**CS632**

**HOMEWORK-2**

**FALL-2017**

**Due Date: 10/17/2017**

**PROFESSOR: SUBMITTED BY:**

**JAMES GELLER NITIN MALHOTRA**

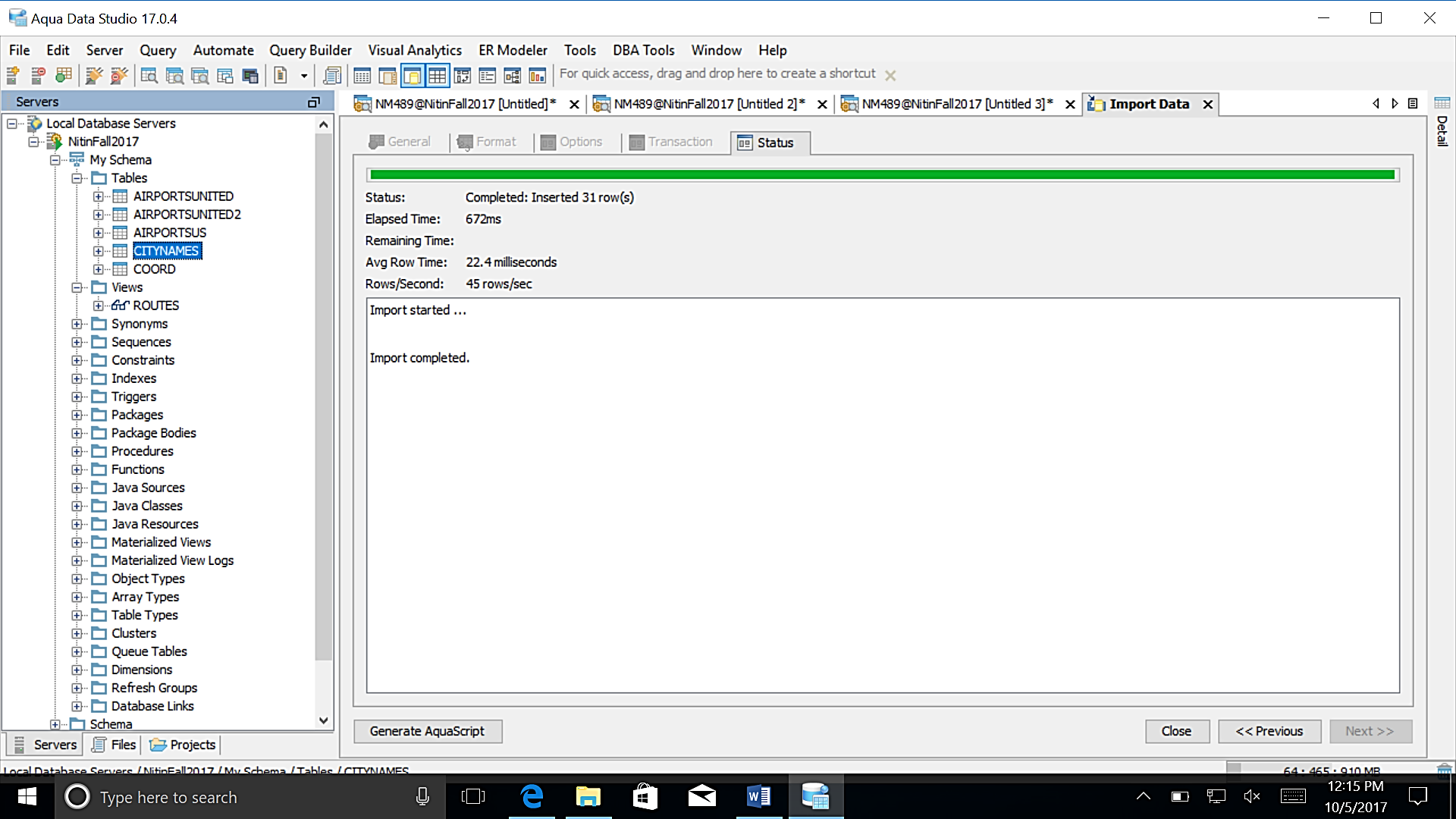
**UCID: nm489**

**1)a) Go to** [**http://www.mapcrow.info/united\_states.html**](http://www.mapcrow.info/united_states.html) **and copy and paste this data into an EXCEL spreadsheet.**

**Delete everything except for the first column with the city names in EXCEL.**

**Import the first column into a table CITYNAMES using AQUA import.**

**Show the table CITYNAMES.**



**b) Create an empty table CITYPAIRS.**

**Write a PL/SQL program using nested cursor loops that will insert into CITYPAIRS all pairs of cities.** However, if you have the pair

ATLANTA AUSTIN

In the table CITIPAIRS then you should **NOT** have the pairs

ATLANTA ATLANTA

AUSTIN ATLANTA

AUSTIN AUSTIN.

In other words, no city appears twice in one row, and no two cities appear together more than once (not even in reverse order).

You may **NOT** use a JOIN to do this. It has to be done purely in PL/SQL with two cursors. **Both have to be initialized by**

**(select \* from CITYNAMES).** In other words, **no WHERE clause** is allowed.

Show the first 50 lines and the last 20 lines of CITYPAIRS.

**CREATE TABLE** CITYPAIRS

(

CITY1 **VARCHAR2**(100) **NOT NULL**,

CITY2 **VARCHAR2**(100) **NOT NULL**

)

**/**

**BEGIN**

**FOR** C1 **IN** (**SELECT** \* **FROM** CITYNAMES)

**LOOP**

**FOR** C2 **IN** (**SELECT** \* **FROM** CITYNAMES)

**LOOP**

**IF** C1.CITY < C2.CITY

**THEN**

**INSERT INTO** CITYPAIRS(CITY1,CITY2)

**VALUES** (C1.CITY,C2.CITY);

**END IF**;

**END LOOP**;

**END LOOP**;

**END**;

**/**

**SELECT** \* **FROM** CITYPAIRS

|  |  |
| --- | --- |
| CITY1 | CITY2 |
| Anchorage | Atlanta |
| Anchorage | Austin |
| Anchorage | Baltimore |
| Anchorage | Boston |
| Anchorage | Chicago |
| Anchorage | Dallas |
| Anchorage | Denver |
| Anchorage | Honolulu |
| Anchorage | Houston |
| Anchorage | Indianapolis |
| Anchorage | Jacksonville |
| Anchorage | Las Vegas |
| Anchorage | Los Angeles |
| Anchorage | Memphis |
| Anchorage | Miami |
| Anchorage | New Orleans |
| Anchorage | New York |
| Anchorage | Newark |
| Anchorage | Oakland |
| Anchorage | Philadelphia |
| Anchorage | Phoenix |
| Anchorage | Portland |
| Anchorage | San Antonio |
| Anchorage | San Diego |
| Anchorage | San Francisco |
| Anchorage | San Jose |
| Anchorage | Seattle |
| Anchorage | Tampa |
| Anchorage | Tucson |
| Anchorage | Washington DC |
| Atlanta | Austin |
| Atlanta | Baltimore |
| Atlanta | Boston |
| Atlanta | Chicago |
| Atlanta | Dallas |
| Atlanta | Denver |
| Atlanta | Honolulu |
| Atlanta | Houston |
| Atlanta | Indianapolis |
| Atlanta | Jacksonville |
| Atlanta | Las Vegas |
| Atlanta | Los Angeles |
| Atlanta | Memphis |
| Atlanta | Miami |
| Atlanta | New Orleans |
| Atlanta | New York |
| Atlanta | Newark |
| Atlanta | Oakland |
| Atlanta | Philadelphia |
| Atlanta | Phoenix |

