1-a) Go to: https://ephtracking.cdc.gov/

Click on Explore Data

STEP 1: CONTENT
Click on "Select Content Area" so that you get "Chronic Obstructive Pulmonary Disease (COPD)".
Choose "Mortality from COPD" on the second drop-down menu.
Choose "Crude Death Rate from COPD among people >= 25 years of age per 100,000 population" on the third drop-down menu.

STEP 2: GEOGRAPHY TYPE
National by State

STEP 3: GEOGRAPHY
All States.

STEP 4: TIME
Choose all these 4 years:
2017, 2018, 2019, 2020

STEP 5: ADVANCED OPTIONS
Race Ethnicity

All 4 Choices for Race

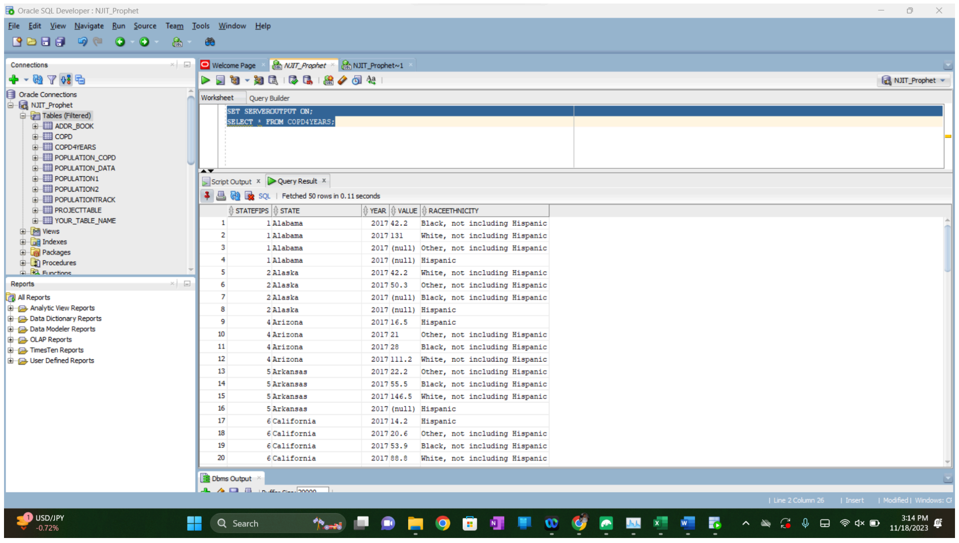
Then download the data and save it as a COPD4YEARS.CSV file [1 point].

Replace all "Suppressed" values in the VALUE column to NULL.

b) Load the Cleaned Data into an Oracle Table COPD4YEARS using SQL Developer.
Write an SQL statement to display the COPD4YEARS table.

►
```
SELECT * FROM COPD4YEARS
```

After the first red arrow below, show a screen dump that shows the select statement and at least the first 20 lines for the table. [2]



```
SET SERVEROUTPUT ON;
SELECT * FROM COPD4YEARS;
```

Fetched 50 rows in 0.11 seconds

	STATEFIPS	STATE	YEAR	VALUE	RACEETHNICITY
1	1	Alabama	2017	42.2	Black, not including Hispanic
2	1	Alabama	2017	131	White, not including Hispanic
3	1	Alabama	2017	(null)	Other, not including Hispanic
4	1	Alabama	2017	(null)	Hispanic
5	2	Alaska	2017	42.2	White, not including Hispanic
6	2	Alaska	2017	50.3	Other, not including Hispanic
7	2	Alaska	2017	(null)	Black, not including Hispanic
8	2	Alaska	2017	(null)	Hispanic
9	4	Arizona	2017	16.5	Hispanic
10	4	Arizona	2017	21	Other, not including Hispanic
11	4	Arizona	2017	28	Black, not including Hispanic
12	4	Arizona	2017	111.2	White, not including Hispanic
13	5	Arkansas	2017	22.2	Other, not including Hispanic
14	5	Arkansas	2017	55.5	Black, not including Hispanic
15	5	Arkansas	2017	146.5	White, not including Hispanic
16	5	Arkansas	2017	(null)	Hispanic
17	6	California	2017	14.2	Hispanic
18	6	California	2017	20.6	Other, not including Hispanic
19	6	California	2017	53.9	Black, not including Hispanic
20	6	California	2017	88.8	White, not including Hispanic

▶

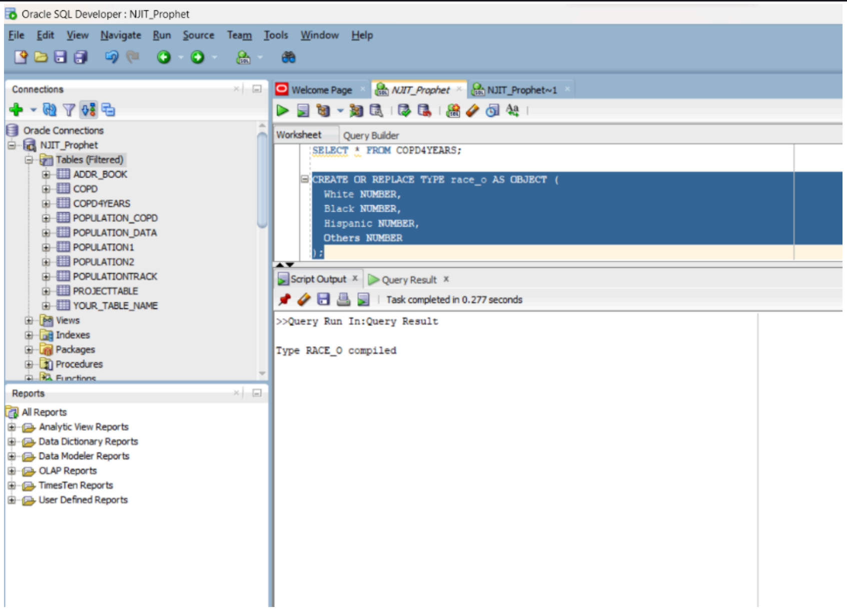
c) Create an Oracle Class **race_o** from the COPD4YEARS table with the following attributes; White, Black, Hispanic, and Others

