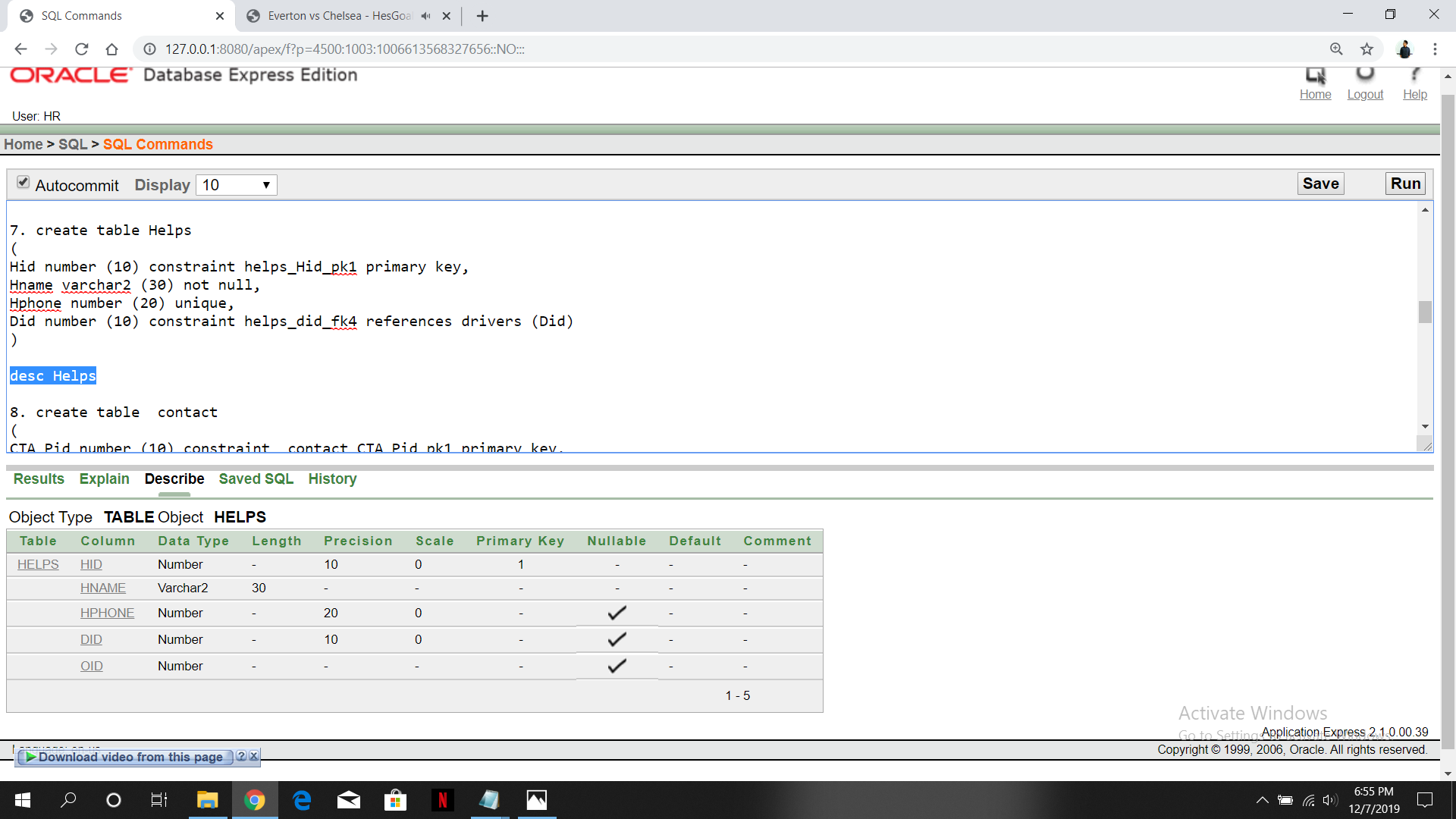


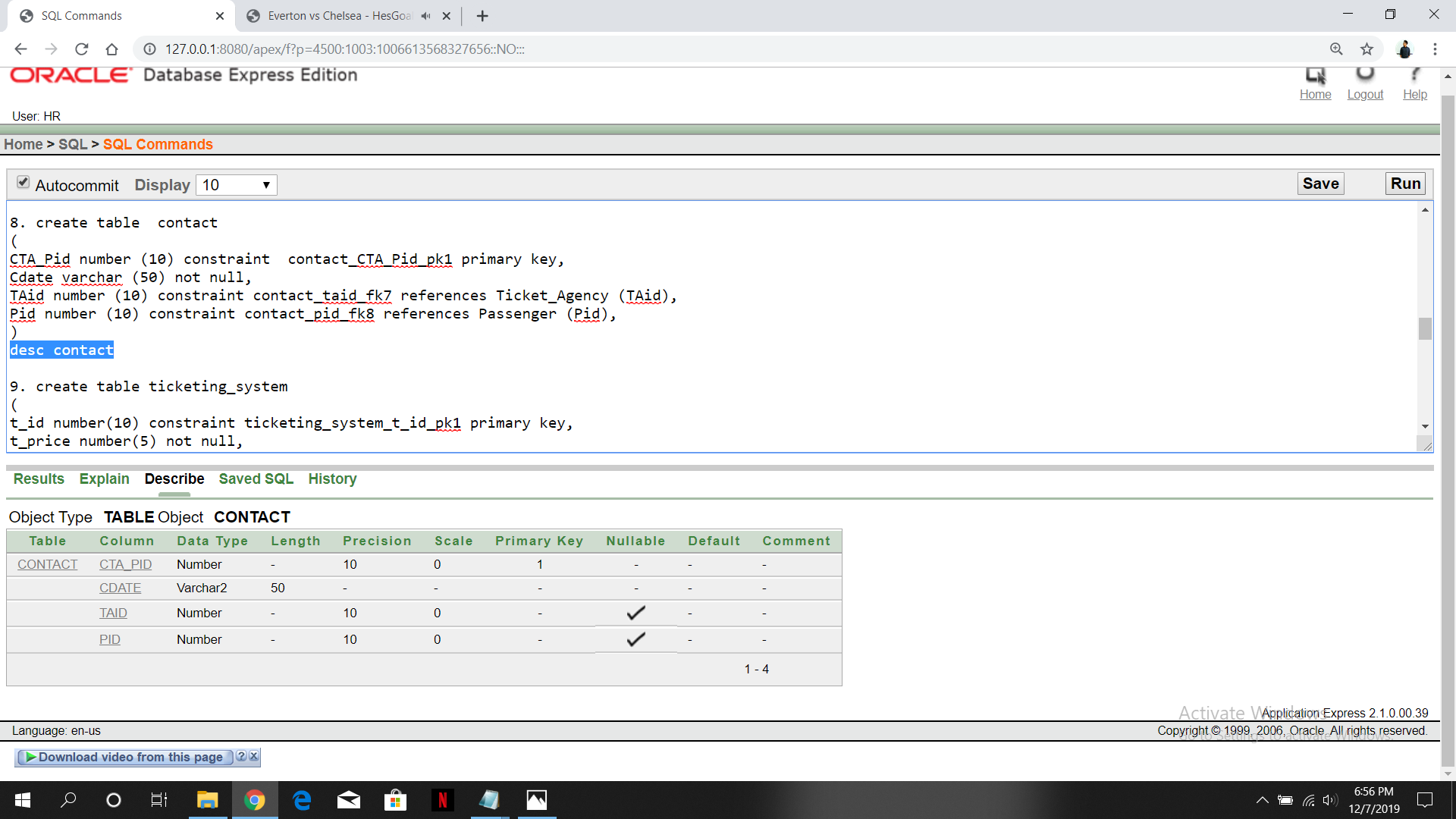


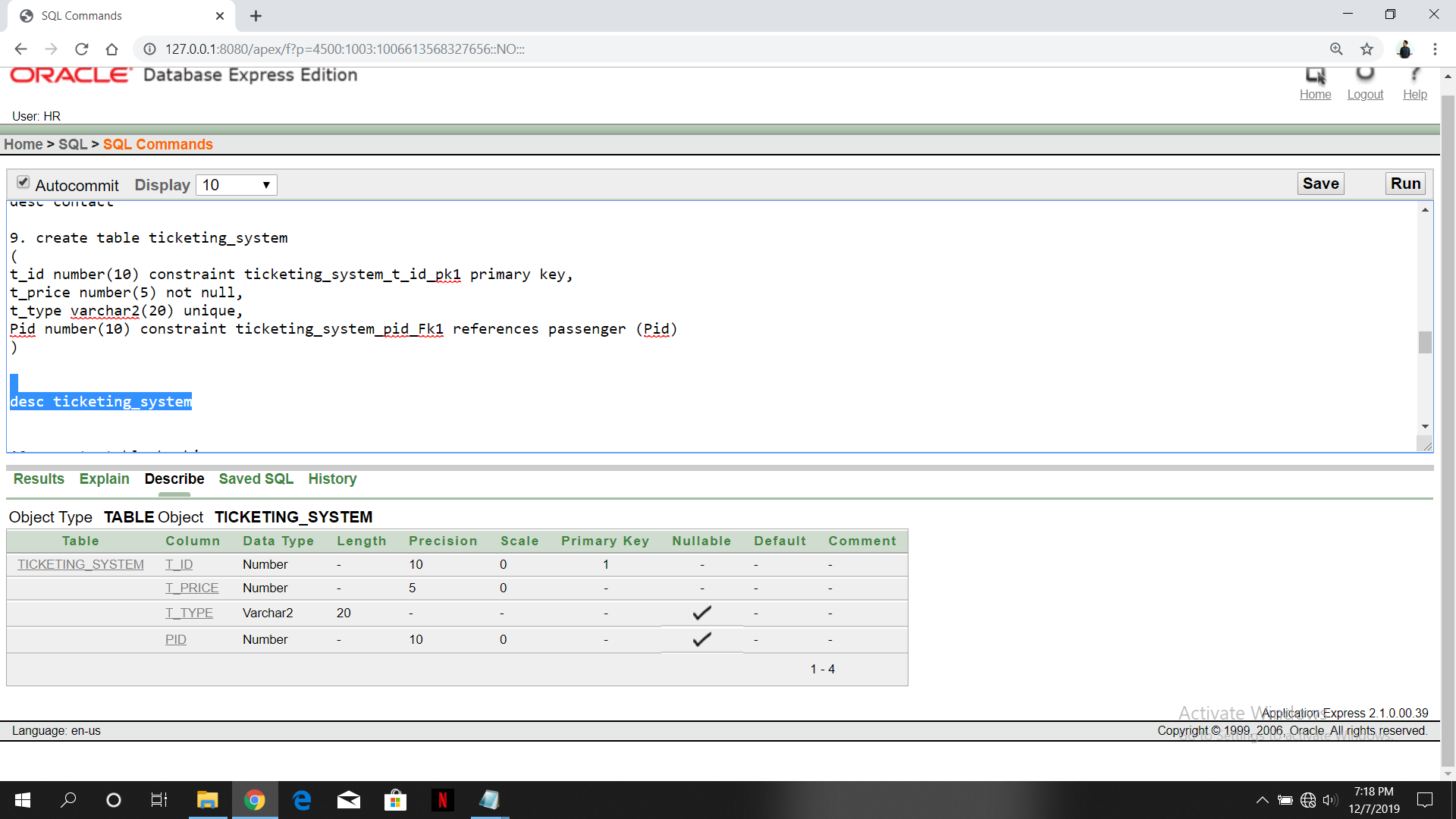


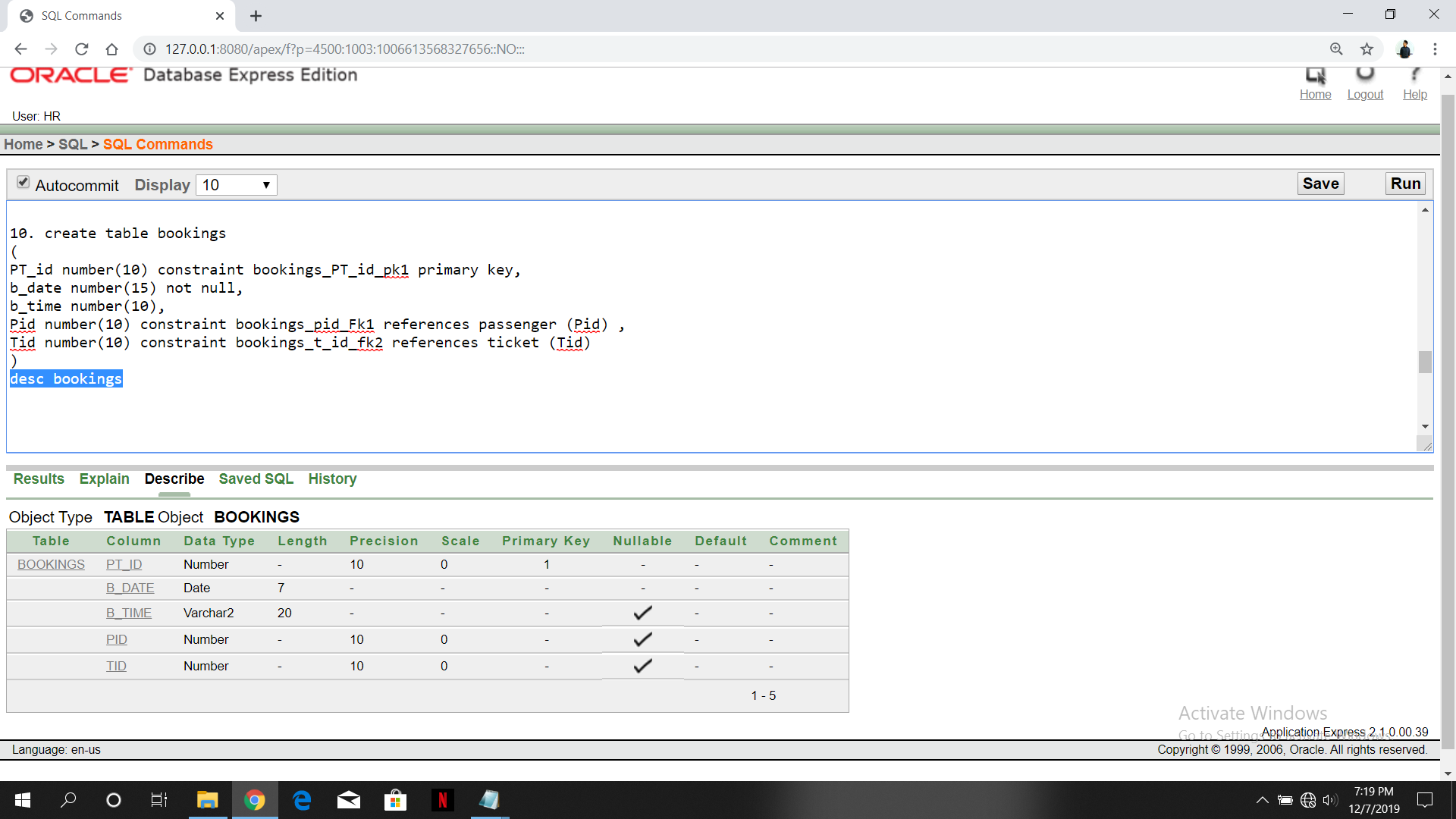


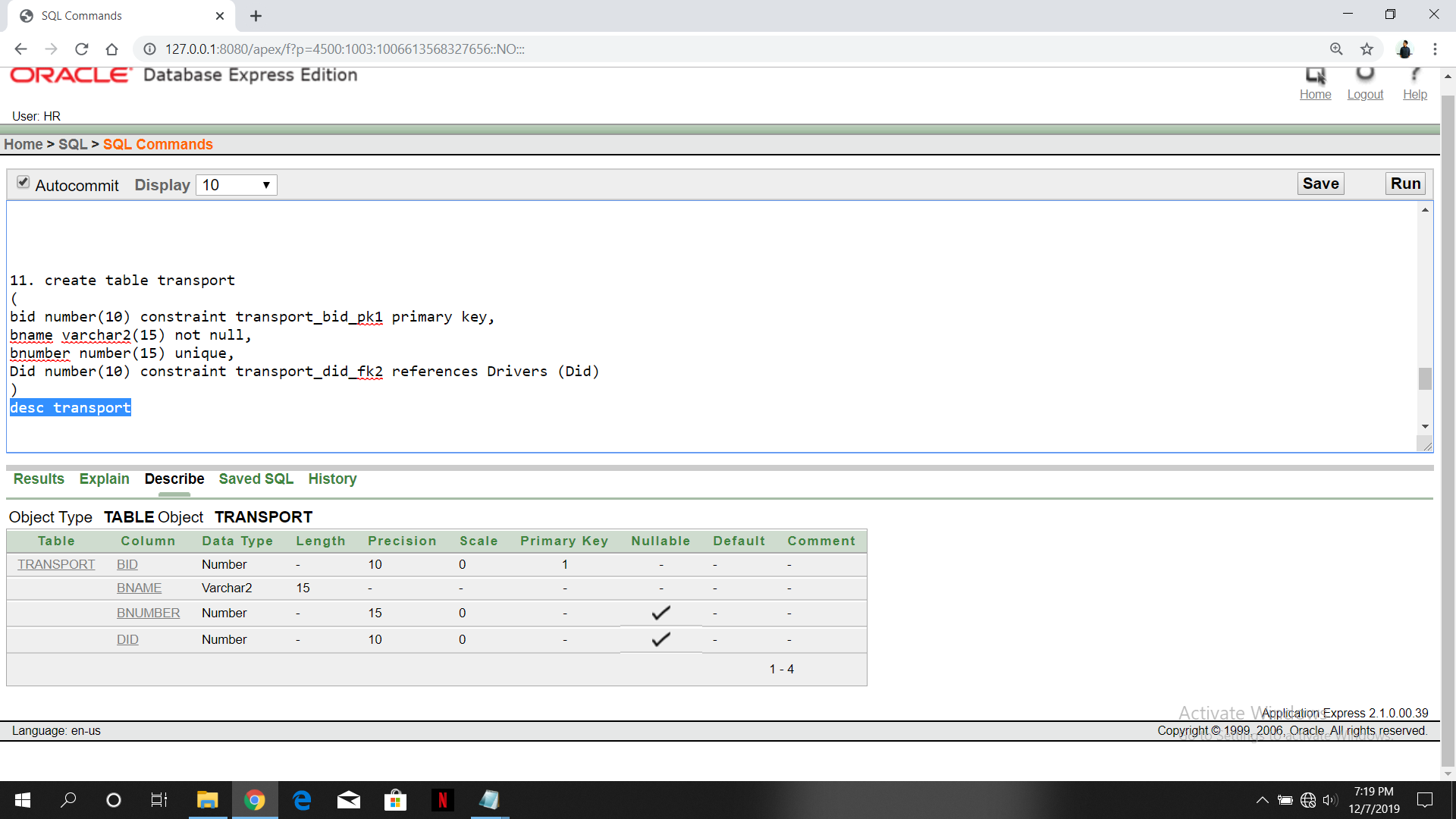


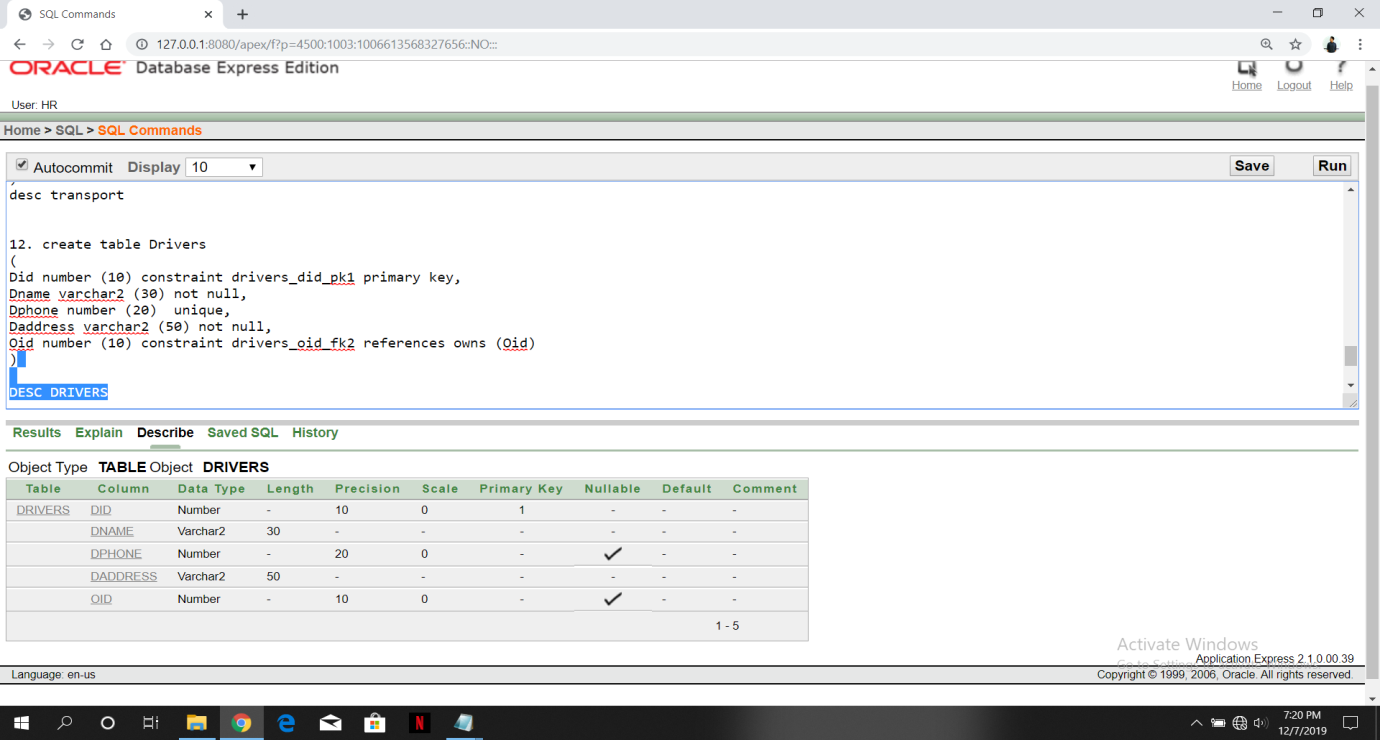


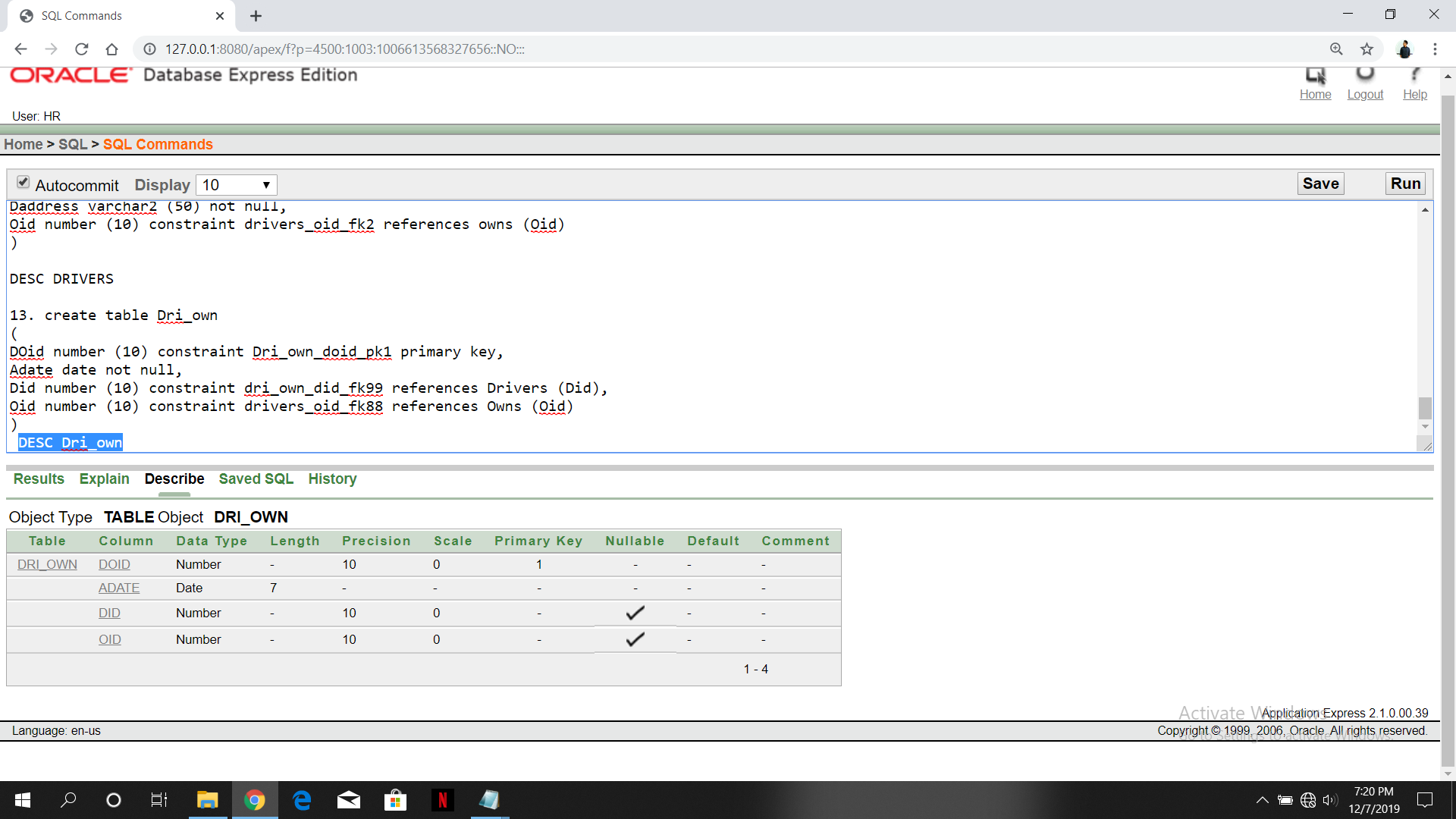




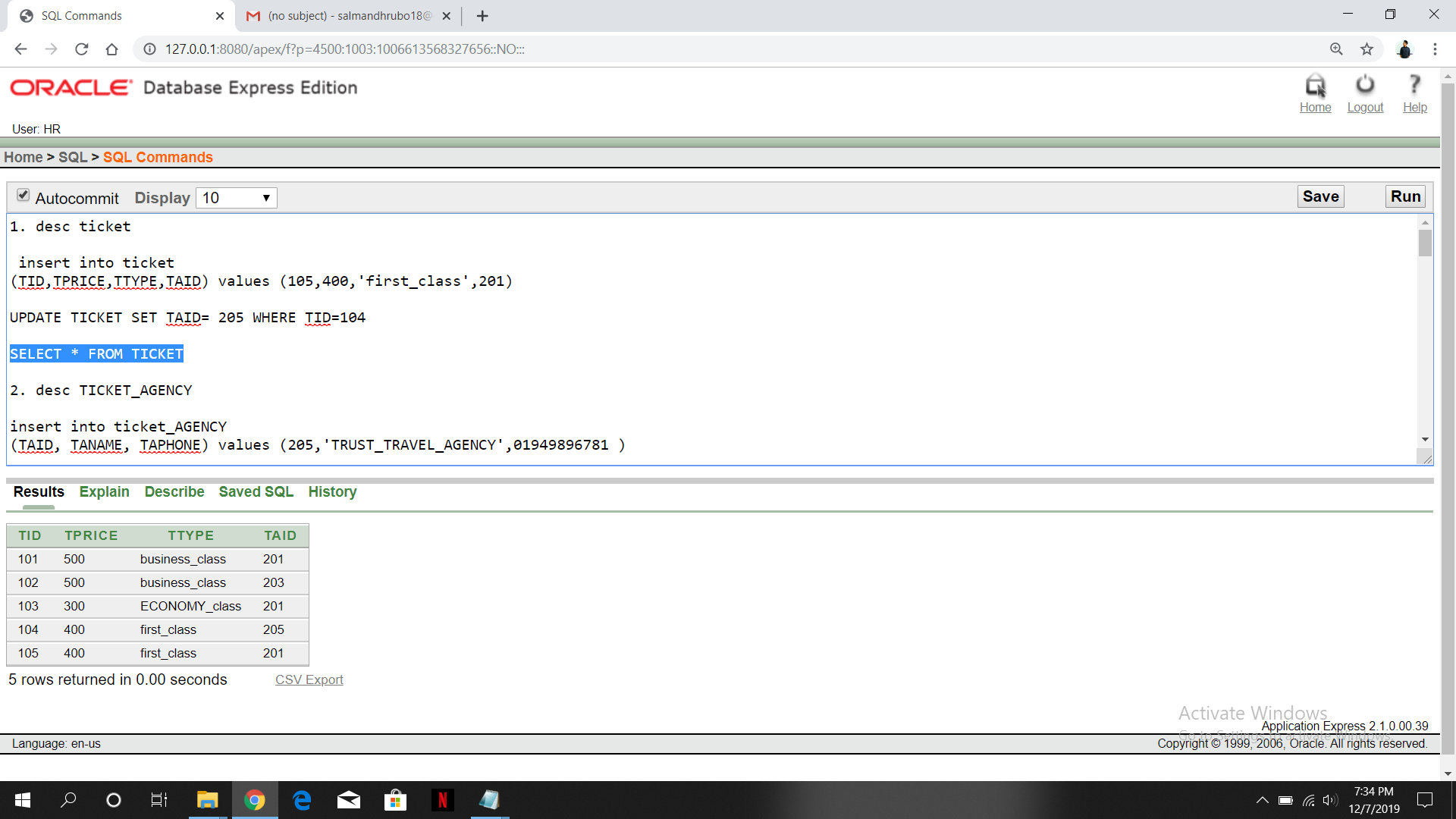


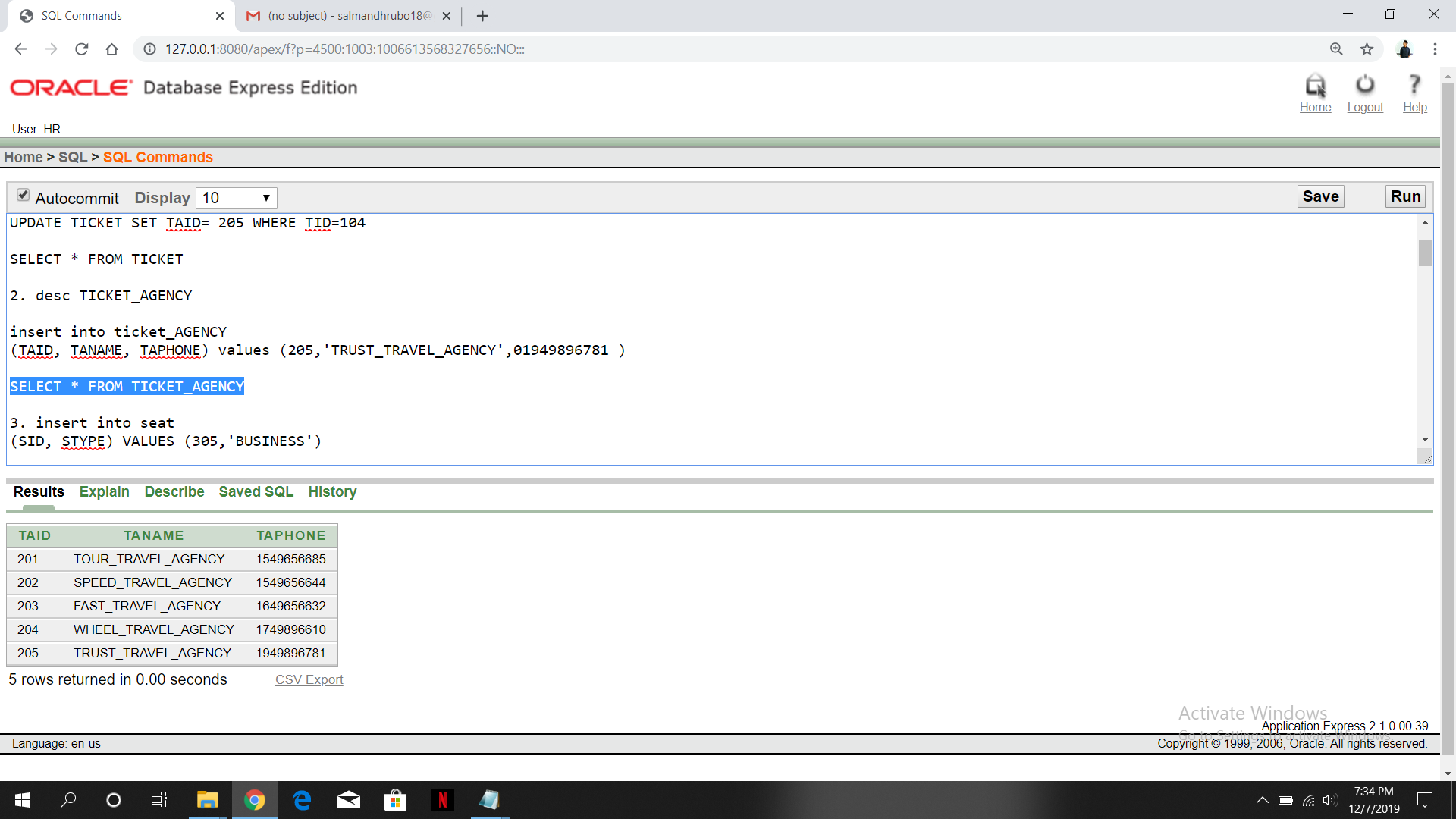


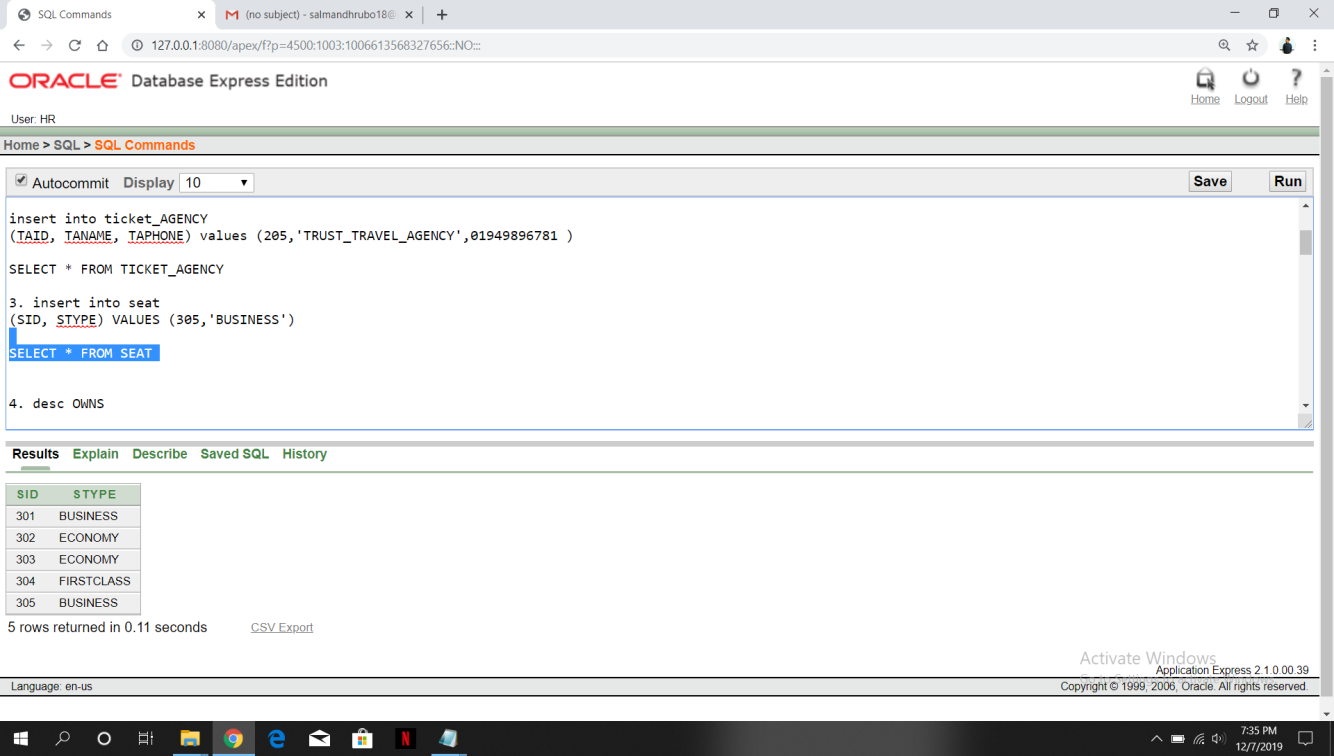


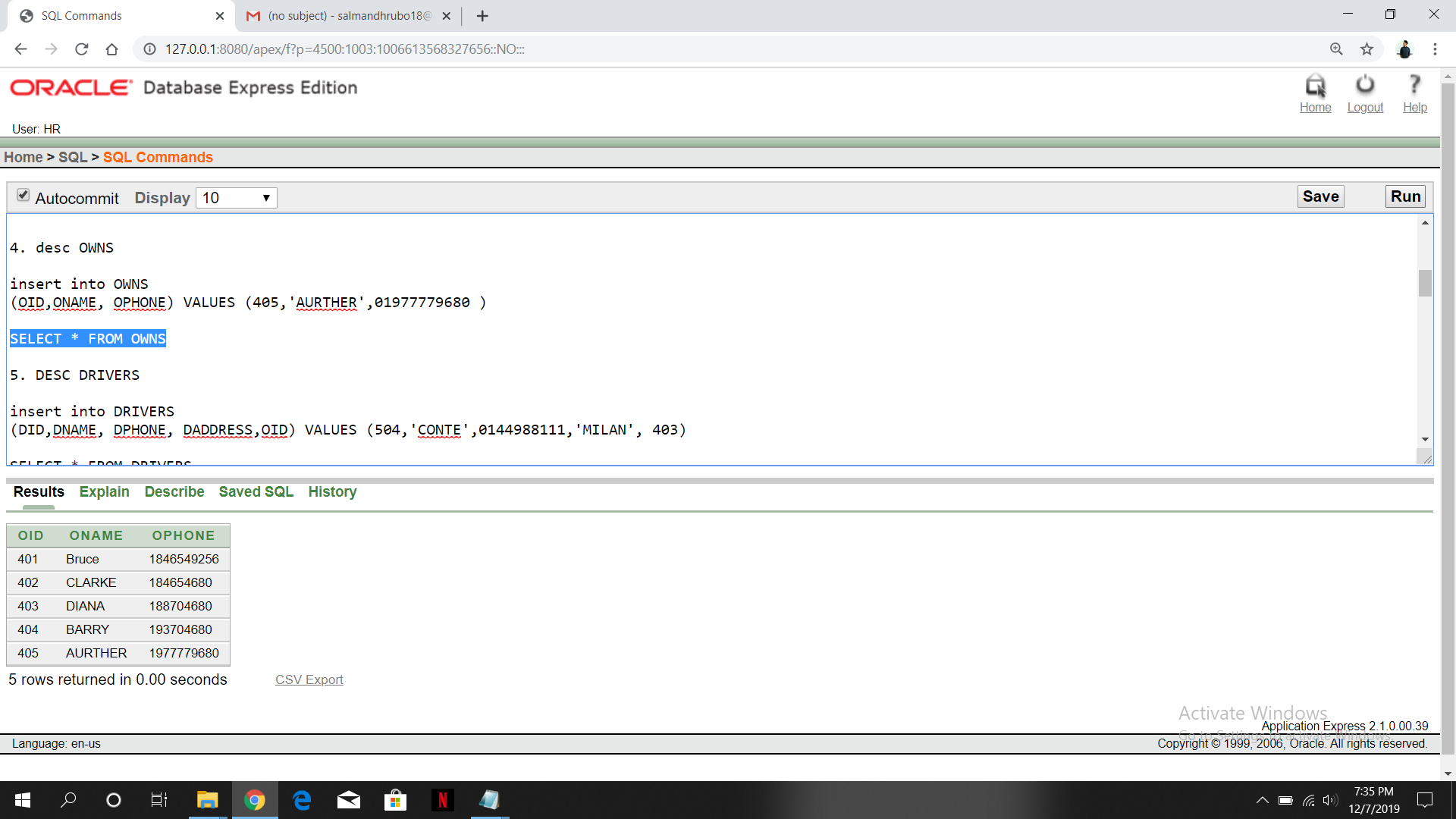
****

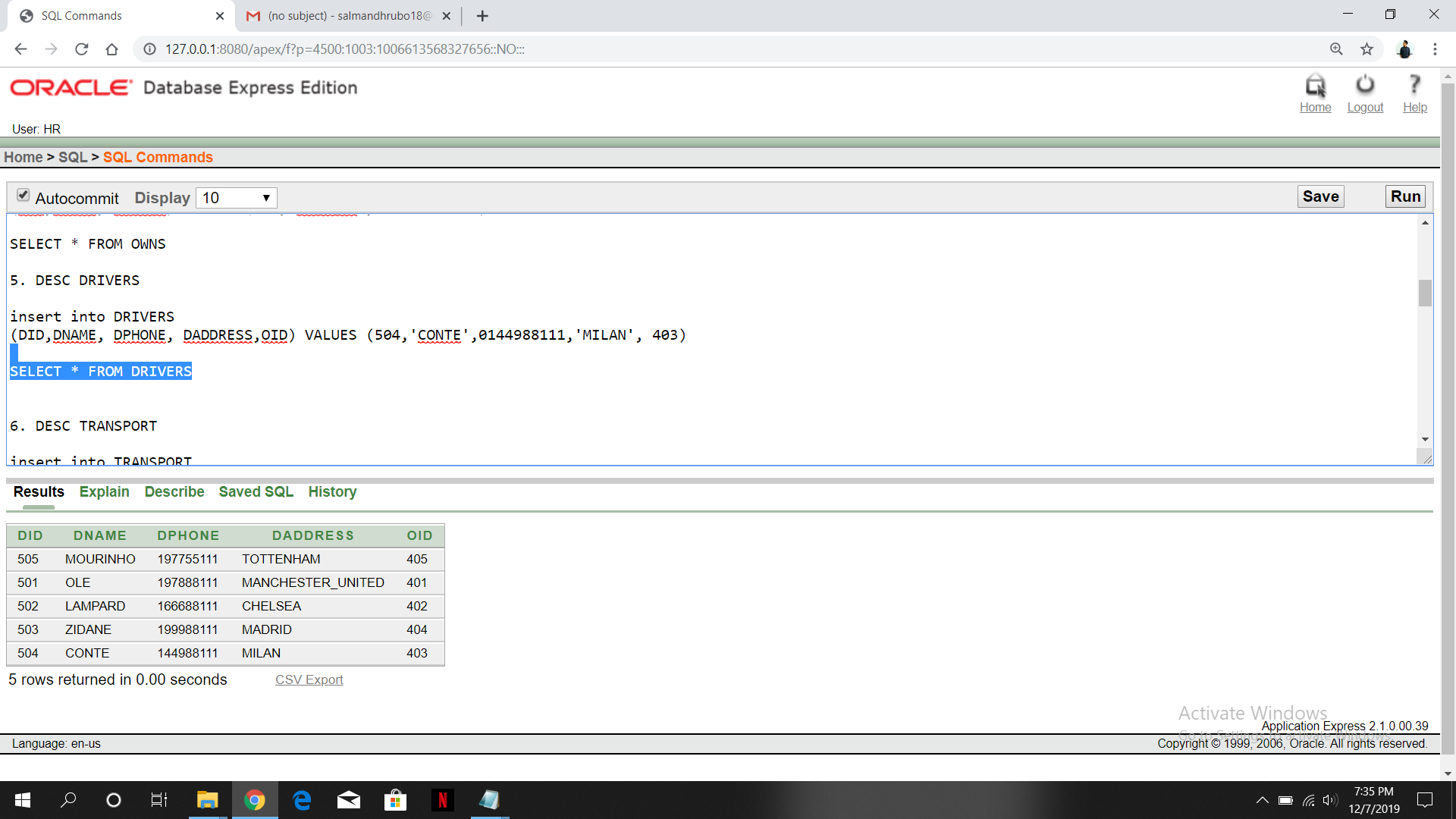
**Insertion**

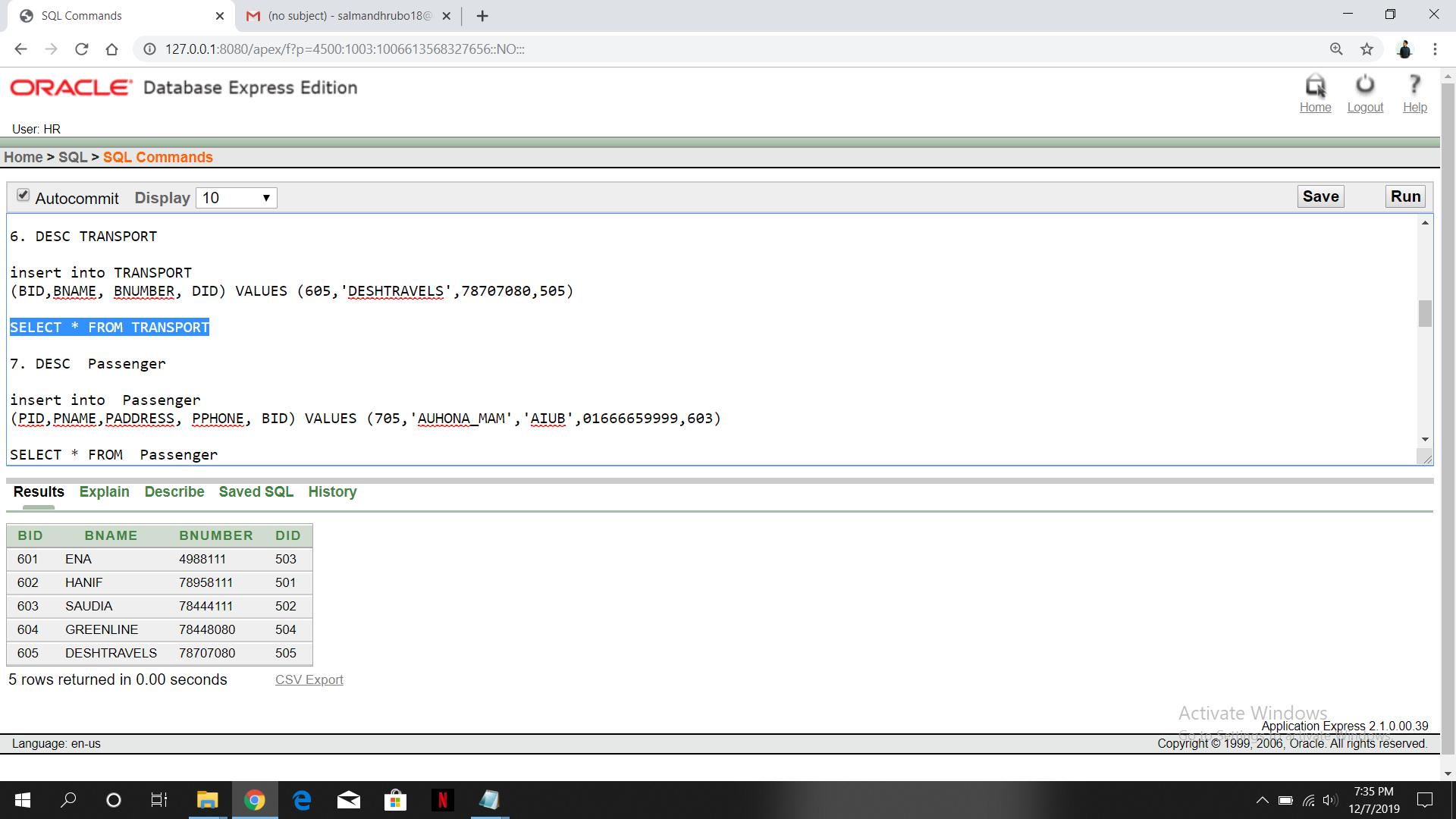