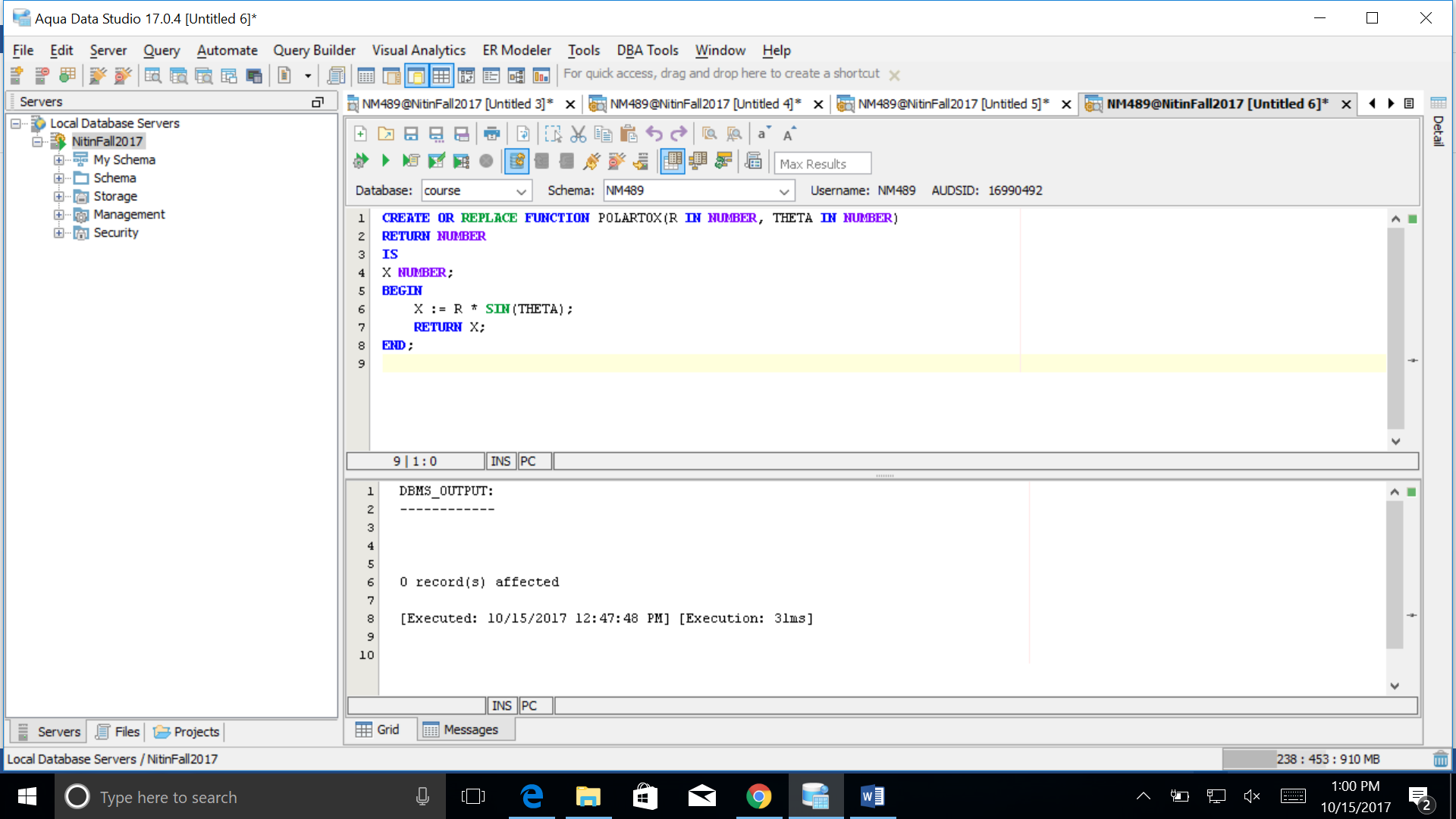
…

|  |  |
| --- | --- |
| San Diego | San Jose |
| San Diego | Seattle |
| San Diego | Tampa |
| San Diego | Tucson |
| San Diego | Washington DC |
| San Francisco | San Jose |
| San Francisco | Seattle |
| San Francisco | Tampa |
| San Francisco | Tucson |
| San Francisco | Washington DC |
| San Jose | Seattle |
| San Jose | Tampa |
| San Jose | Tucson |
| San Jose | Washington DC |
| Seattle | Tampa |
| Seattle | Tucson |
| Seattle | Washington DC |
| Tampa | Tucson |
| Tampa | Washington DC |
| Tucson | Washington DC |

2)a) Write two PL/SQL **functions** called POLARTOX(R in number, Theta in number)returns number and POLARTOY(R in number, Theta in number)returns number.

POLARTOX returns the X coordinate for a given radius and angle.

POLARTOY returns the Y coordinate for a given radius and angle.