Show the code for the types after the second red arrow. [1]

▶

```
CREATE OR REPLACE TYPE race_o AS OBJECT (
  White NUMBER,
  Black NUMBER,
  Hispanic NUMBER,
  Others NUMBER
);
```

▶



```
SELECT * FROM COPD4YEARS;

CREATE OR REPLACE TYPE race_o AS OBJECT (
  White NUMBER,
  Black NUMBER,
  Hispanic NUMBER,
  Others NUMBER
);
```

Task completed in 0.277 seconds

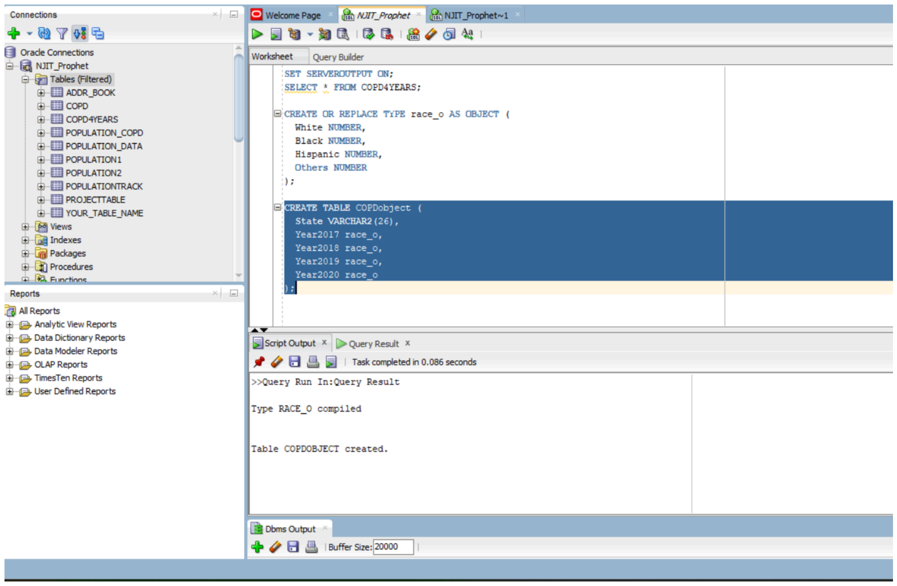
```
>>Query Run In:Query Result

Type RACE_O compiled
```

d) Create a table **COPDobject** with **State** attribute data type of VARCHAR2(26) and four attributes with (data type of race_o) named Year2017, Year2018, year2019, and Year2020 respectively.

Show the code for the table after the third red arrow [1].

▶

```
 CREATE TABLE COPDobject (
  State VARCHAR2(26),
  Year2017 race_o,
  Year2018 race_o,
  Year2019 race_o,
  Year2020 race_o
);
```

▶ 

```
SET SERVEROUTPUT ON;
SELECT * FROM COPD4YEARS;

CREATE OR REPLACE TYPE race_o AS OBJECT (
  White NUMBER,
  Black NUMBER,
  Hispanic NUMBER,
  Others NUMBER
);

CREATE TABLE COPDobject (
  State VARCHAR2(26),
  Year2017 race_o,
  Year2018 race_o,
  Year2019 race_o,
  Year2020 race_o
);
```

Task completed in 0.086 seconds

```
>>Query Run In:Query Result

Type RACE_O compiled


Table COPDOBJECT created.
```

2) a) Write a PL/SQL program using an **implicit** cursor that inserts into a **COPDobject** table all states' names and each year all race data.

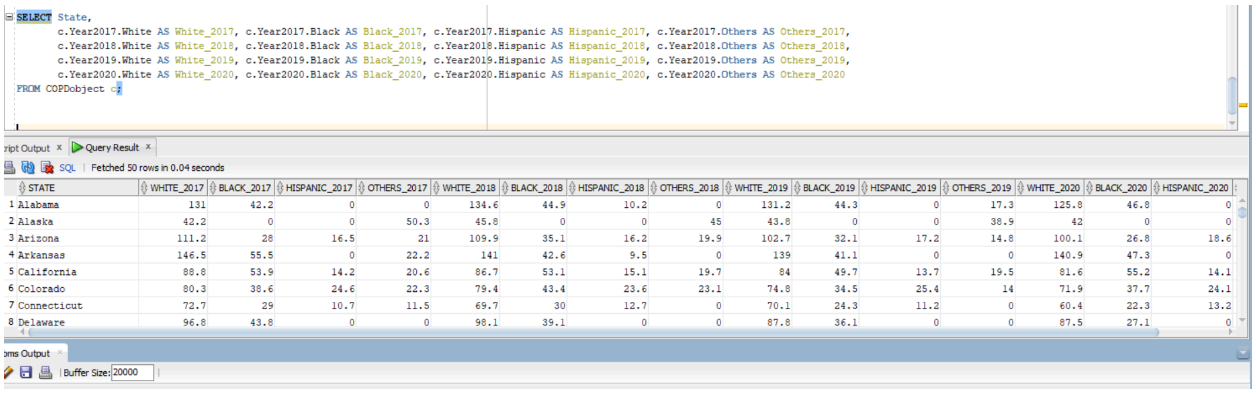
Show the program after the first red arrow.