****

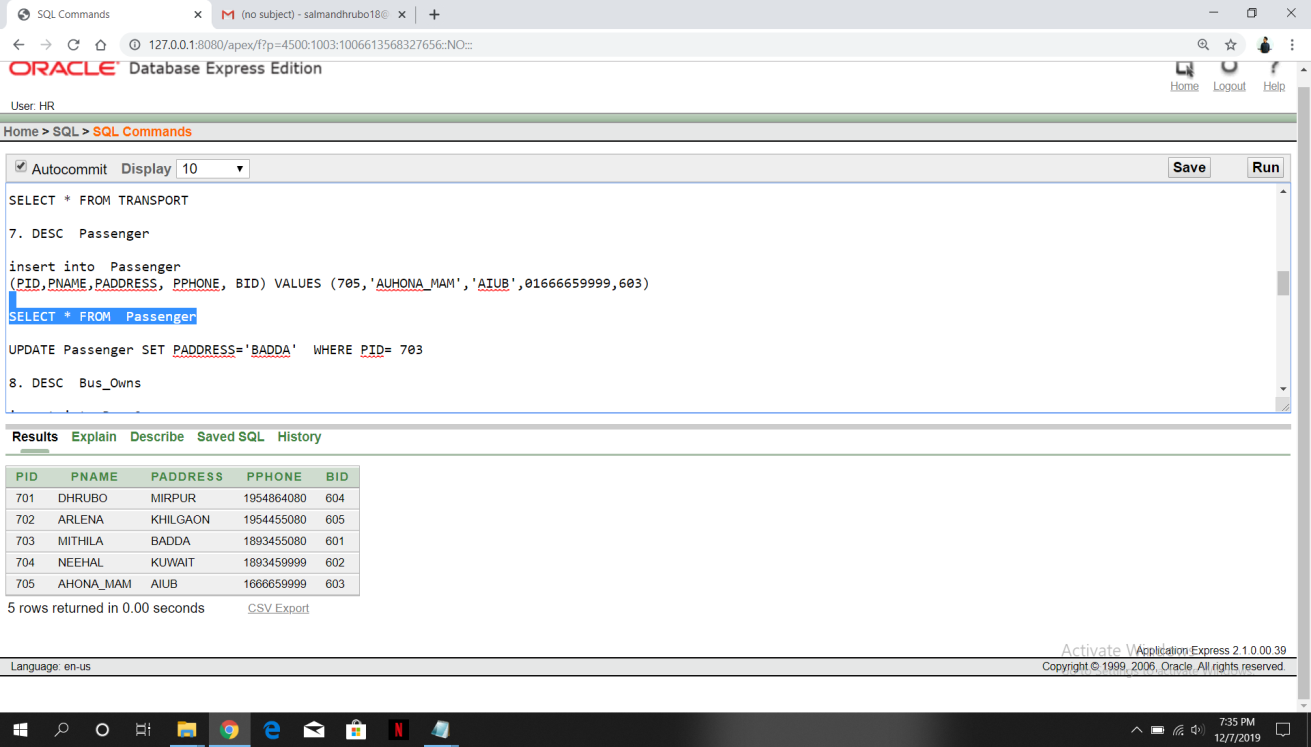
****

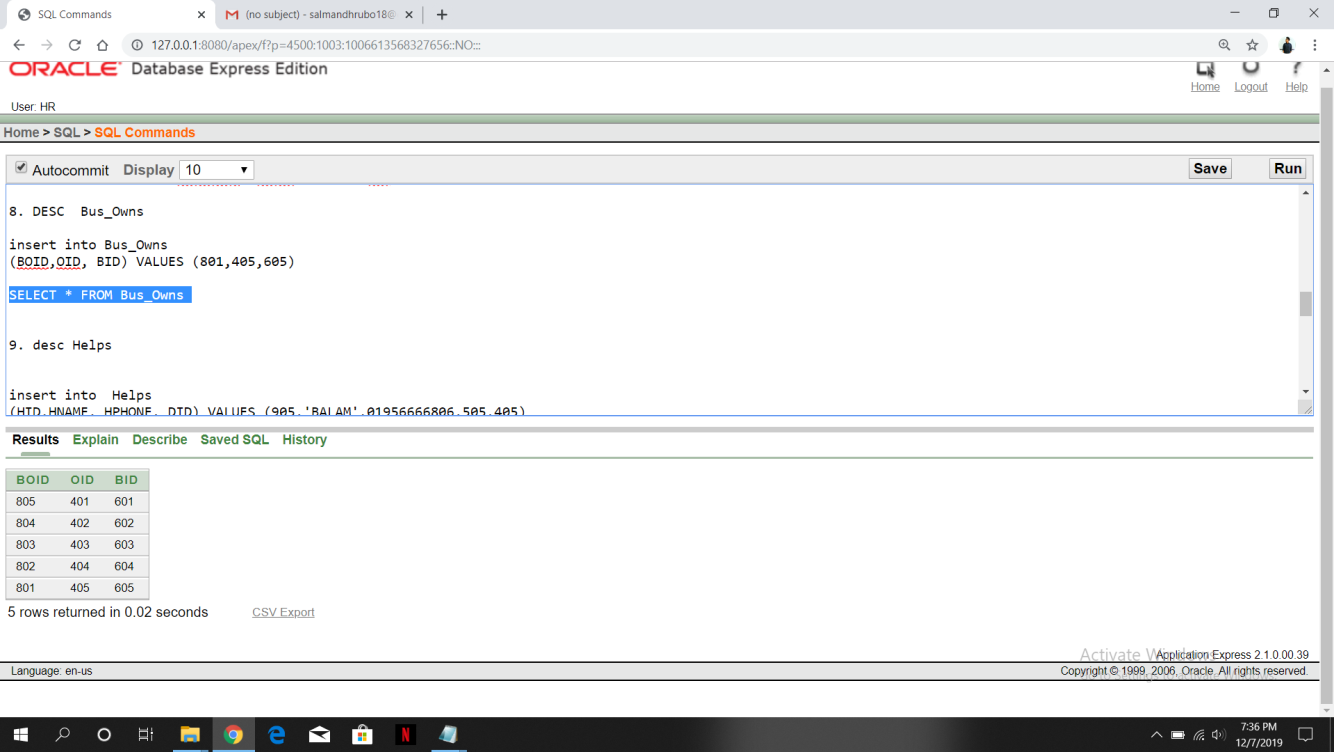
****

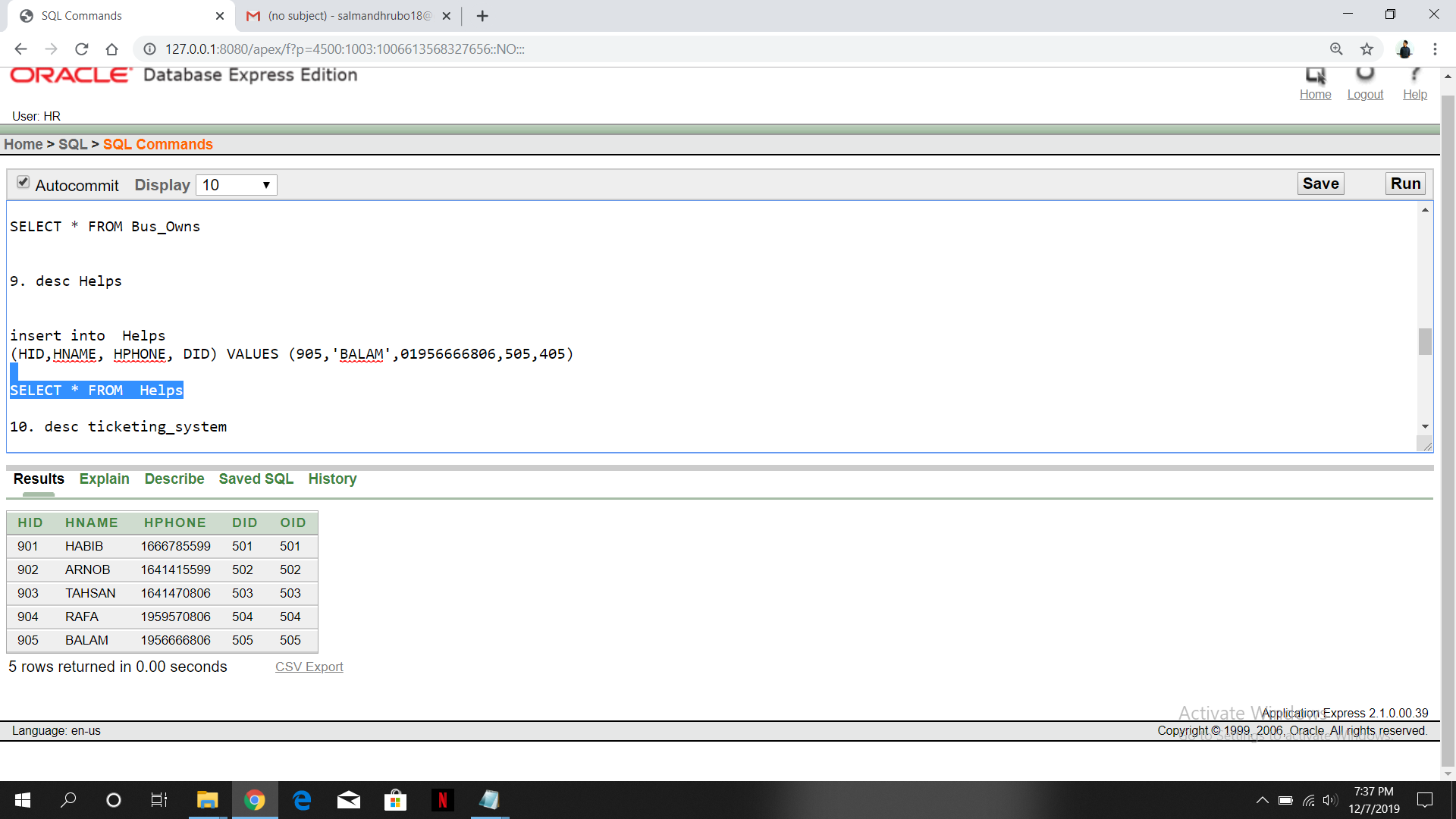
****

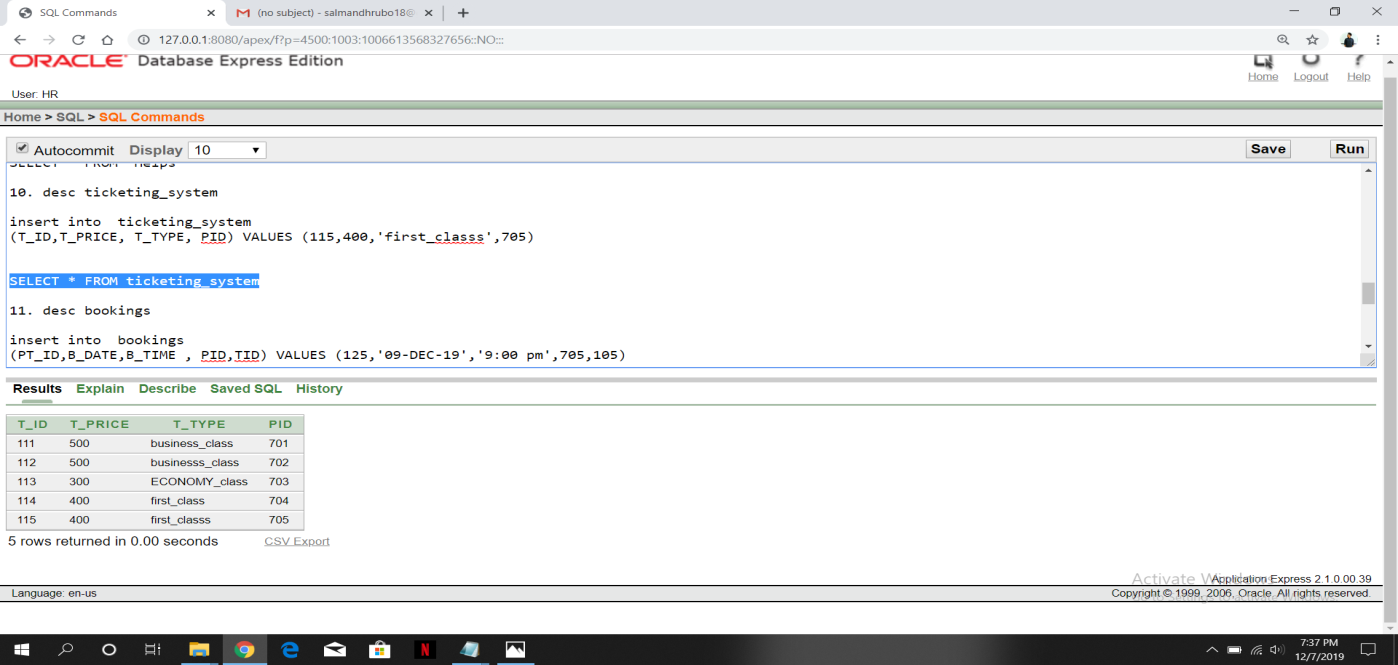
****

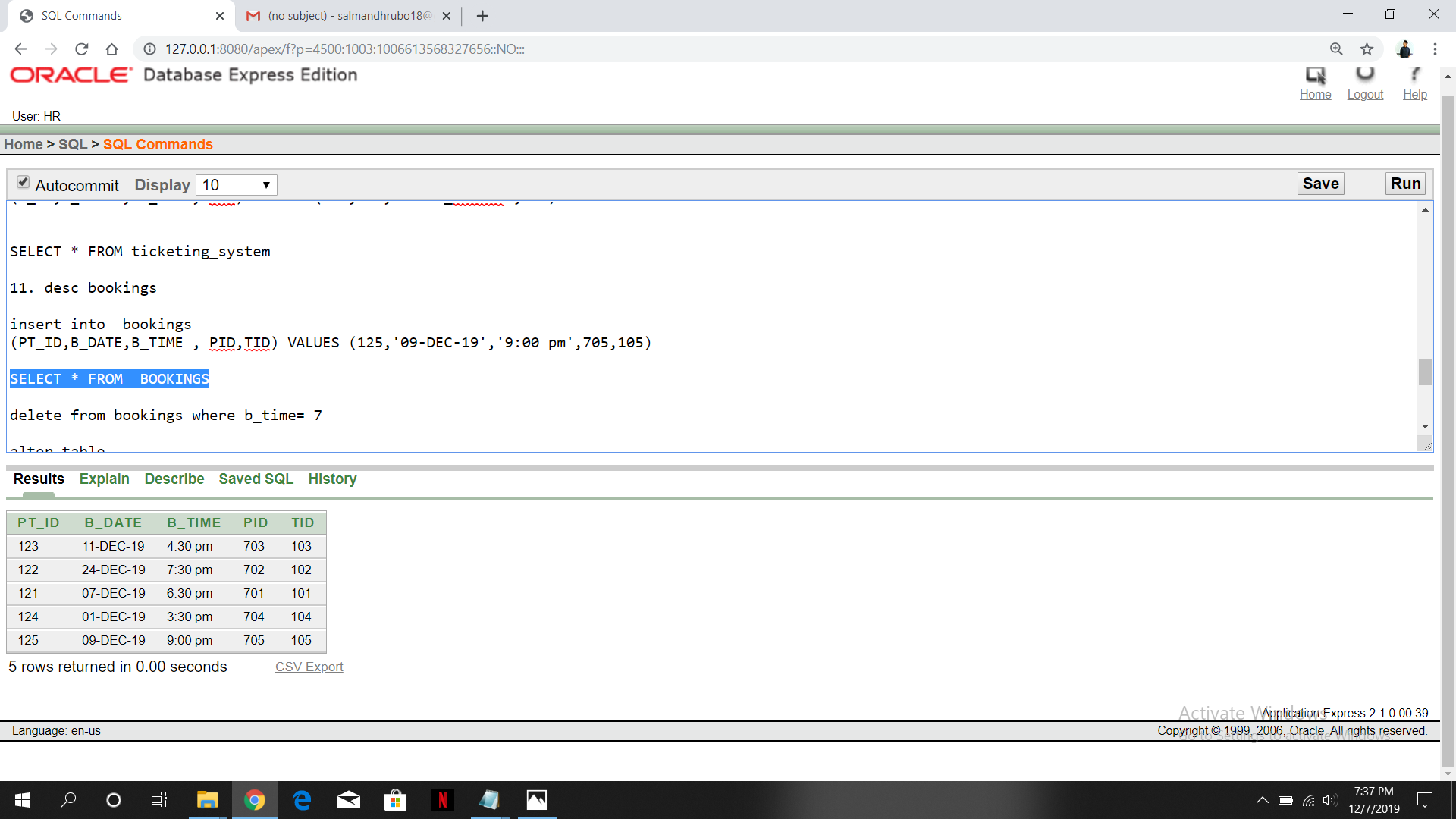
****

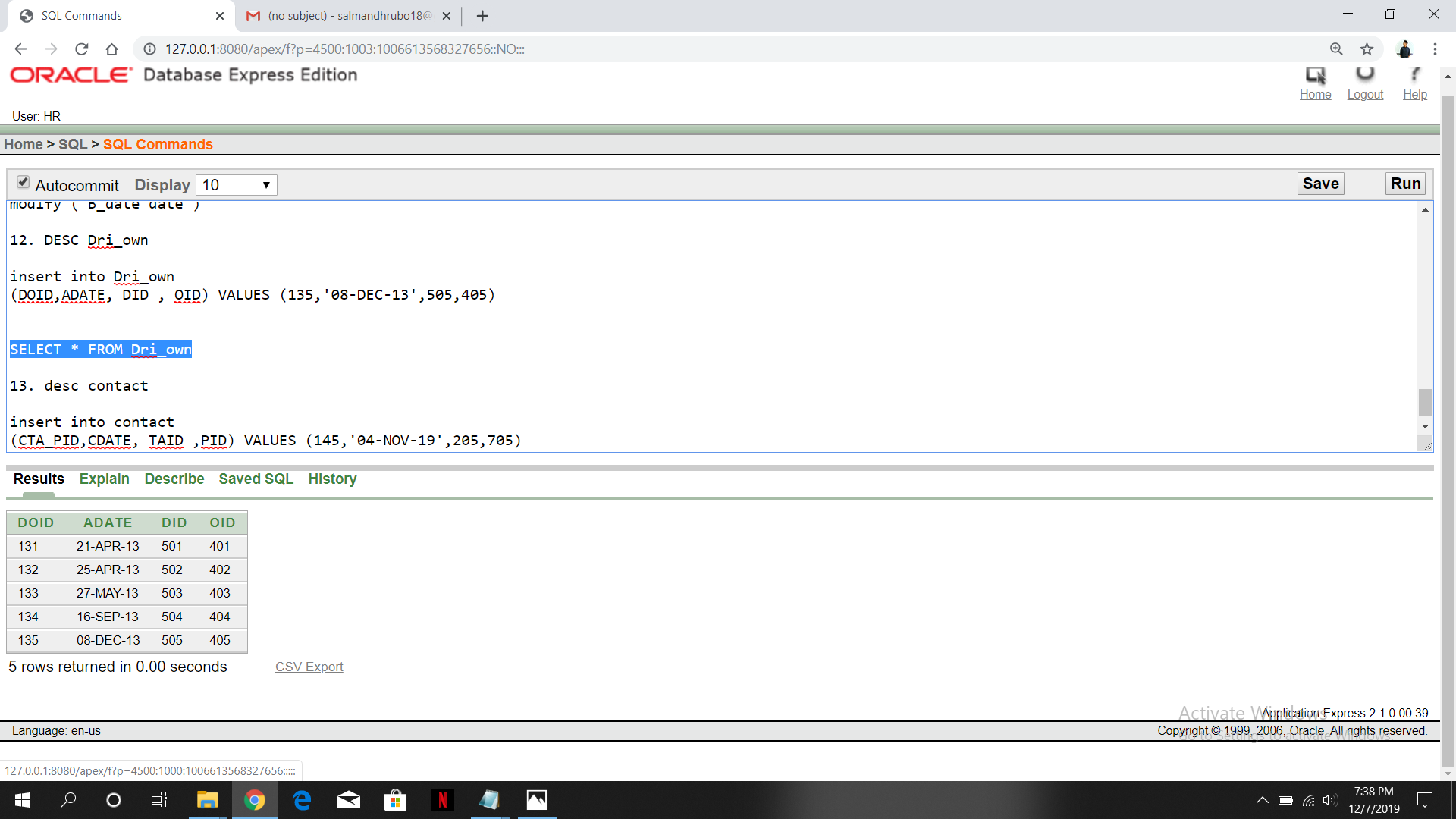
****

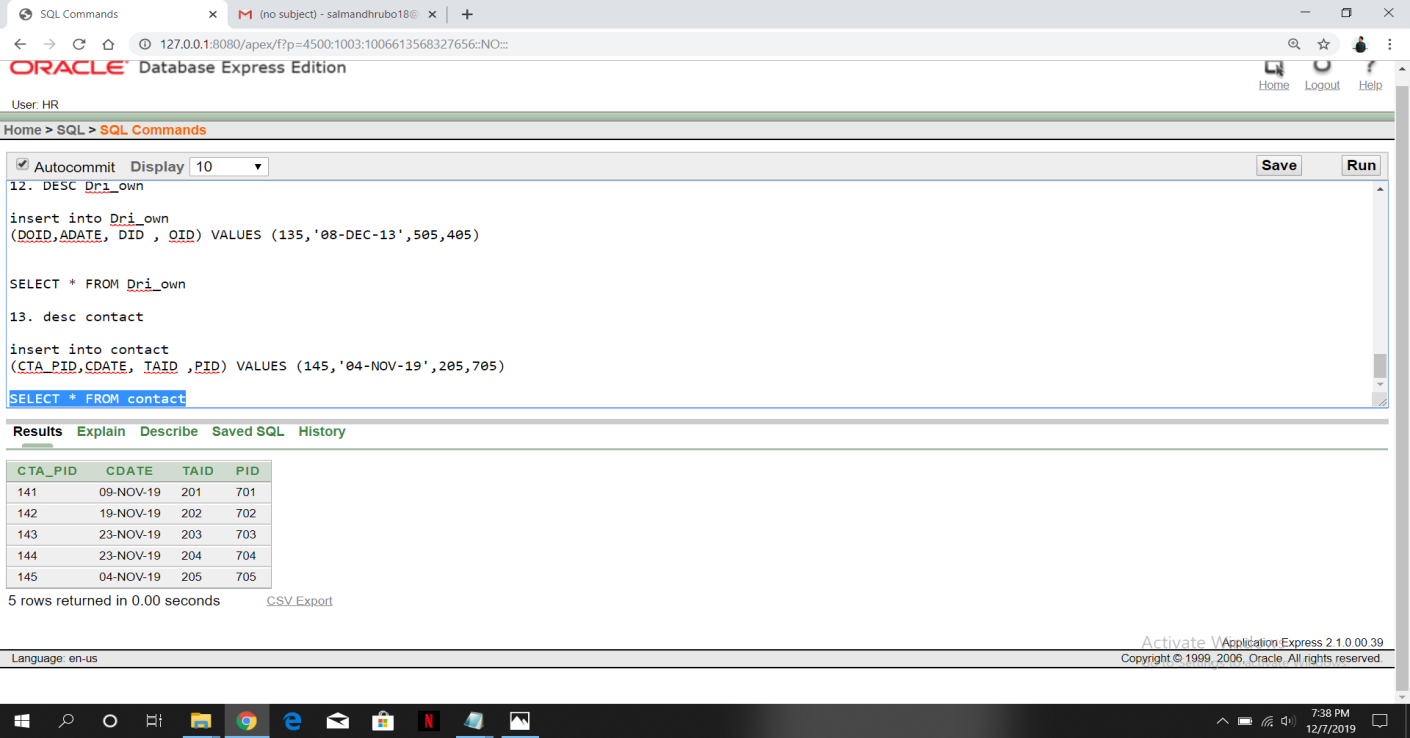
****

****

****

****

****

****

**SQL QUERIES**

1. Find average ticket price and summation of the ticket price. Label the columns as average and sum.

->SELECT AVG(TPRICE),SUM(TPRICE) FROM TICKET