**CREATE OR REPLACE FUNCTION** POLARTOY(R **IN NUMBER**, THETA **IN NUMBER**)

**RETURN NUMBER**

**AS**

Y **NUMBER;**

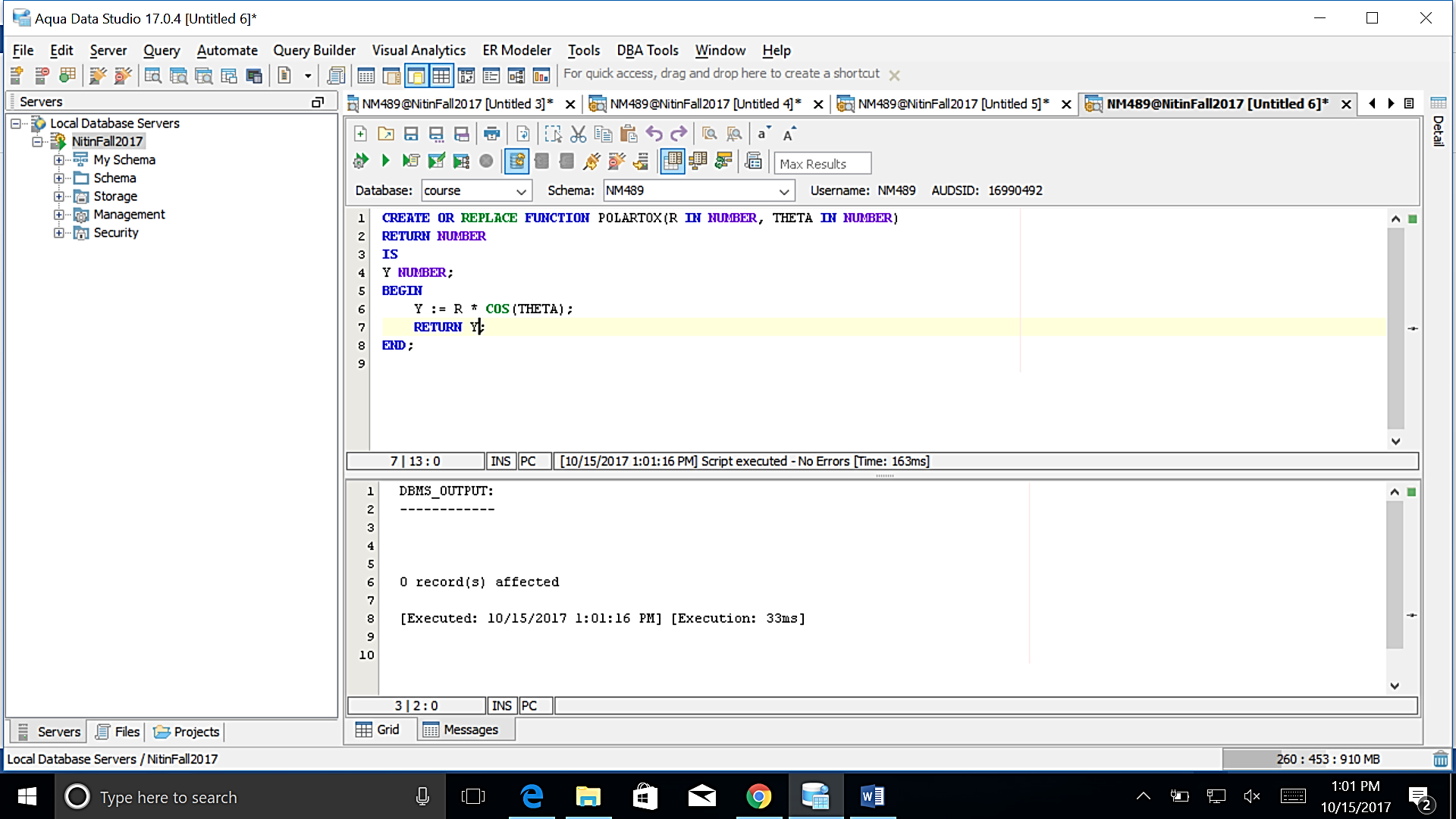
**BEGIN**

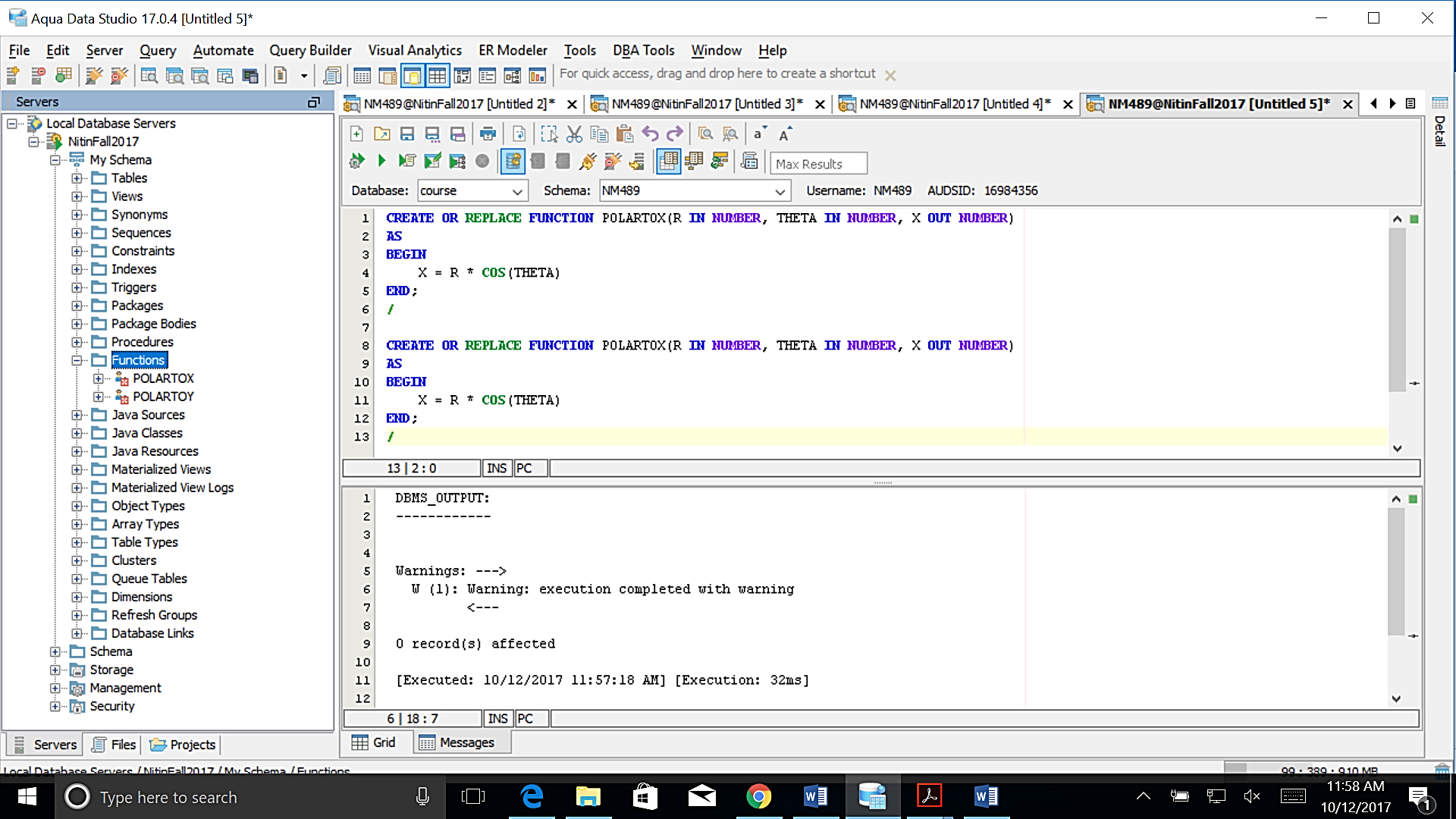
Y := R \* **SIN**(THETA);