▶

```
 DECLARE
  v_year_2017 race_o;
  v_year_2018 race_o;
  v_year_2019 race_o;
  v_year_2020 race_o;
  v_current_state VARCHAR2(100);
  v_previous_state VARCHAR2(100) := NULL;

  FUNCTION reset_race_o RETURN race_o IS
  BEGIN
    RETURN race_o(0, 0, 0, 0);
  END reset_race_o;
BEGIN
  FOR rec IN (SELECT State, Year, Value, RACEETHNICITY FROM
COPD4YEARS ORDER BY State, Year, RACEETHNICITY) LOOP
    v_current_state := rec.State;
    IF v_current_state != v_previous_state AND v_previous_state IS
NOT NULL THEN
      INSERT INTO COPDobject (State, Year2017, Year2018, Year2019,
Year2020)
      VALUES (v_previous_state, v_year_2017, v_year_2018,
v_year_2019, v_year_2020);
      v_year_2017 := reset_race_o;
      v_year_2018 := reset_race_o;
      v_year_2019 := reset_race_o;
      v_year_2020 := reset_race_o;
    END IF;
    IF rec.Year = 2017 THEN
      v_year_2017 := race_o(
        CASE WHEN rec.RACEETHNICITY = 'White, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2017.White END,
        CASE WHEN rec.RACEETHNICITY = 'Black, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2017.Black END,
        CASE WHEN rec.RACEETHNICITY = 'Hispanic' THEN NVL(rec.Value,
0) ELSE v_year_2017.Hispanic END,
        CASE WHEN rec.RACEETHNICITY = 'Other, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2017.Others END
      );
    ELSIF rec.Year = 2018 THEN
      v_year_2018 := race_o(
        CASE WHEN rec.RACEETHNICITY = 'White, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2018.White END,
        CASE WHEN rec.RACEETHNICITY = 'Black, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2018.Black END,
        CASE WHEN rec.RACEETHNICITY = 'Hispanic' THEN NVL(rec.Value,
0) ELSE v_year_2018.Hispanic END,
        CASE WHEN rec.RACEETHNICITY = 'Other, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2018.Others END
      );
    ELSIF rec.Year = 2019 THEN
      v_year_2019 := race_o(
        CASE WHEN rec.RACEETHNICITY = 'White, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2019.White END,
        CASE WHEN rec.RACEETHNICITY = 'Black, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2019.Black END,
        CASE WHEN rec.RACEETHNICITY = 'Hispanic' THEN NVL(rec.Value,
0) ELSE v_year_2019.Hispanic END,
        CASE WHEN rec.RACEETHNICITY = 'Other, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2019.Others END
      );
    ELSIF rec.Year = 2020 THEN
      v_year_2020 := race_o(
        CASE WHEN rec.RACEETHNICITY = 'White, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2020.White END,
        CASE WHEN rec.RACEETHNICITY = 'Black, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2020.Black END,
        CASE WHEN rec.RACEETHNICITY = 'Hispanic' THEN NVL(rec.Value,
0) ELSE v_year_2020.Hispanic END,
        CASE WHEN rec.RACEETHNICITY = 'Other, not including Hispanic'
THEN NVL(rec.Value, 0) ELSE v_year_2020.Others END
      );
    END IF;

    v_previous_state := v_current_state;
  END LOOP;
  IF v_previous_state IS NOT NULL THEN
    INSERT INTO COPDobject (State, Year2017, Year2018, Year2019,
Year2020)
    VALUES (v_previous_state, v_year_2017, v_year_2018, v_year_2019,
v_year_2020);
```

```
   END IF;
END;
```
Show the first 10 rows of the result at the second red arrow by snipping them. [10]

► Output after iterating each race_o object:



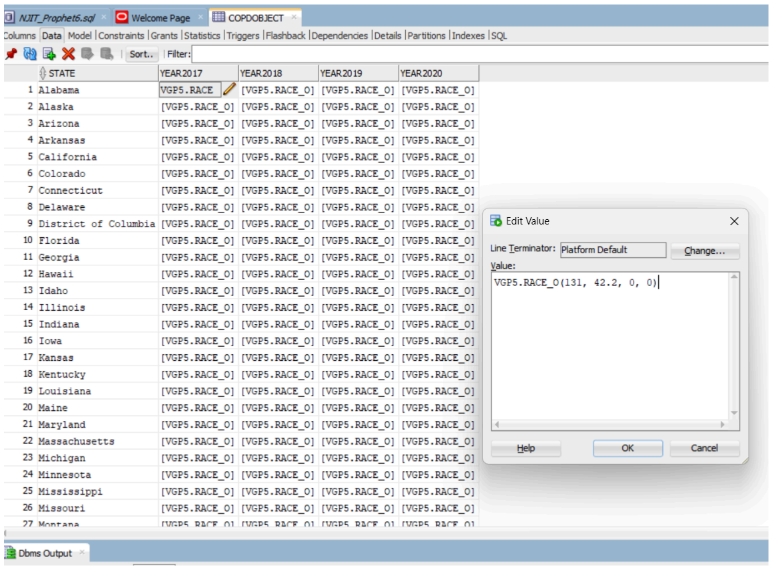
```
SELECT State,
       c.Year2017.White AS White_2017, c.Year2017.Black AS Black_2017, c.Year2017.Hispanic AS Hispanic_2017, c.Year2017.Others AS Others_2017,
       c.Year2018.White AS White_2018, c.Year2018.Black AS Black_2018, c.Year2018.Hispanic AS Hispanic_2018, c.Year2018.Others AS Others_2018,
       c.Year2019.White AS White_2019, c.Year2019.Black AS Black_2019, c.Year2019.Hispanic AS Hispanic_2019, c.Year2019.Others AS Others_2019,
       c.Year2020.White AS White_2020, c.Year2020.Black AS Black_2020, c.Year2020.Hispanic AS Hispanic_2020, c.Year2020.Others AS Others_2020
FROM COPDobject c;
```