2. Find the maximum ticket price of the business class ticket type.

->SELECT MAX(TPRICE) FROM TICKET WHERE TTYPE= 'BUSINESS\_CLASS'

3. Display the passenger name, passenger\_id whose bus\_id= 601

->SELECT P.PNAME, B.BID, P.PID FROM PASSENGER P , TRANSPORT B WHERE B.BID= P.BID AND B.BID = 601

4. Display the passenger id, passenger name, ticket id whose ticket type is Business\_class.

->SELECT P.PID, P.PNAME, T.TTYPE,T.TID FROM PASSENGER P,TICKET T WHERE P.PID (+) =T.TID AND T.TTYPE= 'BUSINESS\_CLASS'

5. Display the Driver name, Driver id, Helper name

->SELECT D.DNAME , H.DID, H.HNAME FROM DRIVERS D, HELPS H WHERE D.DID= H.DID

6. Find out passenger id, passenger name whose Bus id is 605.

-> SELECT PID,PNAME,BID FROM PASSENGER WHERE BID= (SELECT BID FROM PASSENGER WHERE BID= 605)

7. Find out passenger id, ticket type whose ticket type is same as passenger id= 701

-> SELECT PID,TTYPE FROM PASSENGER WHERE TTYPE= (SELECT TTYPE FROM TICKET WHERE PID=701)

8. Display the Driver id, Driver name who was appointed after

‘21-April-2019’

-> SELECT DID,DNAME,ADATE FROM DRIVERS, DRI\_OWN WHERE ADATE = (SELECT ADATE FROM DRI\_OWN WHERE ADATE= '21-APR-13' )