**RETURN** Y;

**END**;

**/**





**b) Create a table POLAR with two columns R, THETA. THETA in degrees.**

**Insert the following values into the table POLAR (by hand): [1 point]**

2 0

2 45

2 90

2 135

2 180

2 225

2 270

2 315

5 0

5 45

5 90

5 135

5 180

5 225

5 270

5 315

**CREATE TABLE** POLAR (

R **INT**,

THETA **INT**

)

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,0)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,45)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,90)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,135)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,180)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,225)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,270)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,315)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,0)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,45)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,90)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,135)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,180)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,225)

**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,270)

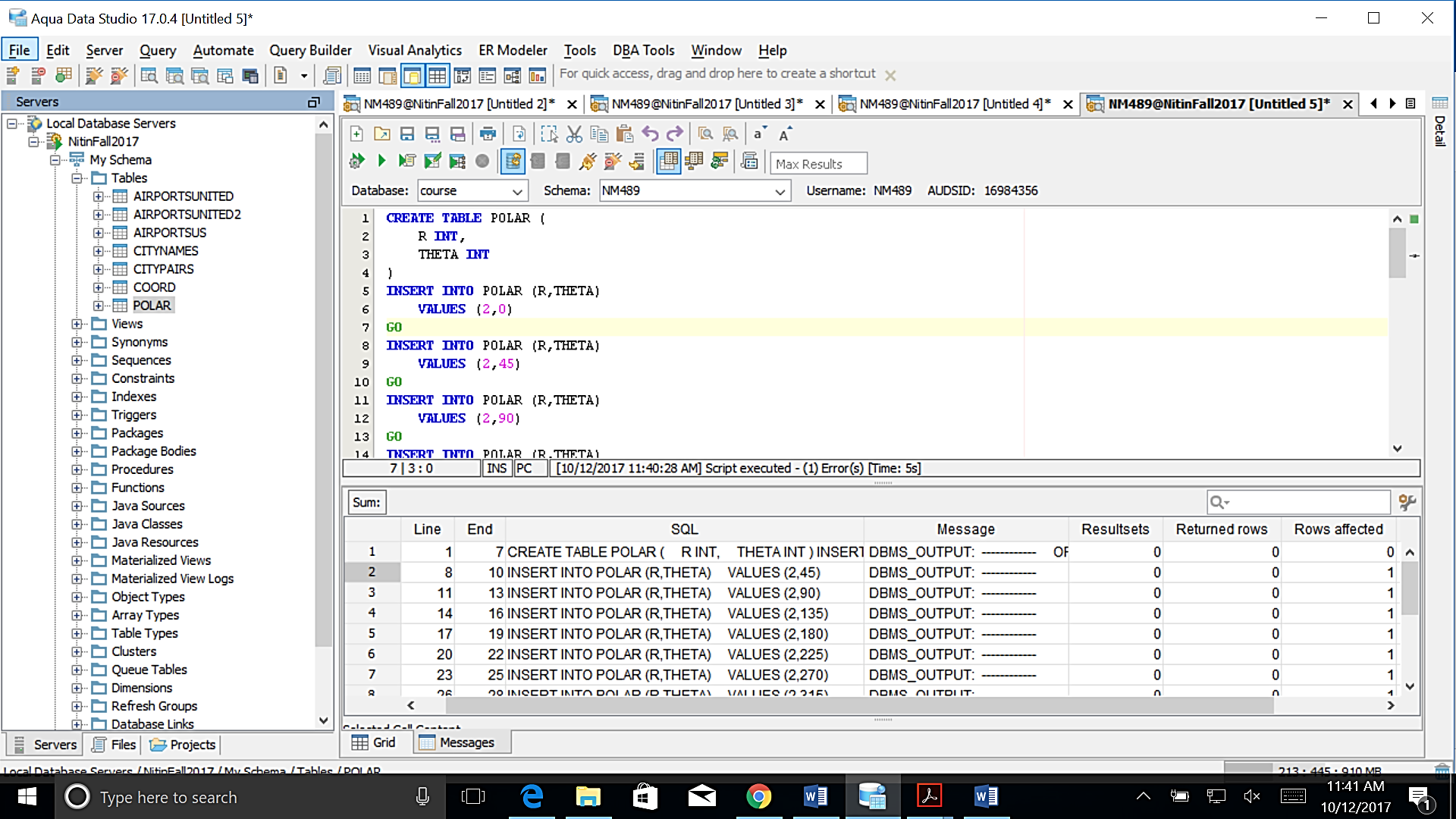
**GO**

**INSERT INTO** POLAR (R,THETA)

**VALUES** (5,315)

**GO**

|  |  |
| --- | --- |
| R | THETA |
| 2 | 45 |
| 2 | 90 |
| 2 | 135 |
| 2 | 180 |
| 2 | 225 |
| 2 | 270 |
| 2 | 315 |
| 5 | 0 |
| 5 | 45 |
| 5 | 90 |
| 5 | 135 |
| 5 | 180 |
| 5 | 225 |
| 5 | 270 |
| 5 | 315 |



**c) Write a PL/SQL main program that displays on the screen for every row of the table POLAR the values R, THETA, X, Y, by calling for every row of POLAR the two functions POLARTOX and POLARTOY.**

**DECLARE**

A **NUMBER**;

B **NUMBER**;

C **NUMBER**;

D **NUMBER**;

**CURSOR** XYRT **IS SELECT** R,THETA **FROM** POLAR;

**BEGIN**

**OPEN** XYRT;

**LOOP**

**FETCH** XYRT **INTO** A,B;

**IF** XYRT%**FOUND THEN**

C := POLARTOX(A,B);

D := POLARTOY(A,B);

DBMS\_OUTPUT.PUT\_LINE(C || ', ' || D || ', ' || A || ', ' || B || 'DEGREES');

**ELSE**

EXIT;