Fetched 50 rows in 0.04 seconds

	STATE	WHITE_2017	BLACK_2017	HISPANIC_2017	OTHERS_2017	WHITE_2018	BLACK_2018	HISPANIC_2018	OTHERS_2018	WHITE_2019	BLACK_2019	HISPANIC_2019	OTHERS_2019	WHITE_2020	BLACK_2020	HISPANIC_2020
1	Alabama	131	42.2	0	0	134.6	44.9	10.2	0	131.2	44.3	0	17.3	125.8	46.8	0
2	Alaska	42.2	0	0	50.3	45.8	0	0	45	43.8	0	0	38.9	42	0	0
3	Arizona	111.2	28	16.5	21	109.9	35.1	16.2	19.9	102.7	32.1	17.2	14.8	100.1	26.8	18.6
4	Arkansas	146.5	55.5	0	22.2	141	42.6	9.5	0	139	41.1	0	0	140.9	47.3	0
5	California	88.8	53.9	14.2	20.6	86.7	53.1	15.1	19.7	84	49.7	13.7	19.5	81.6	55.2	14.1
6	Colorado	80.3	38.6	24.6	22.3	79.4	43.4	23.6	23.1	74.8	34.5	25.4	14	71.9	37.7	24.1
7	Connecticut	72.7	29	10.7	11.5	69.7	30	12.7	0	70.1	24.3	11.2	0	60.4	22.3	13.2
8	Delaware	96.8	43.8	0	0	98.1	39.1	0	0	87.8	36.1	0	0	87.5	27.1	0

Buffer Size: 20000

And ouput as table itself (but sqldeveloper didn't let me display race_o object in its format. Rather it used my accountName.race_o as:



	STATE	YEAR2017	YEAR2018	YEAR2019	YEAR2020
1	Alabama	VGP5.RACE	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
2	Alaska	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
3	Arizona	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
4	Arkansas	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
5	California	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
6	Colorado	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
7	Connecticut	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
8	Delaware	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
9	District of Columbia	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
10	Florida	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
11	Georgia	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
12	Hawaii	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
13	Idaho	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
14	Illinois	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
15	Indiana	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
16	Iowa	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
17	Kansas	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
18	Kentucky	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
19	Louisiana	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
20	Maine	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
21	Maryland	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
22	Massachusetts	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
23	Michigan	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
24	Minnesota	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
25	Mississippi	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
26	Missouri	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]
27	Montana	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]	[VGP5.RACE_O]

Edit Value

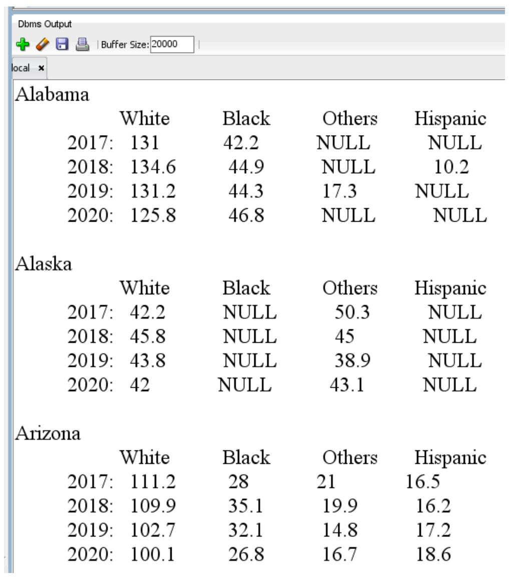
Line Terminator: Platform Default

Value:

```
VGP5.RACE_O(131, 42.2, 0, 0)
```

b) Write a PL/SQL program using an **implicit** cursor that displays information about all states in the table **COPDobject** in a nice format. Use rpad().

Your output should look like this:


Dbms Output

Buffer Size: 20000

local

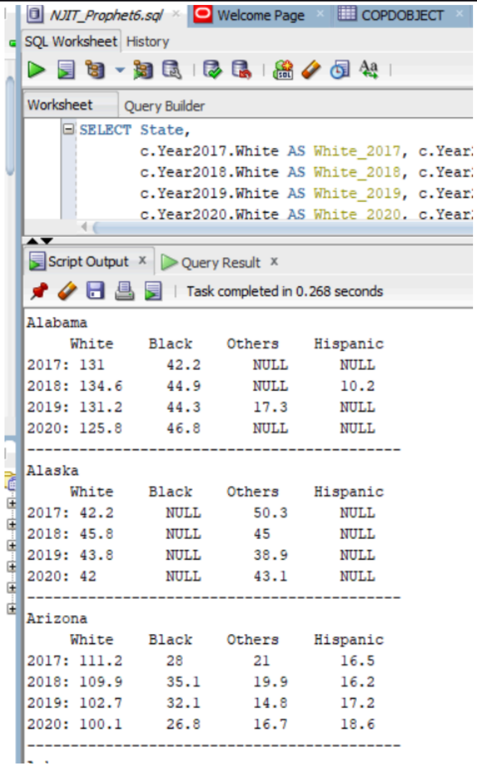
```
Alabama
          White      Black      Others     Hispanic
   2017:  131        42.2       NULL       NULL
   2018:  134.6      44.9       NULL       10.2
   2019:  131.2      44.3       17.3       NULL
   2020:  125.8      46.8       NULL       NULL

Alaska
          White      Black      Others     Hispanic
   2017:  42.2       NULL       50.3       NULL
   2018:  45.8       NULL       45         NULL
   2019:  43.8       NULL       38.9       NULL
   2020:  42         NULL       43.1       NULL

Arizona
          White      Black      Others     Hispanic
   2017:  111.2      28         21         16.5
   2018:  109.9      35.1       19.9       16.2
   2019:  102.7      32.1       14.8       17.2
   2020:  100.1      26.8       16.7       18.6
```

Show the program at the third red arrow.