9. Find out Bus name, Bus id whose Driver id is equal to Bus name =

‘Green Line’.

-> SELECT BNAME, BID FROM TRANSPORT WHERE DID = (SELECT DID FROM TRANSPORT WHERE BNAME= 'GREENLINE'

1. Create a view passenger, that contains details of passengers whose Bus id = 603

->Create view passvu

as select Pid, Pname, Paddress, Pphone, bid

from passenger

where bid=603

**RELATIONAL ALGEBRA**

1. Find average ticket price and summation of the ticket price. Label the columns as average and sum.

2. Find the maximum ticket price of the business class ticket type.

3. Display the passenger name, passenger\_id whose bus\_id= 601

4. Display the passenger id, passenger name, ticket id whose ticket type is Business\_class.

5. Display the Driver name, Driver id, Helper name

6. Find out passenger id, passenger name whose Bus id is Arlena.

7. Find out passenger id, ticket type whose ticket type is same as passenger id= 701

8. Display the Driver id, Driver name who was appointed after

‘21-April-2019’

9. Find out Bus name, Bus id whose Driver id is equal to Bus name =

‘Green Line’.

CARTESIAN :

1. Find out ticket price and ticket agency name whose id is

->

2. Find out passenger name and the name of the bus whose B.name is “Green Dhaka”

->

FULL OUTER JOIN:

1. Find out passenger name and the name of the bus whose B.name is “Green Dhaka”

->

2. Find out all details of passenger, bookings and ticket details of all passengers.

->

LEFT OUTER JOIN:

1. Find out ticket details, booking time where passenger name is “Dhrubo”

->

2. find out passenger details, B.date where T.id =

->

RIGHT OUTER JOIN:

1. Find out all details of bookings done by passenger name= “ Arlena” and travel agency name=

->

2. Find out ticket details, booking time where passenger name is “Neehal”

->

**CONCLUSION**

In this project we have built a “Bus Counter Management System”.

It is developed in visual basic and database has been built in SQL only.

For elaborating this system we have developed a simple E-R diagram to

represent the entire project visually.

Overall this project teaches us about the essential skills as follows:

* Using design techniques to build a data flow diagram
* Understanding the language and logic
* Using various queries to run tables.