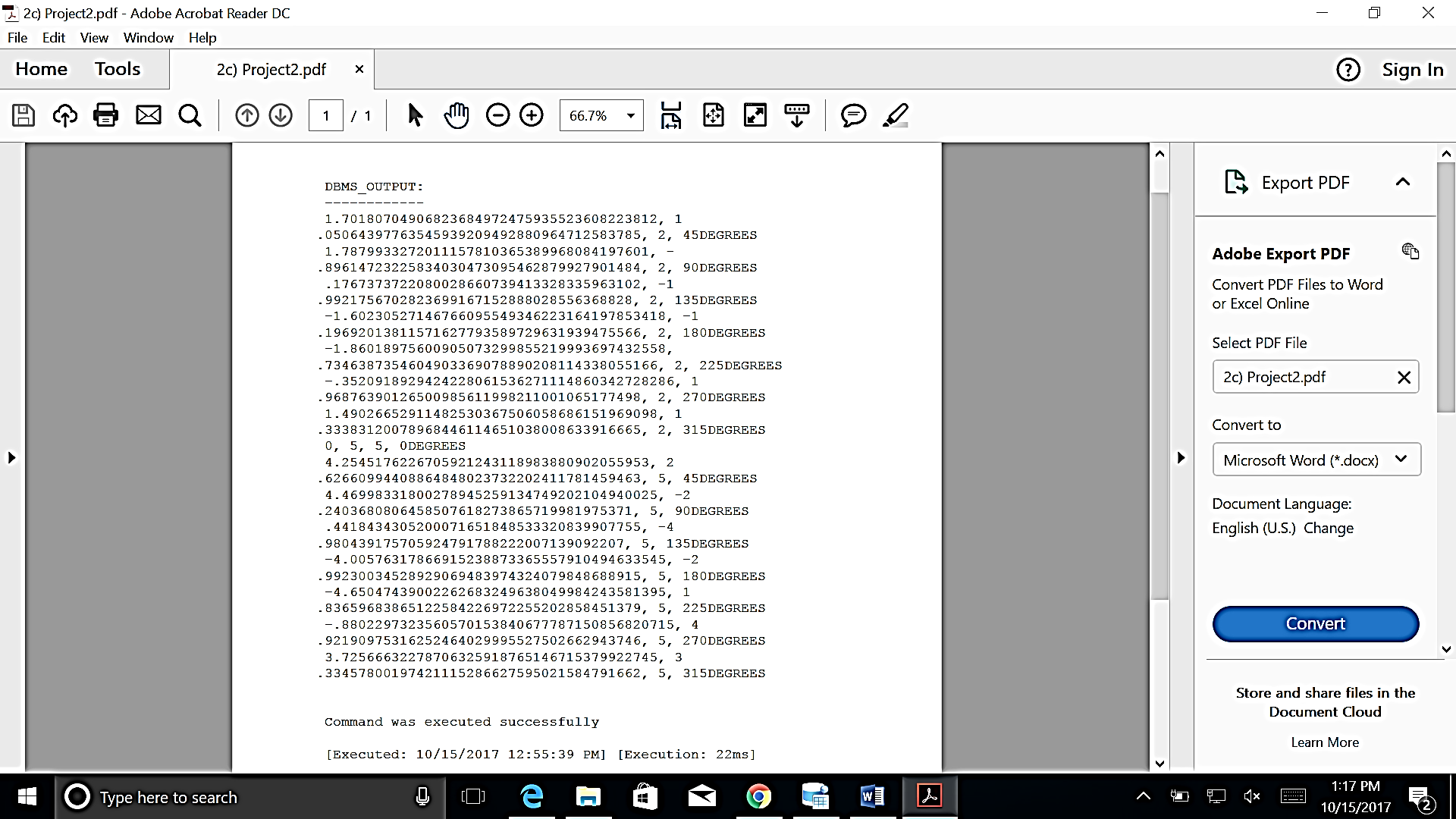
**END IF**;

**END LOOP**;

**CLOSE** XYRT;

**END**;

**/**



**d) Insert two new rows into POLAR.**

2 360

5 360.

**INSERT INTO** POLAR (R,THETA)

**VALUES** (2,360)

**GO**

**INSERT INTO** POLAR (R,THETA)

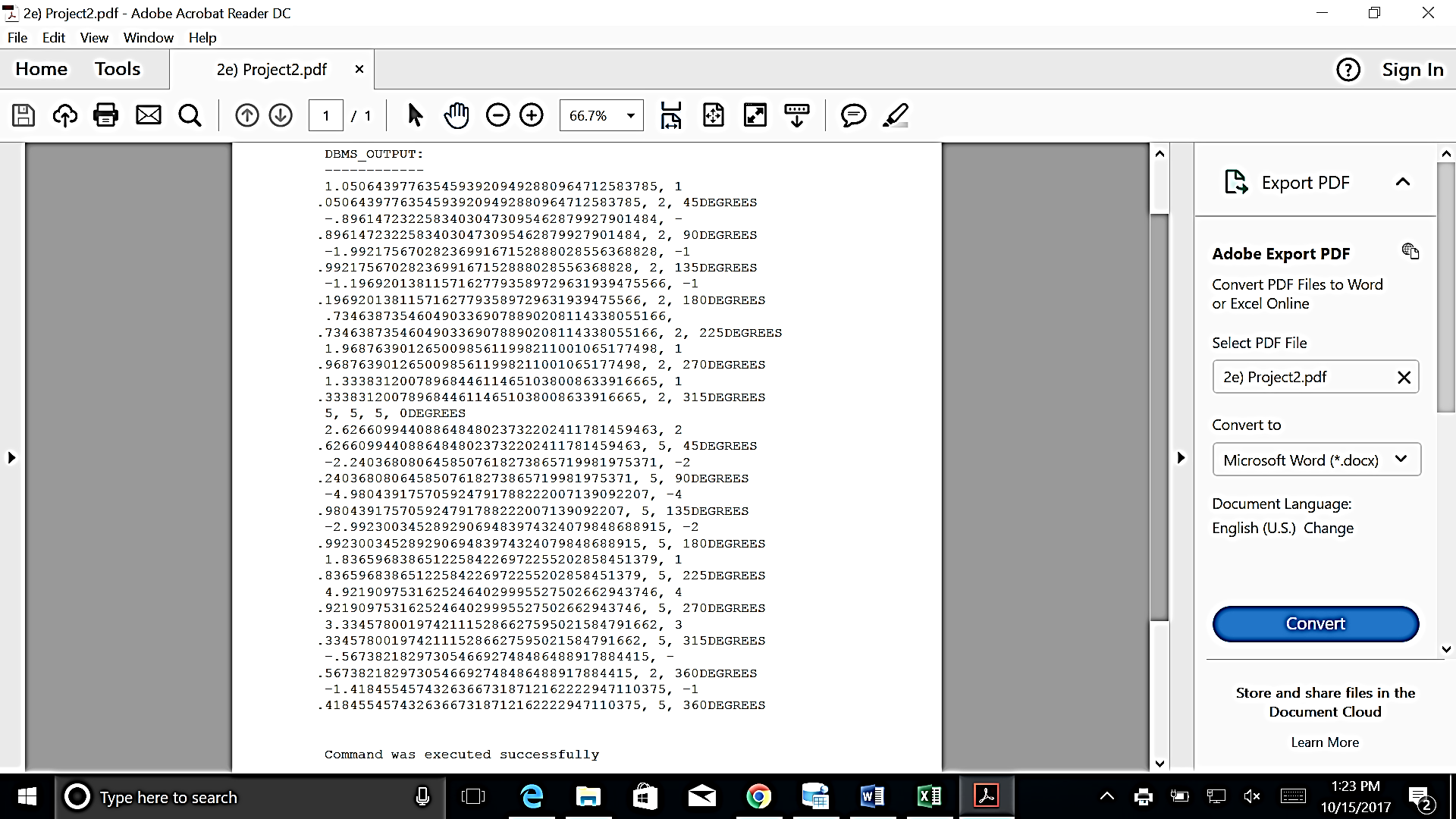
**VALUES** (5,360)

