File Edit View History Bookmarks Tools Help

Pay raise examples

www.pgrocer.net/Cis50/payraise.html

PL/SQL CODE:

Think of a table as you would think of an array. The table t_ManagerTable is going to have a group of unique manager_id refered to as v_ManagerIdTable(with an index to tell which one).

```
SET SERVEROUTPUT ON
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        v_ManagerId employeez.manager_id%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;


BEGIN
        OPEN c_Employee1;
        LOOP
          FETCH c_Employee1 INTO v_ManagerId;
          EXIT WHEN c_Employee1%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_ManagerId = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
              EXIT;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = FALSE THEN
            v_ManagerIdTable(v_ManagerIdTable.COUNT + 1) := v_ManagerId;
          END IF;
        END LOOP;
        CLOSE c_Employee1;
        FOR i IN 1..v_ManagerIdTable.COUNT LOOP
           dbms_output.put_line(v_ManagerIdTable(i));
        END LOOP;
END;
/
SET SERVEROUTPUT OFF
```

Style wise it is better to have the EXIT at the bottom just before you
go back and check to see if you should do it again.
The outer LOOP controls moving through the information in the cursor.
The inner loop checks to see if the manager_id has already been placed in the table.
If it has then you set the flag to true so you will not enter a copy of it.
The if after the while checks the flag and only puts the manager_id in the table if the flag is
False indicating a match was not found.

Because I am creating a table which will show no output, I used the dbms line to show the contents of the table after I had completed it. For this reason, I am using SET SERVEROUTPUT ON/OFF.

In the DECLARE, I first declare a table type that will hold the manager_id. Again remember that I want the results to be all manager ids with no duplicates. The type is t_managerTable and it is a table of employeez.manager_id%TYPE meaning the elements in the table will have the same type as the manager id on the employeez table. Then I assign the v_ManagerIdTable which is the name of this type that I am using in my code.

I then set up a cursor called c_employee1 which selects all of the manager ids from the employeez table.

Finally I set up variables to hold the manager_id, a boolean flag to be used to determine if the manager id is already in the table and a binary integer field that will be used in my processing.

In the begin, I first open the c_Employee1 cursor.

I then enter a outer loop, with an EXIT when there is nothing more to fetch from the c_Employee1 cursor. In the loop I will fetch an entry from the cursor into the field called v_ManagerId. I will also initialize the v_FoundFlag to False and the v_LoopCnt to 1.

I then enter an inner WHILE loop that will process while the v_LoopCtr that I initialized at 1 is less then or equal to the COUNT of the elements in the v_ManagerIDTable. Inside the loop, I am checking the v_managerId that I fetched against the things that are already in the v_ManagerIdTable one by one. If I find a match I am setting the v_FoundFlag to true so I will know the manager id is already in the table and exiting the loop. Notice that I can have an EXIT in a while loop to let me exit the loop before

Type here to search

9:41 AM
11/5/2019

The purpose of this program is to create a table containing the each unique manager_id. I will then use this in subsequent programs to calculate the salary.

H3>SQL CODE:

```
SQL> select * from employeez;

EMPLOYEE_ID LAST_NAME       FIRST_NAME      MANAGER_ID    SALARY
----------- --------------- --------------- ---------- ---------
       7369 SMITH           JOHN                  7902       800
       7499 ALLEN           KEVIN                 7698      1600
       7505 DOYLE           JEAN                  7839      2850
       7506 DENNIS          LYNN                  7839      2750
       7507 BAKER           LESLIE                7839      2200
       7521 WARD            CYNTHIA               7698      1250
       7555 PETERS          DANIEL                7505      1250
       7557 SHAW            KAREN                 7505      1250
       7560 DUNCAN          SARAH                 7506      1250
       7564 LANGE           GREGORY               7506      1250
       7566 JONES           TERRY                 7839      2975
       7569 ALBERTS         CHRIS                 7839      3000
       7600 PORTER          RAYMOND               7505      1250
       7609 LEWIS           RICHARD               7507      1800
       7654 MARTIN          KENNETH               7698      1250
       7676 SOMMERS         DENISE                7507      1850
       7698 BLAKE           MARION                7839      2850
       7782 CLARK           CAROL                 7839      2450
       7788 SCOTT           DONALD                7566      3000
       7789 WEST            LIVIA                 7506      1500
       7799 FISHER          MATTHEW               7569      3000

EMPLOYEE_ID LAST_NAME       FIRST_NAME      MANAGER_ID    SALARY
----------- --------------- --------------- ---------- ---------
       7820 ROSS            PAUL                  7505      1300
       7839 KING            FRANCIS                         5000
       7844 TURNER          MARY                  7698      1500
       7876 ADAMS           DIANE                 7788      1100
       7900 JAMES           FRED                  7698       950
       7902 FORD            JENNIFER              7566      3000
       7916 ROBERTS         GRACE                 7569      2875
       7919 DOUGLAS         MICHAEL               7799       800
       7934 MILLER          BARBARA               7782      1300
       7950 JENSEN          ALICE                 7505       750
       7954 MURRAY          JAMES                 7506       750

32 rows selected.
```