Show the output of the first three state data **COPDobject** table at the fourth red arrow, the last three state data at the fifth red arrow by snipping them [10].

►

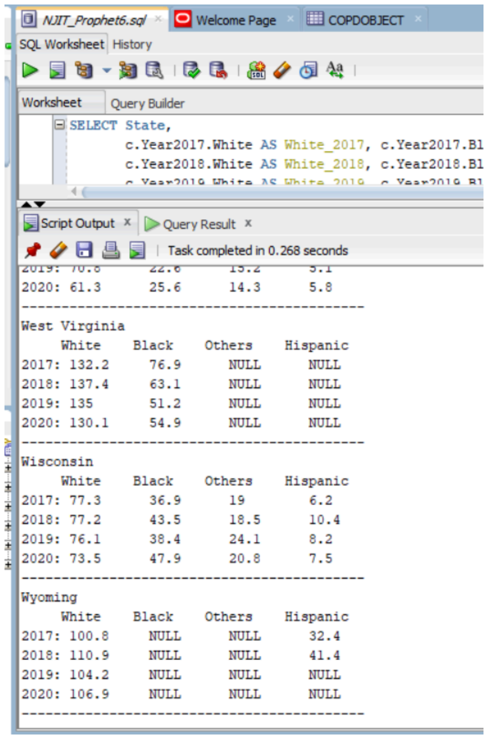
```
 DECLARE
    CURSOR copd_cursor IS
        SELECT State,
               Year2017, Year2018, Year2019, Year2020
        FROM COPDobject;
    rec copd_cursor%ROWTYPE;
    FUNCTION zero_to_null(p_value IN NUMBER) RETURN VARCHAR2 IS
    BEGIN
        IF p_value = 0 THEN
            RETURN 'NULL';
        ELSE
            RETURN TO_CHAR(p_value);
        END IF;
    END zero_to_null;
BEGIN
    FOR rec IN copd_cursor LOOP
        DBMS_OUTPUT.PUT_LINE(rec.State);
        DBMS_OUTPUT.PUT_LINE(RPAD(' ', 5) || 'White' || RPAD(' ', 4) || 'Black' || RPAD(' ', 4) ||
'Others' || RPAD(' ', 4) || 'Hispanic');
        DBMS_OUTPUT.PUT_LINE('2017: ' || RPAD(zero_to_null(rec.Year2017.White), 10) ||
RPAD(zero_to_null(rec.Year2017.Black), 10) || RPAD(zero_to_null(rec.Year2017.Others), 10) ||
RPAD(zero_to_null(rec.Year2017.Hispanic), 10));
        DBMS_OUTPUT.PUT_LINE('2018: ' || RPAD(zero_to_null(rec.Year2018.White), 10) ||
RPAD(zero_to_null(rec.Year2018.Black), 10) || RPAD(zero_to_null(rec.Year2018.Others), 10) ||
RPAD(zero_to_null(rec.Year2018.Hispanic), 10));
        DBMS_OUTPUT.PUT_LINE('2019: ' || RPAD(zero_to_null(rec.Year2019.White), 10) ||
RPAD(zero_to_null(rec.Year2019.Black), 10) || RPAD(zero_to_null(rec.Year2019.Others), 10) ||
RPAD(zero_to_null(rec.Year2019.Hispanic), 10));
        DBMS_OUTPUT.PUT_LINE('2020: ' || RPAD(zero_to_null(rec.Year2020.White), 10) ||
RPAD(zero_to_null(rec.Year2020.Black), 10) || RPAD(zero_to_null(rec.Year2020.Others), 10) ||
RPAD(zero_to_null(rec.Year2020.Hispanic), 10));
        DBMS_OUTPUT.PUT_LINE('-------------------------------------------');
    END LOOP;
END;
SET SERVEROUTPUT ON;
```



```
SELECT State,
        c.Year2017.White AS White_2017, c.Year
        c.Year2018.White AS White_2018, c.Year
        c.Year2019.White AS White_2019, c.Year
        c.Year2020.White AS White_2020, c.Year
```

Task completed in 0.268 seconds

```
Alabama
     White    Black    Others    Hispanic
2017: 131      42.2      NULL      NULL
2018: 134.6    44.9      NULL      10.2
2019: 131.2    44.3      17.3      NULL
2020: 125.8    46.8      NULL      NULL
-------------------------------------------
Alaska
     White    Black    Others    Hispanic
2017: 42.2     NULL      50.3      NULL
2018: 45.8     NULL      45        NULL
2019: 43.8     NULL      38.9      NULL
2020: 42       NULL      43.1      NULL
-------------------------------------------
Arizona
     White    Black    Others    Hispanic
2017: 111.2    28        21        16.5
2018: 109.9    35.1      19.9      16.2
2019: 102.7    32.1      14.8      17.2
2020: 100.1    26.8      16.7      18.6
-------------------------------------------
```



```
SELECT State,
        c.Year2017.White AS White_2017, c.Year2017.Bl
        c.Year2018.White AS White_2018, c.Year2018.Bl
```