**GO**

**SELECT** \* **FROM** POLAR

|  |  |
| --- | --- |
| R | THETA |
| 2 | 45 |
| 2 | 90 |
| 2 | 135 |
| 2 | 180 |
| 2 | 225 |
| 2 | 270 |
| 2 | 315 |
| 5 | 0 |
| 5 | 45 |
| 5 | 90 |
| 5 | 135 |
| 5 | 180 |
| 5 | 225 |
| 5 | 270 |
| 5 | 315 |
| 2 | 360 |
| 5 | 360 |

**e) Rerun the main program from question c) without changing it. IT SHOULD STILL WORK for all the rows. If it does not, you need to go back and fix it.**

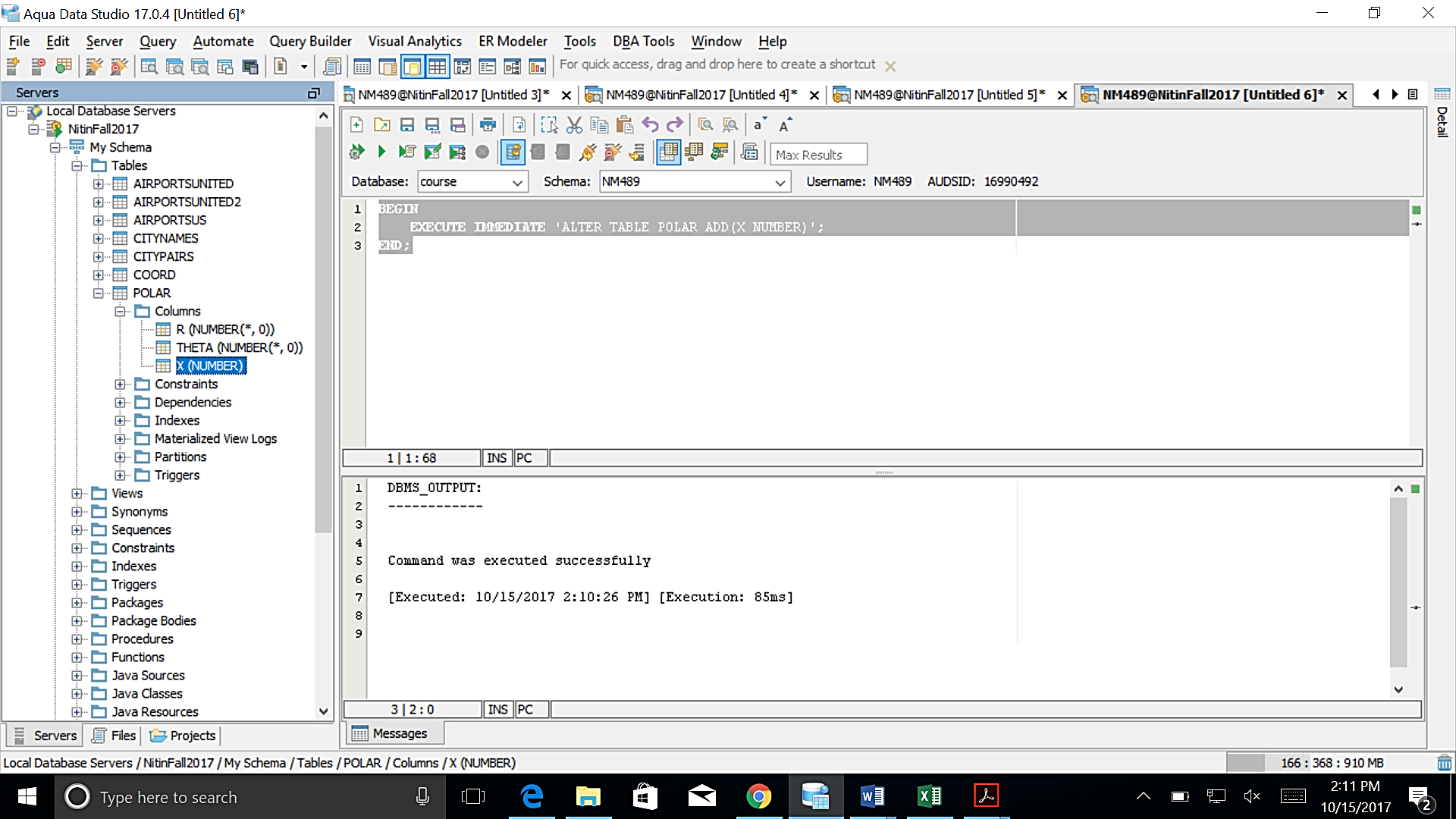


**3)a) Write two ALTER statements that add columns called X and Y to the table POLAR. But do it from within two PL/SQL programs.**