PL/SQL CODE:

```
SET SERVEROUTPUT ON
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        v_ManagerId employeez.manager_id%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;
```

File Edit View History Bookmarks Tools Help

Pay raise examples

www.pgrocer.net/Cis50/payraise.html

over to SCOTT ownership using the technique explained in the copy file to another owner handout.

The purpose of this program is to create a table containing the each unique manager_id. I will then use this in subsequent programs to calculate the salary.

H3>SQL CODE:

```
SQL> select * from employeez;

EMPLOYEE_ID LAST_NAME       FIRST_NAME      MANAGER_ID    SALARY
----------- --------------- --------------- ---------- ---------
       7369 SMITH           JOHN                  7902       800
       7499 ALLEN           KEVIN                 7698      1600
       7505 DOYLE           JEAN                  7839      2850
       7506 DENNIS          LYNN                  7839      2750
       7507 BAKER           LESLIE                7839      2200
       7521 WARD            CYNTHIA               7698      1250
       7555 PETERS          DANIEL                7505      1250
       7557 SHAW            KAREN                 7505      1250
       7560 DUNCAN          SARAH                 7506      1250
       7564 LANGE           GREGORY               7506      1250
       7566 JONES           TERRY                 7839      2975
       7569 ALBERTS         CHRIS                 7839      3000
       7600 PORTER          RAYMOND               7505      1250
       7609 LEWIS           RICHARD               7507      1800
       7654 MARTIN          KENNETH               7698      1250
       7676 SOMMERS         DENISE                7507      1850
       7698 BLAKE           MARION                7839      2850
       7782 CLARK           CAROL                 7839      2450
       7788 SCOTT           DONALD                7566      3000
       7789 WEST            LIVIA                 7506      1500
       7799 FISHER          MATTHEW               7569      3000

EMPLOYEE_ID LAST_NAME       FIRST_NAME      MANAGER_ID    SALARY
----------- --------------- --------------- ---------- ---------
       7820 ROSS            PAUL                  7505      1300
       7839 KING            FRANCIS                         5000
       7844 TURNER          MARY                  7698      1500
       7876 ADAMS           DIANE                 7788      1100
       7900 JAMES           FRED                  7698       950
       7902 FORD            JENNIFER              7566      3000
       7916 ROBERTS         GRACE                 7569      2875
       7919 DOUGLAS         MICHAEL               7799       800
       7934 MILLER          BARBARA               7782      1300
       7950 JENSEN          ALICE                 7505       750
       7954 MURRAY          JAMES                 7506       750

32 rows selected.
```

in

table

7902
7698
7839

7839 is already in the table so
I do not want to add it again

PL/SQL CODE:

```
SET SERVEROUTPUT ON
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        v_ManagerId employeez.manager_id%TYPE;
        v_FoundFlag BOOLEAN;
```