Task completed in 0.268 seconds

```
2020: 61.3      25.6      14.3      5.8
-----------------------------------------
West Virginia
     White     Black     Others    Hispanic
2017: 132.2     76.9      NULL      NULL
2018: 137.4     63.1      NULL      NULL
2019: 135       51.2      NULL      NULL
2020: 130.1     54.9      NULL      NULL
-----------------------------------------
Wisconsin
     White     Black     Others    Hispanic
2017: 77.3      36.9      19        6.2
2018: 77.2      43.5      18.5      10.4
2019: 76.1      38.4      24.1      8.2
2020: 73.5      47.9      20.8      7.5
-----------------------------------------
Wyoming
     White     Black     Others    Hispanic
2017: 100.8     NULL      NULL      32.4
2018: 110.9     NULL      NULL      41.4
2019: 104.2     NULL      NULL      NULL
2020: 106.9     NULL      NULL      NULL
-----------------------------------------
```

▶

3- Write a PL/SQL program using an **implicit cursor** that loads all the data from the POPULATION2 (from Homework1) table into the POPULATIONXML table.

To clarify: POPULATIONXML has to have all rows with the following information from POPULATION2, but in a single column, as XML expressions.

STATE, WHITE, AFRICANAMERICAN, MULTIRACIAL, OTHER for
WHITE>1,000,000 and AFRICANAMERICAN >10,000 and MULTIRACIAL>50,000

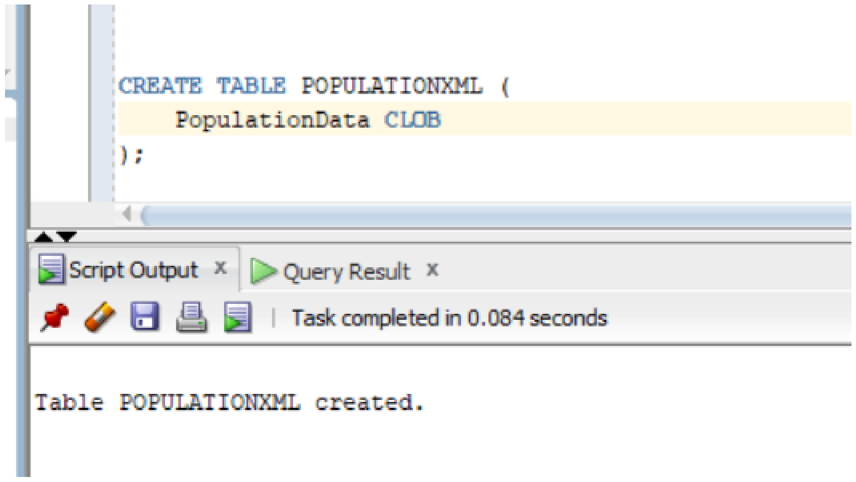
HINTS: Students usually have a really hard time with this.

You need to construct the XML expression from strings with lots of concatenation (||) operators.

'<population>
<STATE>' || variablefromcursorwithSTATEinit || '</STATE>
<WHITE>' || variablefromcursorwithWHITEinit || </WHITE>'
Etc. etc.

Show the POPULATIONXML table creation statement at the first red arrow below [1 point].

Show the INDENTED program to insert data with cursor at the second red arrow below [6].



```
CREATE TABLE POPULATIONXML (
    PopulationData CLOB
);
```

Script Output × Query Result ×

Task completed in 0.084 seconds

```
Table POPULATIONXML created.
```

▶

▶

```
 DECLARE
    CURSOR pop_cursor IS
        SELECT STATE,
               WHITE,
               BLACK_OR_AFRICAN_AMERICAN,
               MIXED_RACE_MULTI_RACIAL,
               OTHER2
        FROM POPULATION2
        WHERE WHITE > 1000000
          AND BLACK_OR_AFRICAN_AMERICAN > 10000
          AND MIXED_RACE_MULTI_RACIAL > 50000;

    v_xml_expression VARCHAR2(4000);

BEGIN

    FOR rec IN pop_cursor LOOP

        v_xml_expression :=
            '<population>' ||
            '<STATE>' || rec.STATE || '</STATE>' ||
            '<WHITE>' || rec.WHITE || '</WHITE>' ||
            '<AFRICANAMERICAN>' || rec.BLACK_OR_AFRICAN_AMERICAN || '</AFRICANAMERICAN>' ||
            '<MULTIRACIAL>' || rec.MIXED_RACE_MULTI_RACIAL || '</MULTIRACIAL>' ||
            '<OTHER>' || rec.OTHER2 || '</OTHER>' ||
            '</population>';

        INSERT INTO POPULATIONXML (PopulationData)
        VALUES (v_xml_expression);
    END LOOP;
END;
```

4- Write a PL/SQL program that uses ONLY the table POPULATIONXML and does NOT use the table POPULATION2 or any other table or any other view. Zero points if you do not follow these rules.

This program should send the complete table POPULATIONXML to the screen in the following format:

```
STATE : Alabama
White            : 3171351
African American : 1288159
Multiracial      : 184618
Others           : 116104

STATE : Arizona
White            : 3816547
African American : 317161
Multiracial      : 266840
Others           : 558701

STATE : Arkansas
White            : 2063550
African American : 449884
Multiracial      : 147157
Others           : 94086

STATE : California
Etc. etc.
```

Note the empty line. There has to be an empty line after every new table row. Every table row becomes 5 lines + 1 empty line after.

