SET SERVEROUTPUT ON;

SET VERIFY OFF;

CLEAR SCREEN;

DECLARE

v\_first\_name VARCHAR2(35);

v\_last\_name VARCHAR2(35);

BEGIN

SELECT first\_name, last\_name

INTO v\_first\_name, v\_last\_name

FROM student

WHERE student\_id = 123;

DBMS\_OUTPUT.PUT\_LINE ('Student name: '||v\_first\_name||' '||

v\_last\_name);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE ('There is no student with '||'student id 123');

END;

.

/

SET SERVEROUTPUT ON

DECLARE

v\_num NUMBER := &sv\_num;

v\_result NUMBER;

BEGIN

v\_result := POWER(v\_num, 2);

DBMS\_OUTPUT.PUT\_LINE ('The value of v\_result is: '||v\_result);

END;

<< outer\_block >>

DECLARE

v\_test NUMBER := 123;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Outer Block, v\_test: ' || v\_test);

<< inner\_block >>

DECLARE

v\_test NUMBER := 456;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Inner Block, v\_test: ' || v\_test);

DBMS\_OUTPUT.PUT\_LINE('Inner Block, outer\_block.v\_test: ' || outer\_block.v\_test);

END inner\_block;

END outer\_block;

DECLARE

v\_average\_cost VARCHAR2(10);

BEGIN

SELECT TO\_CHAR(AVG(cost), '$9,999.99')

INTO v\_average\_cost

FROM course;

DBMS\_OUTPUT.PUT\_LINE('The average cost of a course ' ||

'in the CTA program is ' || v\_average\_cost);

END;

DECLARE

v\_city zipcode.city%TYPE;

BEGIN

SELECT 'COLUMBUS'

INTO v\_city

FROM dual;

UPDATE zipcode

SET city = v\_city

WHERE zip = 43224;

END;

DECLARE

v\_zip zipcode.zip%TYPE;

v\_user zipcode.created\_by%TYPE;

v\_date zipcode.created\_date%TYPE;

BEGIN

SELECT 43438, USER, SYSDATE

INTO v\_zip, v\_user, v\_date

FROM dual;

INSERT INTO zipcode

(zip, created\_by, created\_date, modified\_by, modified\_date)

VALUES(v\_zip, v\_user, v\_date, v\_user, v\_date);

END;

DECLARE

v\_max\_id NUMBER;

BEGIN

SELECT MAX(student\_id)

INTO v\_max\_id

FROM student;

INSERT INTO student

(student\_id, last\_name, zip, created\_by, created\_date, modified\_by, modified\_date, registration\_date)

VALUES(v\_max\_id + 1, 'Loblaw', 11238, 'bloblaw', '5-JAN-2015', 'bloblaw', '5-JAN-2015', '5-JAN-2015');

END;

CREATE TABLE test01

(col01 NUMBER);

CREATE SEQUENCE test\_seq

INCREMENT BY 5;

INSERT INTO test01

VALUES(test\_seq.NEXTVAL);

SELECT \* FROM test01;

BEGIN

INSERT INTO student

(student\_id, last\_name, zip, registration\_date, created\_by, created\_date, modified\_by, modified\_date)

VALUES(student\_id\_seq.NEXTVAL, 'Tashi', 10015, '1-JAN-2015', 'STUDENTA', '1-JAN-2015', 'STUDENTA', '1-JAN-2015');

SAVEPOINT A;

INSERT INTO student

(student\_id, last\_name, zip, registration\_date, created\_by, created\_date, modified\_by, modified\_date)

VALUES(student\_id\_seq.NEXTVAL, 'Sonam', 10015, '1-JAN-2015', 'STUDENTB', '1-JAN-2015', 'STUDENTB', '1-JAN-2015');

SAVEPOINT B;

INSERT INTO student

(student\_id, last\_name, zip, registration\_date, created\_by, created\_date, modified\_by, modified\_date)

VALUES(student\_id\_seq.NEXTVAL, 'Norbu', 10015, '1-JAN-2015', 'STUDENTB', '1-JAN-2015', 'STUDENTB', '1-JAN-2015');

SAVEPOINT C;

ROLLBACK TO B;

END;

DECLARE

v\_counter NUMBER := 0;

BEGIN

LOOP

DBMS\_OUTPUT.PUT\_LINE ('v\_counter = '||v\_counter);

EXIT;

END LOOP;

END;

DECLARE

v\_counter BINARY\_INTEGER := 0;