Type here to search

9:55 AM
11/5/2019

File Edit View History Bookmarks Tools Help

Pay raise examples

www.pgrocer.net/Cis50/payraise.html

```
SET SERVEROUTPUT ON
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        v_ManagerId employeez.manager_id%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;


BEGIN
        OPEN c_Employee1;
        LOOP
          FETCH c_Employee1 INTO v_ManagerId;
          EXIT WHEN c_Employee1%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_ManagerId = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
              EXIT;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = FALSE THEN
            v_ManagerIdTable(v_ManagerIdTable.COUNT + 1) := v_ManagerId;
          END IF;
        END LOOP;
        CLOSE c_Employee1;
        FOR i IN 1..v_ManagerIdTable.COUNT LOOP
           dbms_output.put_line(v_ManagerIdTable(i));
        END LOOP;
END;
/
SET SERVEROUTPUT OFF
```

Stepping thru cursor

checking for match

Because I am creating a table which will show no output, I used the dbms line to show the contents of the table after I had completed it. For this reason, I am using SET SERVEROUTPUT ON/OFF.

In the DECLARE, I first declare a table type that will hold the manager_id. Again remember that I want the results to be all manager ids with no duplicates. The type is t_managerTable and it is a table of employeez.manager_id%TYPE meaning the elements in the table will have the same type as the manager id on the employeez table. Then I assign the v_ManagerIdTable which is the name of this type that I am using in my code.

I then set up a cursor called c_employee1 which selects all of the manager ids from the employeez table.

Finally I set up variables to hold the manager_id, a boolean flag to be used to determine if the manager id is already in the table and a binary integer field that will be used in my processing.

In the begin, I first open the c_Employee1 cursor.

I then enter a outer loop, with an EXIT when there is nothing more to fetch from the c_Employee1 cursor. In the loop I will fetch an entry from the cursor into the field called v_ManagerId. I will also initialize the v_FoundFlag to False and the v_LoopCnt to 1.

I then enter an inner WHILE loop that will process while the v_LoopCtr that I initialized at 1 is less then or equal to the COUNT of the elements in the v_ManagerIDTable. Inside the loop, I am checking the v_managerId that I fetched against the things that are already in the v_ManagerIdTable one by one. If I find a match I am setting the v_FoundFlag to true so I will know the manager id is already in the table and exiting the loop. Notice that I can have an EXIT in a while loop to let me exit the loop before the condition in the while loop has een met. If I do not exit the table, after the IF, I increment the v_LoopCnt by 1. The inner loop has now ended.

In summary, the inner loop is checking the manager id from the fetch against each element in the manager table. When I have either found a match I will leave the loop because of the EXIT, if I have checked against all of the elements in the table and not

Type here to search

10:05 AM 11/5/2019

After the if, the END LOOP sends me back to the outer loop where I will fetch another manager id from the selection in the cursor.

When the outer loop is complete, there are no more manager ids in the cursor, the cursor is closed.

To show the results, I now have a loop that goes through the table and prints each element. This is done with a FOR loop that moves from 1 to the count of the number of elements in the table. The output that is produced from the dbms line is shown below.

SQL CODE:

```
SQL> @ pay_raise_1a
7902
7698
7839
7505
7506
7507
7566
7569
7788
7799
7782

PL/SQL procedure successfully completed.
```

dbms output from the for loop

Second version of the program - this one actually gives pay raises:

This version of the program updates the file to give managers a 10% raise and non managers a 2% raise. The found flag is used to make this determination.

PL/SQL CODE:

```
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        CURSOR c_Employee2 IS
          SELECT employee_id, salary FROM employeez
          FOR UPDATE OF salary;
        v_ManagerId employeez.manager_id%TYPE;
        v_EmployeeId employeez.employee_id%TYPE;
        v_Salary employeez.salary%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;
        v_TableIndx BINARY_INTEGER;

BEGIN
        OPEN c_Employee1;
        LOOP
          FETCH c_Employee1 INTO v_ManagerId;
          EXIT WHEN c_Employee1%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_ManagerId = v_ManagerIdTable(v_LoopCnt) THEN
```