**BEGIN**

**EXECUTE IMMEDIATE** 'ALTER TABLE POLAR ADD(X NUMBER)';

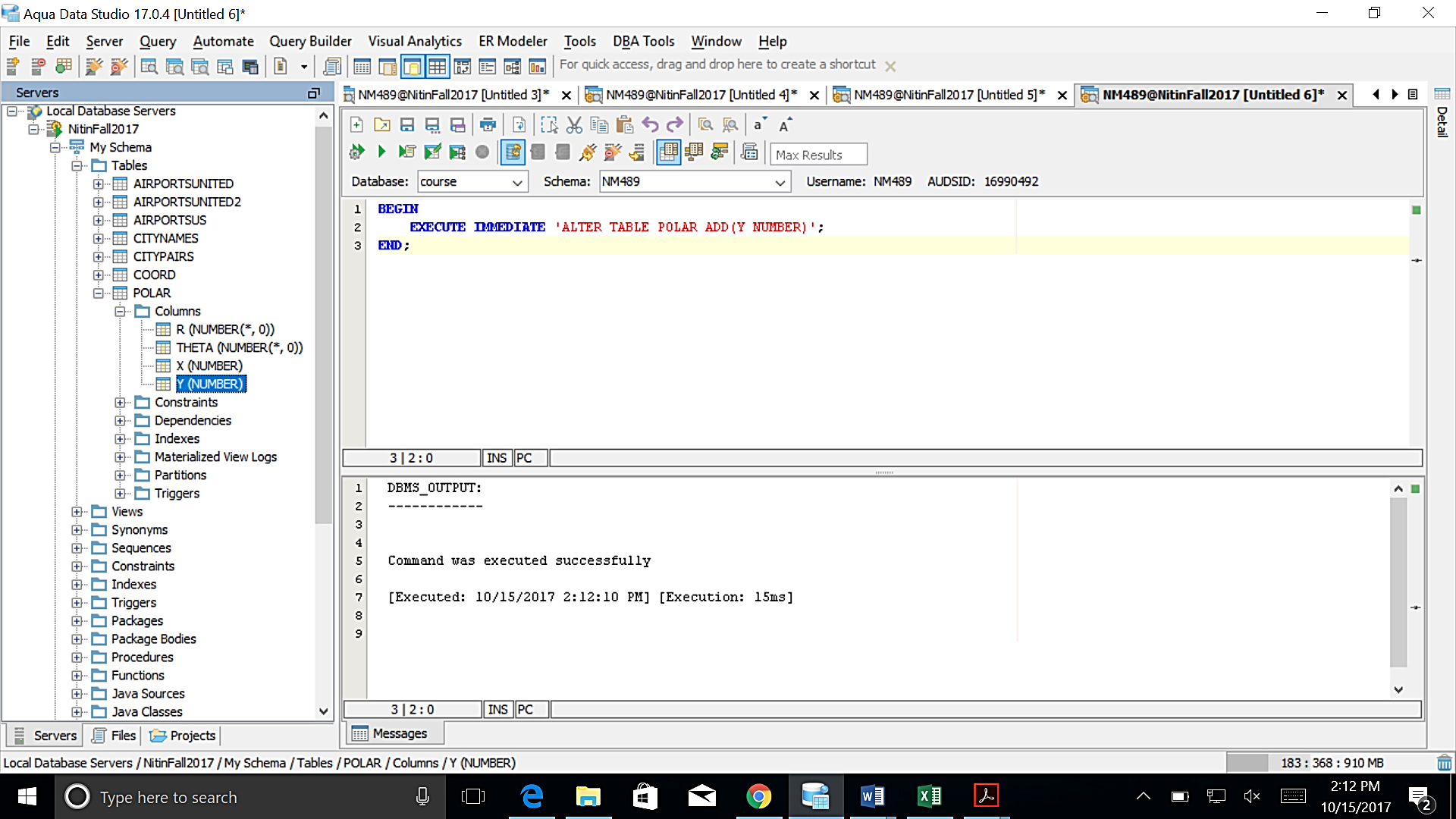
**END**;



**BEGIN**

**EXECUTE IMMEDIATE** 'ALTER TABLE POLAR ADD(Y NUMBER)';

**END**;



**b) Write a new main program that does the same things as the main program in 2)c) but also UPDATES the table POLAR so that the values X and Y are now stored into the table POLAR.**

Show the new table POLAR.

Hint: Use an Update Cursor.