BEGIN

LOOP

-- increment loop counter by one

v\_counter := v\_counter + 1;

DBMS\_OUTPUT.PUT\_LINE ('v\_counter = '||v\_counter);

-- if EXIT condition yields TRUE exit the loop

IF v\_counter = 5 THEN

EXIT;

END IF;

END LOOP;

-- control resumes here

DBMS\_OUTPUT.PUT\_LINE ('Done…');

END;

DECLARE

v\_course course.course\_no%TYPE := 430;

v\_instructor\_id instructor.instructor\_id%TYPE := 102;

v\_sec\_num section.section\_no%TYPE := 0;

BEGIN

LOOP

-- increment section number by one

v\_sec\_num := v\_sec\_num + 1;

INSERT INTO section

(section\_id, course\_no, section\_no, instructor\_id,

created\_date, created\_by, modified\_date,

modified\_by)

VALUES

(section\_id\_seq.NEXTVAL, v\_course, v\_sec\_num,

v\_instructor\_id, SYSDATE, USER, SYSDATE, USER);

-- if number of sections added is four exit the loop

EXIT WHEN v\_sec\_num = 4;

END LOOP;

-- control resumes here

COMMIT;

END;

DECLARE

v\_counter NUMBER := 5;

BEGIN

WHILE v\_counter < 5 LOOP

DBMS\_OUTPUT.PUT\_LINE ('v\_counter = '||v\_counter);

-- decrement the value of v\_counter by one

v\_counter := v\_counter - 1;

END LOOP;

END;

DECLARE

v\_counter NUMBER := 1;

BEGIN

WHILE v\_counter <= 5 LOOP

DBMS\_OUTPUT.PUT\_LINE ('v\_counter = '||v\_counter);

IF v\_counter = 2 THEN

EXIT;

END IF;

v\_counter := v\_counter + 1;

END LOOP;

END;

BEGIN

FOR v\_counter IN REVERSE 1..5 LOOP

DBMS\_OUTPUT.PUT\_LINE ('v\_counter = '||v\_counter);

END LOOP;

END;

BEGIN

FOR v\_counter IN 1..5 LOOP

DBMS\_OUTPUT.PUT\_LINE ('v\_counter = '||v\_counter);

EXIT WHEN v\_counter = 3;

END LOOP;

END;

SET SERVEROUTPUT ON

DECLARE

vr\_student student%ROWTYPE;

BEGIN

SELECT \*

INTO vr\_student

FROM student

WHERE student\_id = 156;

DBMS\_OUTPUT.PUT\_LINE (vr\_student.first\_name||' '||vr\_student.last\_name||' has an ID of 156');

EXCEPTION

WHEN no\_data\_found

THEN

RAISE\_APPLICATION\_ERROR(-2001,'The Student '||'is not in the database');

END;

SET SERVEROUTPUT ON;

DECLARE

CURSOR c\_student\_name IS

SELECT first\_name, last\_name

FROM student

WHERE rownum <= 5;

vr\_student\_name c\_student\_name%ROWTYPE;

BEGIN

OPEN c\_student\_name;

LOOP

FETCH c\_student\_name INTO vr\_student\_name;

EXIT WHEN c\_student\_name%NOTFOUND;

END LOOP;

CLOSE c\_student\_name;

DBMS\_OUTPUT.PUT\_LINE('Student name: '||

vr\_student\_name.first\_name||’ '

||vr\_student\_name.last\_name);

END;

SET SERVEROUTPUT ON;

DECLARE

TYPE instructor\_info IS RECORD

(first\_name instructor.first\_name%TYPE,

last\_name instructor.last\_name%TYPE,

sections NUMBER);

rv\_instructor instructor\_info;

BEGIN

SELECT RTRIM(i.first\_name), RTRIM(i.last\_name), COUNT(\*)

INTO rv\_instructor

FROM instructor i, section s

WHERE i.instructor\_id = s.instructor\_id

AND i.instructor\_id = 102

GROUP BY i.first\_name, i.last\_name;

DBMS\_OUTPUT.PUT\_LINE('Instructor, '||

rv\_instructor.first\_name||

' '||rv\_instructor.last\_name||

', teaches '||rv\_instructor.sections||

' section(s)');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE

('There is no such instructor');

END;

**CURSOR ATTRIBUTE SYNTAX DESCRIPTION**

%NOTFOUND *cursor\_name*%NOTFOUND A Boolean attribute that returns TRUE if

the previous FETCH did not return a row

and FALSE if it did.