File Edit View History Bookmarks Tools Help

Pay raise examples

www.pgrocer.net/Cis50/payraise.html

```
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        CURSOR c_Employee2 IS
          SELECT employee_id, salary FROM employeez
          FOR UPDATE OF salary;
        v_ManagerId employeez.manager_id%TYPE;
        v_EmployeeId employeez.employee_id%TYPE;
        v_Salary employeez.salary%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;
        v_TableIndx BINARY_INTEGER;

BEGIN
        OPEN c_Employee1;
        LOOP
          FETCH c_Employee1 INTO v_ManagerId;
          EXIT WHEN c_Employee1%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_ManagerId = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
              EXIT;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = FALSE THEN
            v_ManagerIdTable(v_ManagerIdTable.COUNT + 1) := v_ManagerId;
          END IF;
        END LOOP;
        CLOSE c_Employee1;
        OPEN c_Employee2;
        LOOP
          FETCH c_Employee2 INTO v_EmployeeId, v_Salary;
          EXIT WHEN c_Employee2%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_EmployeeId = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
              EXIT;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = TRUE THEN
            v_Salary := 1.1 * v_Salary;
          ELSE
            v_Salary := 1.02 * v_Salary;
          END IF;
          UPDATE employeez
            SET salary = v_Salary
            WHERE CURRENT OF c_Employee2;
        END LOOP;
        CLOSE c_Employee2;
        COMMIT;
END;
/
```

Same as first

Type here to search

10:07 AM 11/5/2019

File Edit View History Bookmarks Tools Help

Pay raise examples

www.pgrocer.net/Cis50/payraise.html

```
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;
        v_ManagerIdTable t_ManagerTable;
        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;
        CURSOR c_Employee2 IS
          SELECT employee_id, salary FROM employeez
          FOR UPDATE OF salary;
        v_ManagerId employeez.manager_id%TYPE;
        v_EmployeeId employeez.employee_id%TYPE;
        v_Salary employeez.salary%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;
        v_TableIndx BINARY_INTEGER;

BEGIN
        OPEN c_Employee1;
        LOOP
          FETCH c_Employee1 INTO v_ManagerId;
          EXIT WHEN c_Employee1%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_ManagerId = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
              EXIT;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = FALSE THEN
            v_ManagerIdTable(v_ManagerIdTable.COUNT + 1) := v_ManagerId;
          END IF;
        END LOOP;
        CLOSE c_Employee1;
        OPEN c_Employee2;
        LOOP
          FETCH c_Employee2 INTO v_EmployeeId, v_Salary;
          EXIT WHEN c_Employee2%NOTFOUND;
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE v_LoopCnt <= v_ManagerIdTable.COUNT LOOP
            IF v_EmployeeId = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
              EXIT;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = TRUE THEN
            v_Salary := 1.1 * v_Salary;
          ELSE
            v_Salary := 1.02 * v_Salary;
          END IF;
          UPDATE employeez
            SET salary = v_Salary
            WHERE CURRENT OF c_Employee2;
        END LOOP;
        CLOSE c_Employee2;
        COMMIT;
END;
/
```

There was a lot of discussion and explanation so please check the audio.

Type here to search

10:12 AM 11/5/2019

Title: Nov 5-10:12 AM (7 of 15)

File Edit View History Bookmarks Tools Help

CIS150/50 Course Page

www.pgrocer.net/PFGCIS150.html

Oracle and SQL
CIS150/50

About This Course:
- Need help or support?
- Introduction to CIS150
- Syllabus
- Withdrawal Policy as of Fall 2017

Site Resources:
- Notes and Handouts
- Presentations
- SmartBoard Presentations
- Audio

Weekly Schedule:
- Weekly schedule - CIS150/50

Using Oracle:
- Installation and login

Links:
- Links & Tutorials

Jayson Rios from the tutoring center is also available on Monday 11-3 and Wednesday 11-1 in Fall River and Tuesday 11-2 in New Bedford.

Week #10
Week of November 4th

A group of alum are coming to my CIS120 classes on November 5th to talk about jobs and careers in IT. Please try to come. Note they will come to both the 11:00 class and the 12:30 class.
Note: The last date for student generated withdrawal is Thursday, November 14, 2019. Remember I cannot withdraw you so if you do not have the grades needed to pass the course, I will have to give you an F.
Continuing with cursor discussion we started last week.
We will continue with cursors and then back up to see some the the things under Notes and handouts that we did not cover yet.
Continuing example 1 using cursors
Another set of cursor examples
Covers: Another set of cursor exampes
Separate speaker notes for Another set of cursor examples
Examples we are going over in class:
payprog.sql

Week #9
Week of (

/SQL.