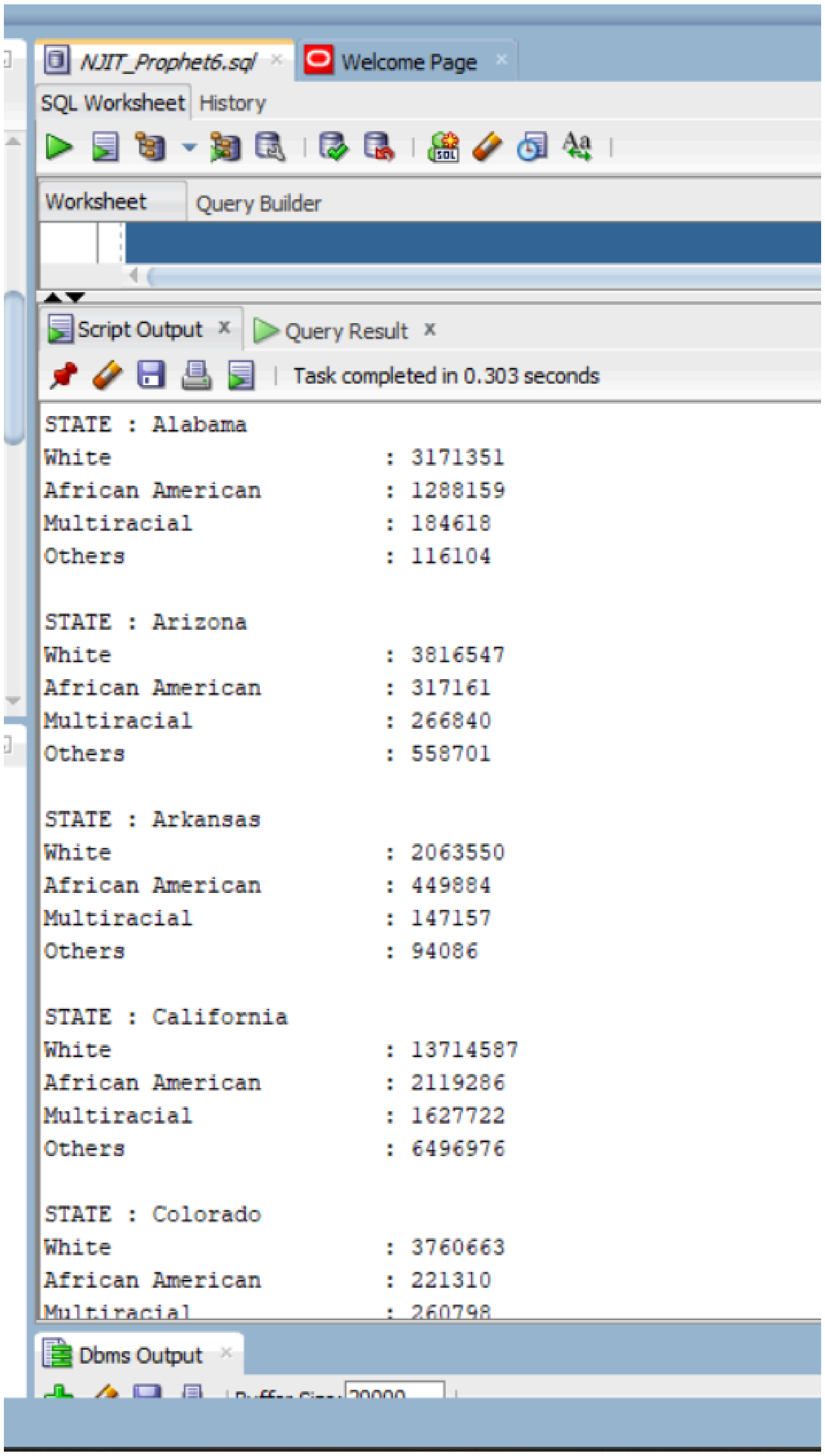
Note that all the ":" are lined up in the same column. It has to be that way.

Show the program at the first red arrow below.

Show the first 12 lines of the result at the second red arrow below [6 points].

►

```
DECLARE

    CURSOR xml_cursor IS
        SELECT PopulationData
        FROM POPULATIONXML;
    v_xml CLOB;

    v_state VARCHAR2(100);
    v_white NUMBER;
    v_african_american NUMBER;
    v_multiracial NUMBER;
    v_others NUMBER;
BEGIN

    FOR rec IN xml_cursor LOOP
        v_xml := rec.PopulationData;


        SELECT EXTRACTVALUE(XMLTYPE(v_xml), '/population/STATE') INTO v_state FROM
DUAL;
        SELECT EXTRACTVALUE(XMLTYPE(v_xml), '/population/WHITE') INTO v_white FROM
DUAL;
        SELECT EXTRACTVALUE(XMLTYPE(v_xml), '/population/AFRICANAMERICAN') INTO
v_african_american FROM DUAL;
        SELECT EXTRACTVALUE(XMLTYPE(v_xml), '/population/MULTIRACIAL') INTO
v_multiracial FROM DUAL;
        SELECT EXTRACTVALUE(XMLTYPE(v_xml), '/population/OTHER') INTO v_others FROM
DUAL;


        DBMS_OUTPUT.PUT_LINE('STATE : ' || RPAD(v_state, 20));
        DBMS_OUTPUT.PUT_LINE(RPAD('White', 25) || ': ' || v_white);
        DBMS_OUTPUT.PUT_LINE(RPAD('African American', 25) || ': ' ||
v_african_american);
        DBMS_OUTPUT.PUT_LINE(RPAD('Multiracial', 25) || ': ' || v_multiracial);
        DBMS_OUTPUT.PUT_LINE(RPAD('Others', 25) || ': ' || v_others);
        DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
END;
```