**BEGIN**

**FOR** UP **IN** (**SELECT** \* **FROM** POLAR)

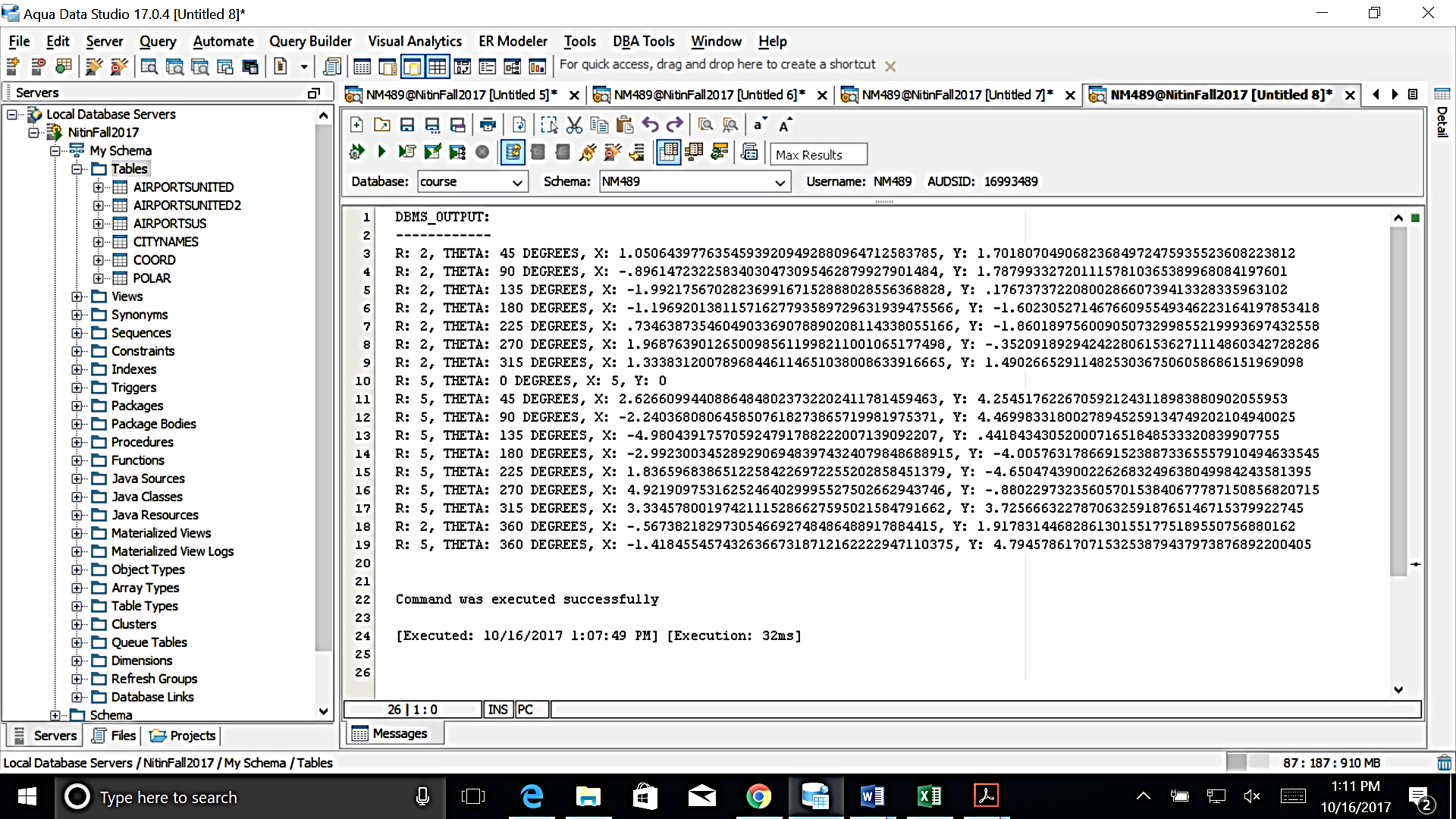
**LOOP**

DBMS\_OUTPUT.PUT\_LINE('R: '||UP.R ||', '||'THETA: '|| UP.THETA || ' DEGREES' ||', '||'X: '|| POLARTOX(UP.R, UP.THETA) ||', '||'Y: '|| POLARTOY(UP.R, UP.THETA));

**END LOOP**;

**END**;

**/**



4) Draw a UML diagram expressing the following knowledge.

CANINE IS-A ANIMAL

FELINE IS-A ANIMAL

DOG IS-A CANINE

WOLF IS-A CANINE

CAT IS-A FELINE

LION IS-A FELINE

WILD-ANIMAL IS-A ANIMAL

DOMESTIC-ANIMAL IS-A ANIMAL

LION IS-A WILD-ANIMAL

WOLF IS-A WILD-ANIMAL

DOG IS-A DOMESTIC-ANIMAL

CAT IS-A DOMESTIC-ANIMAL

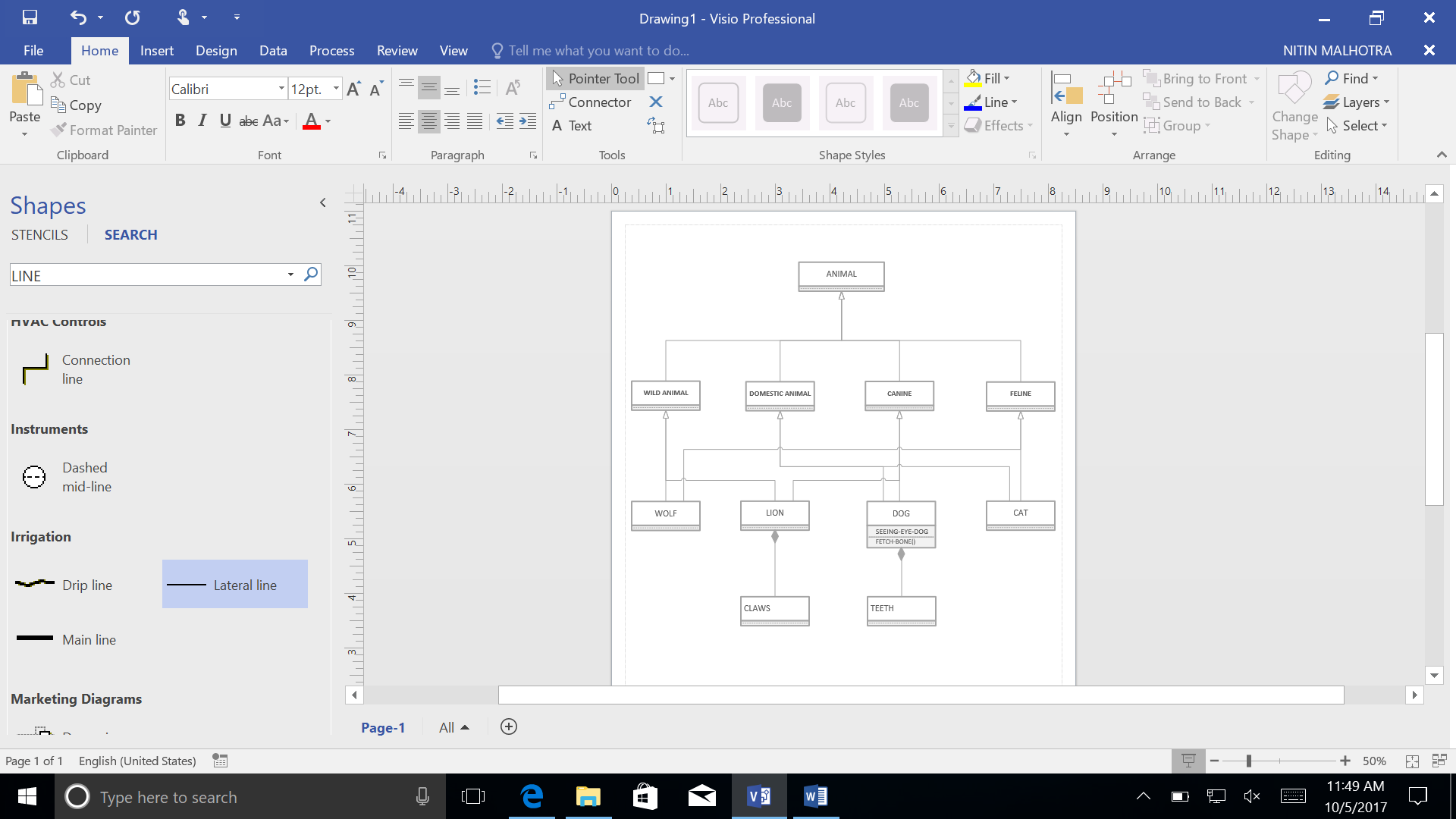
LION HAS-PART CLAWS

DOG HAS-PART TEETH

DOG has a BOOLEAN attribute: SEEING-EYE-DOG

DOG has an operation: FETCH-BONE()

You can draw the UML diagram with any tool you like. NetBeans, Rational, ERWIN (if you have access to it), Visio, or even just MS Word.



**UML Diagram drawn using MS Visio**

5) The following questions should just encourage you to review all material in all lectures.

**a) What is the URL of the annotation website that we are using? [1]**

*https://genius.it/web.njit.edu/~geller/632/ports.html*

**b) What is the UNIX command to create a new folder? [1]**

The UNIX command to create new folder or a directory is ‘mkdir’.

**c) What is the vi command to delete a single character?**

The vi command to delete a single character is ‘x’.

**d) What is the AQUA port number we use to connect to prophet? [1]**

The port number was 1521 the default that is the on the system.

|  |  |
| --- | --- |
| Username | NM489 (UCID) |
| Host | prophet.njit.edu |
| SID | course |

**e) When is the first Midterm exam? [1]**

Midterm 1 is due on October 24, 2017.