Week #8
Week of (

h Joe or myself to answer

ut/output procedures (we will go
ll look at next week.

payprog - Notepad

File Edit Format View Help

cursor joining two tables

```
set serveroutput on
declare
  v_name varchar2(25);
  v_hours number(4);
  cursor payinfo is
      select paydata1.name, paytran1.hourswk
          from paydata1, paytran1
          where paydata1.idno = paytran1.idno;
  begin
     open payinfo;
     fetch payinfo into v_name, v_hours;
     while payinfo%found loop
         dbms_output.put_line(v_name || ' ' || v_hours);
         fetch payinfo into v_name, v_hours;
     end loop;
     close payinfo;
  end;
/
set serveroutput off
```

Type here to search

10:15 AM
11/5/2019

Title: Nov 5-10:15 AM (8 of 15)

File Edit View History Bookmarks Tools Help

CIS150/50 Course Page

www.pgrocer.net/PFGCIS150.html

Oracle and SQL CIS150/50

About This Course:
- Need help or support?
- Introduction to CIS150
- Syllabus
- Withdrawal Policy as of Fall 2017

Site Resources:
- Notes and Handouts
- Presentations
- SmartBoard Presentations
- Audio

Weekly Schedule:
- Weekly schedule - CIS150/50

Using Oracle:
- Installation and login

Links:
- Links & Tutorials

Jayson Rios from the tutoring center is also available on Monday 11-3 and Wednesday 11-1 in Fall River and Tuesday 11-2 in New Bedford.

A group of alum are coming to my CIS120 classes on November 5th to talk about jobs and careers in IT. Please try to come. Note they will come to both the

do not have the grades

Week #10
Week of Novem

payprog1 - Notepad

```
set serveroutput on
declare
  v_idno    paydata1.idno%TYPE;
  v_name    varchar2(25);
  v_hourswk number(4);
  v_tot     number(4);
  cursor payinfo_cursor is
       select idno, name
          from paydata1
          order by idno;
  cursor paytran_cursor is
       select hourswk
          from paytran1
          where v_idno = idno
          order by idno;
  begin
     open payinfo_cursor;
     fetch payinfo_cursor into v_idno, v_name;
     while payinfo_cursor%found loop
        open paytran_cursor;
        v_tot := 0;
        fetch paytran_cursor into v_hourswk;
        while paytran_cursor%found loop
           v_tot := v_tot + v_hourswk;
           dbms_output.put_line(v_name || ' ' || v_tot);
           fetch paytran_cursor into v_hourswk;
        end loop;
        close paytran_cursor;
        fetch payinfo_cursor into v_idno, v_name;
     end loop;
     close payinfo_cursor;
  end;
```

Two file/tables in a 1 to many relationship. There is one record from the payinfo and potentially many records from the paytran.

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Week #9
Week of Octobe

r myself to answer

ut procedures (we will go
t at next week.

Week #8
Week of Octobe

Separate speaker notes for More on PL/SQL IF statements

Type here to search

10:23 AM
11/5/2019

Title: Nov 5-10:23 AM (9 of 15)

File Edit View History Bookmarks Tools Help

CIS150/50 Course Page

www.pgrocer.net/PFGCIS150.html

Oracle and SQL
CIS150/50

About This Course:

- **Need help or support?**
- **Introduction to CIS150**
- **Syllabus**
- **Withdrawal Policy as of Fall 2017**

Site Resources:

- **Notes and Handouts**
- **Presentations**
- **SmartBoard Presentations**
- **Audio**

Weekly Schedule:

- **Weekly schedule - CIS150/50**

Using Oracle:

- **Installation and login**

Links:

- **Links & Tutorials**

	Jayson Rios from the tutoring center is also available on Monday 11-3 and Wednesday 11-1 in Fall River and Tuesday 11-2 in New Bedford.
Week #10 Week of November 4th	A group of alum are coming to my CIS120 classes on November 5th to talk about jobs and careers in IT. Please try to come. Note they will come to both the 11:00 class and the 12:30 class. Note: The last date for student generated withdrawal is Thursday, November 14, 2019. Remember I cannot withdraw you so if you do not have the grades needed to pass the course, I will have to give you an F. Continuing with cursor discussion we started last week. We will continue with cursors and then back up to see some the the things under Notes and handouts that we did not cover yet. **Continuing example 1 using cursors** **Another set of cursor examples** **Covers: Another set of cursor exampes** **Separate speaker notes for Another set of cursor examples** Examples we are going over in class: **payprog.sql** **payprog1.sql** **payprog2.sql** **data.txt** Data layout for studentchk.sql **studentchk.sql** **createdonor.sql** Script to create donor **donorcursor1.sql** **donorcursor2.sql** **donorcursor3.sql** **decodecursor1.sql** Look at these examples
Week #9 Week of October 28th	We will look at the loops and data manipulation in PL/SQL that are listed under last week. Then we will continue on with PL/SQL. Returned to finish this handout: **Introduction to PL/SQL** Note the accompanying Powerpoint is listed on week #8. Then started cursors: **Introduction to Cursors** **Example 1 using cursors** **Covers: Cursors in PL/SQL, cursor example and continuation of first cursor example** **Separate speaker notes for Cursors in PL/SQL, cursor example and continuation of first cursor example** *Assignments:* **Oracle Loop Assignment** **Oracle PL/SQL Intro Quiz** - Start in class
Week #8 Week of October 21st	Thursday is an open lab day from 9:30 until 5:00 in K101. Come for as many hours as you can and work to get caught up with Joe or myself to answer questions. We will start PL/SQL this week. This week, I am going to look at The Introduction to PL/SQL (the beginning part), skip for a while the More examples of input/output procedures (we will go back to them) and work on Introduction to IF statements. The things below will probably be partially done and the rest we will look at next week. **Introduction to PL/SQL** **Introduction to IF statements** **More on PL/SQL IF statements** **Introduction to loops** **Data manipulation in PL/SQL - introduction** With the accompanying PowerPoint presentations: **Introduction to PL/SQL** **Separate speaker notes for Introduction to PL/SQL** **PL/SQL - Using IF statements** **Separate speaker notes for PL/SQL - Using IF statements** **More on PL/SQL IF statements** **Separate speaker notes for More on PL/SQL IF statements**

Type here to search

10:26 AM 11/5/2019

Title: Nov 5-10:26 AM (10 of 15)

donorcursor1 - Notepad

```
SET SERVEROUTPUT ON
DECLARE
  v_state      donor.state%TYPE;
  v_contact    donor.contact%TYPE;
  v_yrgoalsum  donor.yrgoal%TYPE;
CURSOR donorcursor is
SELECT state, contact, SUM(yrgoal)
    FROM donor
    GROUP BY state, contact
    HAVING SUM(yrgoal) >
       (SELECT AVG(yrgoal)
          FROM donor);
BEGIN
    OPEN donorcursor;
    FETCH donorcursor into v_state, v_contact, v_yrgoalsum;
    WHILE donorcursor%FOUND loop
        dbms_output.put_line(v_state || ' ' || v_contact || ' ' || v_yrgoalsum);
        FETCH donorcursor into v_state, v_contact, v_yrgoalsum;
    END loop;
    CLOSE donorcursor;
END;
/
SET SERVEROUTPUT OFF
```

It is good to have the cursor holding specific records you want to handle.

Title: Nov 5-10:29 AM (11 of 15)

File Edit View History Bookmarks Tools Help

CIS150/50 Course Page

www.pgrocer.net/PFGCIS150.html

Oracle and SQL CIS150/50

About This Course:

Need help or support?
Introduction to CIS150
Syllabus
Withdrawal Policy as of Fall 2017

Site Resources:

Notes and Handouts
Presentations
SmartBoard Presentations
Audio

Weekly Schedule:

Weekly schedule - CIS150/50

Using Oracle:

Installation and login

Links:

Links & Tutorials

Jayson Rios from the tutoring center is also available on Monday 11-3 and Wednesday 11-1 in Fall River and Tuesday 11-2 in New Bedford.