Task completed in 0.303 seconds

```
STATE : Alabama
White                        : 3171351
African American             : 1288159
Multiracial                  : 184618
Others                       : 116104

STATE : Arizona
White                        : 3816547
African American             : 317161
Multiracial                  : 266840
Others                       : 558701

STATE : Arkansas
White                        : 2063550
African American             : 449884
Multiracial                  : 147157
Others                       : 94086

STATE : California
White                        : 13714587
African American             : 2119286
Multiracial                  : 1627722
Others                       : 6496976

STATE : Colorado
White                        : 3760663
African American             : 221310
Multiracial                  : 260798
```

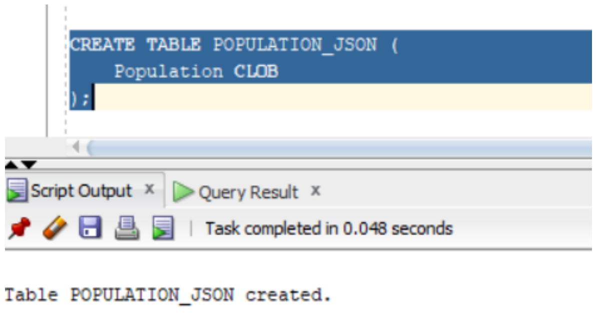
Buffer Size: 20000

5- Create a table POPULATION_JSON in Oracle. It should contain a single column POPULATION that contains STATE, WHITE, AFRICANAMERICAN, MULTIRACIAL, OTHER attributes of POPULATION2.

Then, write a PL/SQL program using an **implicit cursor** that loads all the data from the POPULATION2 table into the POPULATION_JSON.

Show the POPULATION_JSON table creation statement at the first red arrow below [1 point].

Show the INDENTED program to insert data with cursor at the second red arrow below [5].



```
CREATE TABLE POPULATION_JSON (
    Population CLOB
);
```

```
Table POPULATION_JSON created.
```

▶

▶

```
DECLARE

    CURSOR pop_cursor IS
        SELECT STATE,
               WHITE,
               BLACK_OR_AFRICAN_AMERICAN,
               MIXED_RACE_MULTI_RACIAL,
               OTHER2
        FROM POPULATION2;


    v_json CLOB;
BEGIN

    FOR rec IN pop_cursor LOOP

        v_json :=
            '{' ||
            '"state": "' || rec.STATE || '",' ||
            '"white": ' || rec.WHITE || ',' ||
            '"africanAmerican": ' || rec.BLACK_OR_AFRICAN_AMERICAN || ','
||
            '"multiracial": ' || rec.MIXED_RACE_MULTI_RACIAL || ',' ||
            '"other": ' || rec.OTHER2 ||
            '}';


        INSERT INTO POPULATION_JSON (Population)
        VALUES (v_json);
    END LOOP;
END;
```

6- Write a program **using a cursor** that displays the complete table POPULATION_JSON on the screen in the following format. NOTE THE NEWLINES.

```
STATE : Alabama
White               : 3171351
African American : 1288159
Multiracial         : 184618
Others               : 116104

STATE : Arizona
White               : 3816547
African American : 317161
Multiracial         : 266840
Others               : 558701

 Etc…etc….
```

Every table row (i.e., JSON expression) becomes five rows followed by an empty row.

Show the program at the first red arrow below.

Show the first 12 lines of the result at the second red arrow below [6 points]

►

```
DECLARE

   CURSOR json_cursor IS
       SELECT Population
```

```
        FROM POPULATION_JSON;


    v_state VARCHAR2(100);
    v_white VARCHAR2(100);
    v_african_american VARCHAR2(100);
    v_multiracial VARCHAR2(100);
    v_others VARCHAR2(100);


    FUNCTION get_json_value(json_data IN CLOB, key IN VARCHAR2) RETURN
VARCHAR2 IS
        v_start_pos PLS_INTEGER;
        v_end_pos PLS_INTEGER;
    BEGIN
        v_start_pos := INSTR(json_data, '"' || key || '": ') + LENGTH(key)
+ 4;
        IF v_start_pos > LENGTH(key) + 4 THEN
            v_end_pos := INSTR(json_data, ',', v_start_pos);
            IF v_end_pos = 0 THEN
                v_end_pos := INSTR(json_data, '}', v_start_pos);
            END IF;
            RETURN TRIM(BOTH '"' FROM SUBSTR(json_data, v_start_pos,
v_end_pos - v_start_pos));
        ELSE
            RETURN NULL;
        END IF;
    END get_json_value;
BEGIN
    FOR rec IN json_cursor LOOP

        v_state := get_json_value(rec.Population, 'state');
        v_white := get_json_value(rec.Population, 'white');
        v_african_american := get_json_value(rec.Population,
'africanAmerican');
        v_multiracial := get_json_value(rec.Population, 'multiracial');
        v_others := get_json_value(rec.Population, 'other');


        DBMS_OUTPUT.PUT_LINE('STATE : ' || v_state);
        DBMS_OUTPUT.PUT_LINE(RPAD('White', 25) || ': ' || v_white);
        DBMS_OUTPUT.PUT_LINE(RPAD('African American', 25) || ': ' ||
v_african_american);
```

```
        DBMS_OUTPUT.PUT_LINE(RPAD('Multiracial', 25) || ': ' ||
v_multiracial);
        DBMS_OUTPUT.PUT_LINE(RPAD('Others', 25) || ': ' || v_others);
        DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
END;
```

▶