%FOUND *cursor\_name*%FOUND A Boolean attribute that returns TRUE if

the previous FETCH returned a row and

FALSE if it did not.

%ROWCOUNT *cursor\_name*%ROWCOUNT The number of records fetched from a

cursor at that point in time.

%ISOPEN *cursor\_name*%ISOPEN A Boolean attribute that returns TRUE if

the cursor is open and FALSE if it is not.

EXIT WHEN c\_student%NOTFOUND;

END LOOP;

CLOSE c\_student;

EXCEPTION

WHEN OTHERS

THEN

IF c\_student%ISOPEN

THEN

CLOSE c\_student;

END IF;

END;

SET SERVEROUTPUT ON

DECLARE

CURSOR c\_student\_enroll IS

SELECT s.student\_id, first\_name, last\_name,

COUNT(\*) enroll,

(CASE

WHEN count(\*) = 1 Then ' class.'

WHEN count(\*) is null then

' no classes.'

ELSE ' classes.'

END) class

FROM student s, enrollment e

WHERE s.student\_id = e.student\_id

AND s.student\_id <110

GROUP BY s.student\_id, first\_name, last\_name;

r\_student\_enroll c\_student\_enroll%ROWTYPE;

BEGIN

OPEN c\_student\_enroll;

LOOP

FETCH c\_student\_enroll INTO r\_student\_enroll;

EXIT WHEN c\_student\_enroll%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Student INFO: ID '||

r\_student\_enroll.student\_id||' is '||

r\_student\_enroll.first\_name|| ' ' ||

r\_student\_enroll.last\_name||

' is enrolled in '||r\_student\_enroll.enroll||

r\_student\_enroll.class);

END LOOP;

CLOSE c\_student\_enroll;

EXCEPTION

WHEN OTHERS

THEN

IF c\_student\_enroll %ISOPEN

THEN

CLOSE c\_student\_enroll;

END IF;

END;

DECLARE

CURSOR c\_group\_discount IS

SELECT DISTINCT s.course\_no

FROM section s, enrollment e

WHERE s.section\_id = e.section\_id

GROUP BY s.course\_no, e.section\_id, s.section\_id

HAVING COUNT(\*)>=8;

BEGIN

FOR r\_group\_discount IN c\_group\_discount LOOP

UPDATE course

SET cost = cost \* .95

WHERE course\_no = r\_group\_discount.course\_no;

END LOOP;

COMMIT;

END;

DECLARE

v\_zip zipcode.zip%TYPE;

v\_student\_flag CHAR;

CURSOR c\_zip IS

SELECT zip, city, state

FROM zipcode

WHERE state = 'CT';

CURSOR c\_student IS

SELECT first\_name, last\_name

FROM student

WHERE zip = v\_zip;

BEGIN

FOR r\_zip IN c\_zip

LOOP

v\_student\_flag := 'N';

v\_zip := r\_zip.zip;

DBMS\_OUTPUT.PUT\_LINE(CHR(10));

DBMS\_OUTPUT.PUT\_LINE('Students living in '||

r\_zip.city);

FOR r\_student in c\_student

LOOP

DBMS\_OUTPUT.PUT\_LINE(

r\_student.first\_name||

' '||r\_student.last\_name);

v\_student\_flag := 'Y';

END LOOP;

IF v\_student\_flag = 'N'

THEN

DBMS\_OUTPUT.PUT\_LINE

('No Students for this zipcode');

END IF;

END LOOP;

END;

DECLARE

v\_sid student.student\_id%TYPE;

CURSOR c\_student IS

SELECT student\_id, first\_name, last\_name

FROM student

WHERE student\_id < 110;

CURSOR c\_course IS

SELECT c.course\_no, c.description

FROM course c, section s, enrollment e

WHERE c.course\_no = s.course\_no

AND s.section\_id = e.section\_id

AND e.student\_id = v\_sid;

BEGIN

FOR r\_student IN c\_student

LOOP

v\_sid := r\_student.student\_id;

DBMS\_OUTPUT.PUT\_LINE(chr(10));

DBMS\_OUTPUT.PUT\_LINE(' The Student '||

r\_student.student\_id||' '||

r\_student.first\_name||' '||

r\_student.last\_name);

DBMS\_OUTPUT.PUT\_LINE(' is enrolled in the '||

'following courses: ');

FOR r\_course IN c\_course

LOOP

DBMS\_OUTPUT.PUT\_LINE(r\_course.course\_no||

' '||r\_course.description);

END LOOP;

END LOOP;

END;