donorcursor2 - Notepad

File Edit Format View Help

```
SET SERVEROUTPUT ON
DECLARE
  v_state      donor.state%TYPE;
  v_contact    donor.contact%TYPE;
  v_yrgoalsum  donor.yrgoal%TYPE;
  v_yrgoalavg  donor.yrgoal%TYPE;
CURSOR donorcursor is
SELECT state, contact, SUM(yrgoal)
    FROM donor
    GROUP BY state, contact
    HAVING SUM(yrgoal) >
       (SELECT AVG(yrgoal)
          FROM donor);
BEGIN
    OPEN donorcursor;
    SELECT avg(yrgoal) into v_yrgoalavg from donor;
    dbms_output.put_line(v_yrgoalavg);
    FETCH donorcursor into v_state, v_contact, v_yrgoalsum;
    WHILE donorcursor%FOUND loop
        dbms_output.put_line(v_state || ' ' || v_contact || ' ' || v_yrgoalsum);
        FETCH donorcursor into v_state, v_contact, v_yrgoalsum;
    END loop;
    CLOSE donorcursor;
END;
/
SET SERVEROUTPUT OFF
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8

do not have the grades

or myself to answer

out procedures (we will go
at next week.

Separate speaker notes for More on PL/SQL IF statements

Type here to search

10:30 AM
11/5/2019

Title: Nov 5-10:30 AM (12 of 15)

File Edit View History Bookmarks Tools Help

CIS150/50 Course Page

www.pgrocer.net/PFGCIS150.html

Oracle and SQL CIS150/50

About This Course:
- Need help or support?
- Introduction to CIS150
- Syllabus
- Withdrawal Policy as of Fall 2017

Site Resources:
- Notes and Handouts
- Presentations
- SmartBoard Presentations
- Audio

Weekly Schedule:
- Weekly schedule - CIS150/50

Using Oracle:
- Installation and login

Links:
- Links & Tutorials

Jayson Rios from the tutoring center is also available on Monday 11-3 and Wednesday 11-1 in Fall River and Tuesday 11-2 in New Bedford.

A group of alum are coming to my CIS120 classes on November 5th to talk about jobs and careers in IT. Please try to come. Note they will come to both the

...do not have the grades

donorcursor3 - Notepad

File Edit Format View Help

```
SET SERVEROUTPUT ON
DECLARE
  v_name       donor.name%TYPE;
  v_state      donor.state%TYPE;
  v_contact    donor.contact%TYPE;
  v_yrgoal     donor.yrgoal%TYPE;
CURSOR donorcursor is
  SELECT name, state, yrgoal,contact into v_name, v_state, v_yrgoal, v_contact  FROM
donor
    WHERE contact IN
      (SELECT contact
         FROM donor
         WHERE state = 'RI')
   OR
      contact IN
      (SELECT contact
         FROM donor
         WHERE yrgoal > 100);

BEGIN
    OPEN donorcursor;
    FETCH donorcursor into v_name, v_state, v_yrgoal, v_contact;
    WHILE donorcursor%FOUND loop
        dbms_output.put_line(v_name || ' ' || v_state || ' ' || v_yrgoal || ' ' ||
v_contact);
        FETCH donorcursor into v_name, v_state, v_yrgoal, v_contact;
    END loop;
    CLOSE donorcursor;
END;
/
SET SERVEROUTPUT OFF
```

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

...r myself to answer

...put procedures (we will go ...t at next week.

Separate speaker notes for More on PL/SQL IF statements

Type here to search

10:30 AM 11/5/2019

Title: Nov 5-10:30 AM (13 of 15)

```
        END LOOP;

        -- Look in manager ID table to determine if employee is a manager
        -- If employee is a manager increase pay by 10% else increase pay by 2%

        FOR v_EmployeeData2 IN c_Employee2 LOOP
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE (v_LoopCnt <= v_ManagerIdTable.COUNT) AND
           (v_FoundFlag = FALSE) LOOP
            IF v_EmployeeData2.employee_id = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = TRUE THEN
            v_Salary := 1.1 * v_EmployeeData2.salary;
          ELSE
            v_Salary := 1.02 * v_EmployeeData2.salary;
          END IF;
          UPDATE employeez
            SET salary = v_Salary
            WHERE CURRENT OF c_Employee2;
        END LOOP;

        COMMIT;
END;
/
```

The logic in this program has changed because the CURSOR FOR LOOP is used. However, the goals and the processing are the same. Notice that there are NO OPEN, CLOSE or FETCH statements in the code. The cursor FOR loop automates the handling of these commands. The syntax of the cursor FOR loop is the FOR followed by the record name (the name of the implicitly declared record) IN followed by the cursor name (the name of the declared cursor) and the word LOOP. This means that the statement of the record name is in essence its definition.

The first FOR loop has v_EmployeeData1 as the implicitly defined record name and c_Employee1 (the first cursor) as the cursor name. The second FOR loop has v_EmployeeData2 as the implicitly defined record name and c_Employee2 (the second cursor) as the cursor name. Again, there are no open and closes statements and every iteration of the loop causes the equivalent of a fetch statement to happen.

Another change in this program is the WHILE does not have an EXIT in it. Instead the while has an AND condition that tests for everything checked AND the flag set to false. If inside the loop the flag gets reset to true, then the next time the WHILE is checked it will not meet both criteria and the loop will terminate. Obviously it will also terminate when all of the elements in the table have been compared to the id.

Notice also the IF statement which compares the implicitly defined record name.column name instead of simply the column name. This is required because the column name is within the record and will only be recognized in the context of the record.

The output below shows a second pay raise given to the employees based on the logic in pay_raise_4.

SQL CODE:

```
SQL> @ pay_raise_4
Input truncated to 1 characters

PL/SQL procedure successfully completed.

SQL> SELECT * FROM employeez;

EMPLOYEE_ID LAST_NAME       FIRST_NAME      MANAGER_ID    SALARY
----------- --------------- --------------- ---------- ---------
       7369 SMITH           JOHN                  7902    832.32
       7499 ALLEN           KEVIN                 7698   1664.64
       7505 DOYLE           JEAN                  7839    3448.5
```

File Edit View History Bookmarks Tools Help

Pay raise examples

www.pgrocer.net/Cis50/payraise.html

```
DECLARE
        TYPE t_ManagerTable IS TABLE OF employeez.manager_id%TYPE
          INDEX BY BINARY_INTEGER;

        v_ManagerIdTable t_ManagerTable;

        CURSOR c_Employee1 IS
          SELECT manager_id FROM employeez;

        CURSOR c_Employee2 IS
          SELECT employee_id, salary FROM employeez
          FOR UPDATE OF salary;

        v_Salary employeez.salary%TYPE;
        v_FoundFlag BOOLEAN;
        v_LoopCnt BINARY_INTEGER;

BEGIN
        -- Build a table of manager ID's

        FOR v_EmployeeData1 IN c_Employee1 LOOP
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE (v_LoopCnt <= v_ManagerIdTable.COUNT) AND
           (v_FoundFlag = FALSE) LOOP
            IF v_EmployeeData1.manager_id = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
            END IF;
              v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = FALSE THEN
            v_ManagerIdTable(v_ManagerIdTable.COUNT + 1) :=
                v_EmployeeData1.manager_id;
          END IF;
        END LOOP;

        -- Look in manager ID table to determine if employee is a manager
        -- If employee is a manager increase pay by 10% else increase pay by 2%

        FOR v_EmployeeData2 IN c_Employee2 LOOP
          v_FoundFlag := FALSE;
          v_LoopCnt := 1;
          WHILE (v_LoopCnt <= v_ManagerIdTable.COUNT) AND
           (v_FoundFlag = FALSE) LOOP
            IF v_EmployeeData2.employee_id = v_ManagerIdTable(v_LoopCnt) THEN
              v_FoundFlag := TRUE;
            END IF;
            v_LoopCnt := v_LoopCnt + 1;
          END LOOP;
          IF v_FoundFlag = TRUE THEN
            v_Salary := 1.1 * v_EmployeeData2.salary;
          ELSE
            v_Salary := 1.02 * v_EmployeeData2.salary;
          END IF;
          UPDATE employeez
            SET salary = v_Salary
            WHERE CURRENT OF c_Employee2;
        END LOOP;

        COMMIT;
```

Type here to search

10:40 AM
11/5/2019